Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет «Высшая школа экономики»
Московский институт электроники и математики
имени А.Н. Тихонова

На правах рукописи

Чеповский Александр Андреевич

МЕТОДЫ РАБОТЫ С НЕЯВНЫМИ СООБЩЕСТВАМИ НА ВЗВЕШЕННЫХ ГРАФАХ ВЗАИМОДЕЙСТВУЮЩИХ ОБЪЕКТОВ

Специальность 1.2.2 –
«Математическое моделирование, численные методы и комплексы программ»

Диссертация на соискание ученой степени
доктора физико-математических наук

Научный консультант:
доктор физико-математических наук, профессор,
академик РАН Сигов Александр Сергеевич

Москва – 2024

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. Системы, представляющие собой сетевые структуры, образованные взаимодействием между собой большого числа объектов, принято объединять под термином «сложные сети» (complex networks). Это, например, биологические, экологические, инфраструктурные, технологические, социальные сети. В качестве математической модели для таких сетей рассматриваются графы, в которых вершины соответствуют узлам сети, а ребра – связям между ними. При этом, как вершины, так и ребра могут обладать некоторой информацией, которая фиксируется как атрибуты соответствующих элементов множества вершин или множества ребер графа. Таким образом, получаются графы взаимодействующих объектов, анализ которых является существенной проблемой в области информационных технологий.

Актуальность рассматриваемой проблемы определяется «Стратегией национальной безопасности Российской Федерации», утвержденной Указом Президента РФ № 400 от 2 июля 2021 г., а именно задачами развития безопасного информационного пространства, защиты российского общества от деструктивного информационно-психологического воздействия. В частности, это задачи создания условий для эффективного предупреждения, выявления и пресечения преступлений и иных правонарушений, совершаемых с использованием информационно-коммуникационных технологий: задачи развития сил и средств информационного противоборства; задачи противодействия использованию информационной инфраструктуры Российской Федерации экстремистскими и террористическими организациями, специальными службами и пропагандистскими структурами иностранных государств для осуществления деструктивного информационного воздействия на граждан и общество.

Для информационно-аналитических систем важной составляющей является анализ графов взаимодействующих объектов, полученных из сетей передачи дан-

ных. Данный анализ есть существенная составляющая управления информационным пространством и аналитическими подсистемами, применяемыми для обеспечения безопасности и контроля деятельности по распространению информации.

Решение данной проблемы имеет существенное значение в рамках современных разведывательных и контрразведывательных мероприятий, необходимости оценки источников размещения информации в социальных сетях, мессенджерах и выявления групп субъектов, использующих и активно поддерживающих данную информацию. Данные задачи актуальны для аналитических подразделений спецслужб и коммерческих структур, решающих как маркетинговые задачи, так и задачи борьбы с мошенничеством.

Анализ графов взаимодействующих объектов, включая построение методов выделения ключевой информации и разработку прикладного программного обеспечения для обработки данных, является важной составляющей для создания информационно-аналитических систем обеспечения безопасности. В частности, при работе с графами, полученными при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями, особую ценность представляют следующие прикладные задачи: определение близости профилей пользователей, совпадения их интересов, степени (очного) знакомства; выявление наиболее активных единомышленников среди контактов заданного исходного пользователя, возможно напрямую с ним и не связанных; распознавание лидеров мнений; выявление каналов распространения и обмена информации между пользователями.

С точки зрения построения информационно-аналитических систем проблема анализа реальных графов взаимодействующих объектов влечет за собой необходимость решения следующих задач. Это разработка алгоритмов для выявления структуры графа; создание методик анализа сформированных данных, включая оценку корректности полученных результатов; программная реализация средств обработки графов больших размеров, включая создание специализированных эффективных графовых хранилищ.

Последние 20 лет в области методов анализа структуры графа ведутся активные исследования по разработке алгоритмов выделения неявных сообществ на графах. Под выделением неявных сообществ на графе понимается разбиение графа на подграфы, такое что плотность связей внутри этих подграфов значительно выше плотности связей между ними. Такое разбиение позволяет, в частности, переходить к выделению различных ролей у вершин графа. Один из самых интуитивно понятных и распространенных подходов к решению задачи выделения сообществ состоит в алгоритмах поиска разбиения графа на основе максимизации некоторого функционала, характеризующего качество разбиения и обычно называемого «модулярность». Различные аспекты выделения сообществ на графах рассматривали Newman M.E.J., Girvan M., Fortunato S., Blondel V. D. и другие авторы. Распространен также алгоритм на основе имитации условного динамического процесса на графе (Rosvall M., Bergstrom C. T.). При этом указанные наиболее развитые подходы не решают в полном объеме задачу выделения пересекающихся сообществ на графах взаимодействующих объектов, полученных из реальных данных о социальных коммуникациях, для которых характерны и играют существенную роль атрибуты ребер и вершин.

Наиболее спорной и практически открытой является проблема оценки корректности и эффективности работы алгоритмов и методов выделения сообществ на графах. Существует множество методов генерации случайных графов с последующим тестированием на них алгоритмов для получения оценки разбиения на сообщества (Lancichinetti A., Fortunato S., Radicchi F.). Есть иные методы, основанные на анализе разбиений анализируемого графа на основе оценки количества информации (Danon L., Díaz-Guilera A., Amelio A., Duch J., Arenas A., Pizzuti C.). Данные подходы не позволяют корректно оценить результаты работы с графами реальных сетей, особенно для графов, полученных при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями.

Среди российских авторов работы, связанные с графами сложных сетей, в основном, носят либо обзорный характер (И.А. Евин, Н.Ф. Гусарова, Н.Г. Щербакова и др.), либо относятся к вопросам моделирования случайных графов и построения

прогнозов их развития (А.М. Райгородский, В.Н. Задорожный, В.А. Бадрызлов и др.) и моделирования распространения информации в социальных сетях (Д.А. Губанов, Д.А. Новиков, А.Г. Чхартишвили).

Таким образом, актуальными являются разработка методов и алгоритмов выделения сообществ на реальных графах взаимодействующих объектов, формирование принципов тестирования алгоритмов, создание прикладного программного обеспечения, реализующего разработанные методы.

Объект исследования – графы взаимодействующих объектов, полученные из сетей передачи данных.

Предмет исследования. Предметом исследований являются разработка алгоритмов выделения сообществ на графах различной природы и моделирование методов формирования графов, полученных при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями, включающих атрибутные данные объектов и их взаимодействия.

Целью диссертационной работы является решение имеющей важное хозяйственное и социально-экономическое значение проблемы анализа коммуникационных данных, служащей существенным фактором в обеспечении технических и технологических подходов в сфере государственной и общественной безопасности, включая вопросы контроля информационного воздействия в социальных сетях и сетях мгновенного обмена сообщениями. Расширение средств и возможностей указанного контроля позволяет повысить безопасность информационного пространства, защиту общества от деструктивного информационно-психологического воздействия. Такое расширение включает в себя создание моделей, разработку численных методов и программного обеспечения для анализа структуры графов взаимодействующих объектов, полученных при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями с целью описания информационного взаимодействия объектов.

Научная проблема, имеющая важное хозяйственное значение, состоит в создании методов, моделей и программного обеспечения для анализа структуры

графов взаимодействующих объектов, полученных при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями, с целью оценки информационного воздействия на субъектов через коммуникационные ресурсы. Решение данной проблемы должно предоставить средства для широкого класса прикладных задач в сфере анализа данных коммуникаций, обеспеченных различными компьютерными технологиями. Указанные средства повышают возможности эффективного предупреждения, выявления и пресечения преступлений, предотвращению поддержки экстремистской и террористической деятельности, и иных правонарушений, совершаемых с использованием информационно-коммуникационных технологий.

Решение проблемы включает следующие задачи:

1. Построение для сетей передачи данных моделей графов взаимодействующих объектов, описывающих их информационное взаимодействие;
2. Построение и разработка итерационных численных методов и универсальных алгоритмов для выделения неявных сообществ и ключевых вершин графов с использованием эвристик;
3. Создание процедур по оценке качества выявленных сообществ на графе взаимодействующих объектов;
4. Разработка программного обеспечения, реализующего методы хранения, анализа и визуализации графов взаимодействующих объектов для сетей передачи данных.

Методы исследования. Для решения сформулированных проблем и поставленных задач использовалась методология информационного моделирования, аппарат решения экстремальных задач теории графов, и вычислительных методов оптимизации функционалов.

На защиту выносятся следующие научные результаты:

1. Модель формирования графа взаимодействующих объектов, полученного при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями, описывающего информационное воздействие на объекты сети.

2. Итерационные численные методы для выделения неявных сообществ и ключевых вершин графов с использованием эвристик, а именно:
   2.1. «Комбинированный алгоритм» для выделения пересекающихся сообществ на графе, позволяющий убирать из рассмотрения малозначимые элементы сети и предусматривающий параметрические модификации для формирования разнородных разбиений в зависимости от задач оператора.
   2.2. «Метод ядра» для выделения непересекающихся сообществ на взвешенных графах, предусматривающий выделение ключевой компоненты на основании вычисляемых в явном виде характеристик графа.
   2.3. «Метод Галактик» выделения пересекающихся сообществ на взвешенных графах, основанный на последовательном выделении сообществ и обработке исходного графа, переходе к мета-графу из мета-сообществ и последующем выделении итоговых пересекающихся сообществ.
3. Методика оценки корректности выделения пересекающихся сообществ на графах взаимодействующих объектов, основанная на анализе методами компьютерной лингвистики текстов – атрибутов вершин выделенных сообществ.
4. Программное обеспечение для построения графа взаимодействующих объектов (импорта данных из сетей коммуникации) и его последующего анализа, включающее:
   - средства хранения графов;
   - реализации алгоритмов выделения сообществ;
   - средства графического отображения обнаруженной структуры графа.

Перечисленные научные результаты представляют основу нового научного направления «комплексный анализ структуры графов взаимодействующих объектов», сочетающего построение и анализ структуры соответствующих графов с целью описания информационного взаимодействия, в том числе в социальных сетях и сетях мгновенного обмена сообщениями.

Научная новизна:

- предложена модель формирования взвешенного графа взаимодействующих объектов при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями, характеризующая информационное взаимодействие;

- построены итерационные числительные методы и алгоритмы для выделения неявных сообществ и ключевых вершин графов с использованием эвристик;
- предложен и реализован «Комбинированный алгоритм» выделения пересекающихся и вложенных сообществ на графе, позволяющий убирать из рассмотрения малозначимые элементы сети;
- предложен и реализован «Метод ядра» для выделения непересекающихся сообществ на взвешенных графах, предусматривающий выделение ключевой компоненты на основании вычисляемых в явном виде характеристиках графа;
- предложен и реализован «Метод Галактик» для выделения пересекающихся сообществ на взвешенных графах;
- разработана методика оценки эффективности выделения сообществ на графе с помощью алгоритмов компьютерной лингвистики для обработки текстовых метаданных – атрибутов вершин выделенных сообществ;
- разработана модель для эффективного хранения графов взаимодействующих объектов, основанная на алгоритмах сжатия и оптимизации операций с графами по памяти и по скоростным характеристикам.

Теоретическая значимость результатов работы состоит в том, что предложены новые методы и алгоритмы выделения неявных сообществ на графе взаимодействующих объектов, опирающиеся на структурные особенности графа, и в том, что предложены принципиально новые методы для оценки качества решения задачи по выделению сообществ.

Практическая значимость состоит в построении методов и алгоритмов для решения проблемы, имеющей важное хозяйственное значение, формировании новых подходов для анализа получаемых решений и реализации в комплексе программ. Получены свидетельства о регистрации программ для ЭВМ. Научные и практические результаты диссертации использованы в 2016 – 2021 годах в грантах РФФИ (участник грантов РФФИ): 16-07-00641 А. «Исследование и разработка математических моделей, методов и алгоритмов визуализации и анализа графов на примере социальных сетей». (2016-2018 г.г.); 16-29-09546 офим-м. «Разработка но-

вых методов мониторинга и комплексного лингвистического и тематического анализа сообщений социальных медиа в целях противодействия экстремизму и терроризму». (2016-2019 г.г.); 18-00-00233 КОМФИ. «Методы комплексного интеллектуального анализа информации различных типов для социо-гуманитарных исследований в социальных медиа». (2018-2020 г.г.); 19-07-00806 А. «Исследование и разработка методов и алгоритмов для создания и комплексного лингвистического анализа специализированных корпусов текстов». (2019-2021 г.г.). Результаты диссертации использованы в учебных пособиях.

Достоверность результатов и обоснованность научных положений диссертационной работы обеспечивается корректным использованием методов теории графов и соответствующего математического аппарата. Достоверность полученных выводов подтверждается согласованностью с результатами экспериментальных исследований и экспертной оценкой результатов. Достоверность результатов работы подтверждается работоспособностью предложенных методик в проведенных экспериментальных вычислениях с помощью разработанного программного обеспечения. Результаты прошли апробацию на международных и российских конференциях, принимались к публикации в рецензируемых журналах.

Апробация работы. Материалы отдельных разделов диссертации докладывались на международных конференциях:

– Международная конференция «Математика в созвездии наук» К юбилею ректора МГУ, академика В.А. Садовничего, 1-2 апреля 2024 года. Секция «Дискретная математика, математическая кибернетика и теория интеллектуальных систем»;

– COMPLEX NETWORKS 2020. The 9th International Conference on Complex Networks and their Applications. December 1-3, 2020 - Madrid, Spain.;

– VII Международная научно-практическая конференция «Управление информационной безопасностью в современном обществе», Москва, 29 мая - 30 мая 2019 г.;

– The 6th International Conference on Complex Networks &Their Applications. Nov. 29 - Dec. 01, 2017, Lyon (France);

– V Международная научно-практическая конференция «Управление информационной безопасностью в современном обществе», Москва, 30 мая - 1 июня 2017 г.;

– IV Международная конференция «Управление информационной безопасностью в современном обществе», Москва, 31 мая - 02 июня 2016, Москва.

– 5 Международная конференция «Ситуационные центры и геоинформационные системы для задач мониторинга и безопасности (SCVRT 2016)», 21-25 ноября 2016, г. Пущино, Московская обл.;

– Международная научная конференция Resilience2014 Международного Центра по ядерной безопасности Института физико-технической информатики, Протвино, 2014 г.;

– Международная научная конференция по физико-технической информатике CPT2014. Протвино, 2014;

– Международная научная конференция Международного центра по ядерной безопасности Института физико-технической информатики SCVRT2013, Протвино, 25-29 ноября 2013 г.

Материалы диссертационных исследований обсуждались на семинарах МИЭМ НИУ ВШЭ, научно-исследовательской лаборатории «Суперкомпьютерные технологии и машинное обучение» СПбПУ (май 2023 г.), Института перспективных технологий и индустриального программирования МИРЭА (март 2023 г.), отделения интеллектуальных кибернетических систем ИАТЭ НИЯУ «МИФИ» (апрель 2023 г.), кафедры Теоретической информатики механико-математического факультета МГУ имени М.В. Ломоносова (ноябрь 2023 г.).

Публикации. Основные результаты диссертационного исследования опубликованы в 37 работах. Из них 22 статьи в ведущих рецензируемых научных журналах, которые входят в утвержденный ВАК Минобрнауки России «Перечень российских рецензируемых научных журналов, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученых степеней доктора и кандидата наук» по специальности 1.2.2 (05.13.18) и приравненных к ним зарубежных рецензируемых изданиях (Scopus) и входящих в базу RSCI (на платформе

Web of Science). В том числе 12 статей в журналах категории К1 и К2 по распределению на 2022 и 2023 годы. Две рецензируемые монографии, 13 публикаций в трудах международных научных конференций. Два свидетельства о регистрации программ для ЭВМ. Результаты диссертации использованы в 2 учебных пособиях. Основные научные результаты получены автором самостоятельно. Результаты, полученные соискателем лично, в работах, опубликованных в соавторстве, представляют собой разработку моделей формирования взвешенного графа информационного взаимодействия для разных сетей; разработку алгоритмов, методику работы с ними и анализ результатов их применения; разработку структур данных для хранения и анализа графов. Другим соавторам принадлежат программные реализации моделей и алгоритмов, анализ данных на сторонних программных продуктах, сопровождение разработанного программного обеспечения.

Соответствие специальности. Направление диссертационного исследования соответствует паспорту специальности 1.2.2. «Математическое моделирование, численные методы и комплексы программ», а именно, п. 1 «Разработка новых математических методов моделирования объектов и явлений», п. 2 «Разработка, обоснование и тестирование эффективных вычислительных методов с применением современных компьютерных технологий» и п. 8 «Комплексные исследования научных и технических проблем с применением современной технологии математического моделирования и вычислительного эксперимента».

Структура и объем работы. Диссертация состоит из введения, семи глав, выводов, библиографического списка, включающего в себя 180 наименований, и 2 приложений. Работа содержит 205 страниц машинописного текста основной части, включающей 78 рисунков, 51 таблицу и 19 страниц библиографии. Приложения содержат 12 страниц машинописного текста.

ГЛАВА 1 НЕЯВНЫЕ СООБЩЕСТВА НА ГРАФАХ ВЗАИМОДЕЙСТВУЮЩИХ ОБЪЕКТОВ

1.1 Графы взаимодействующих объектов в разных предметных областях

Многие системы, состоящие из большого числа объектов, можно изучать, представляя их как сетевые структуры, образованные взаимодействием их элементов между собой. Речь идет, например, о биологических, экологических, технологических, инфраструктурных, социальных сетях. Такие сетевые структуры принято называть «сложные сети» (complex networks). Наиболее интуитивно понятной математической моделью для таких сетей являются графы. Узлы исходной сети представляются вершинами, а связи между узлами характеризуют ребра, инцидентные соответствующим вершинам. Важным аспектом при построении таких графов является то, что вершины, как и ребра могут обладать некоторой информацией, которая представляется в виде их атрибутов, в том числе и текстовых. Таким образом, получаются графы взаимодействующих объектов.

Анализ построенных графов взаимодействующих объектов является актуальной на сегодняшний день проблемой, в ходе исследования которой возникают задачи построения алгоритмов и методов выделения ключевой информации, разработки прикладного программного обеспечения для обработки данных.

Существенный интерес к методам анализа графов взаимодействующих объектов наблюдается в предметных областях, связанных с вопросами информационной безопасности, криминальных расследований, борьбе с экстремизмом и терроризмом. Так же интерес к таким методам имеется и для задач, возникающих в биологии, экологии, экономике, социологии, маркетинге и многих других. Проблемы из таких разнообразных предметных областей объединяет то, что в реальных задачах исследуемые модели сводятся к описанию сетевого взаимодействия, требующего анализ устройства достаточно больших по размеру и сложных по структуре гра-

фов [1]. Для графов, представляющих реальные сети, часто можно выделить подграфы с высокой плотностью ребер внутри них и сравнительно низкой плотностью ребер между такими подграфами. Разбиение графа на такие подграфы и называется выделением неявных сообществ [1, 2, 3, 4].

При этом определение сообщества может варьироваться в зависимости от предметной области и даже решаемой задачи, тут не существует единственно верного и однозначного определения. Вместе с тем, как и было обозначено выше, разбиение графа взаимодействующих объектов на неявные сообщества характеризуется как раз высокой плотностью ребер внутри сообществ и сравнительно низкой плотностью ребер между ними (рисунок 1.1). Случай, если вершина не состоит ни в одном сообществе, обычно сводится к тому, что такая вершина состоит в своем собственном сообществе из одной вершины.

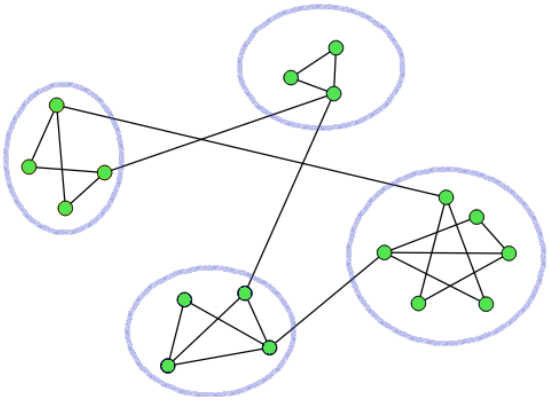

Рисунок 1.1 – Граф, на котором выделены непересекающиеся сообщества.

Особенность устройства некоторых сетей заключается в том, что на их графах взаимодействующих объектов могут быть выделены неявные пересекающиеся сообщества. Это означает, что вершина может принадлежать одновременно более, чем одному сообществу. Для некоторых сетей такая структура сообществ на графе более естественна, чем структура непересекающиеся сообществ, ибо лучше отражает роль отдельных объектов. Это имеет место, например, для графа информационного взаимодействия сети *Telegram*-каналов, что подробнее изложено в главе 5. Граф с выделенными на нем пересекающимися сообществами продемонстрирован для наглядности на рисунке 1.2.

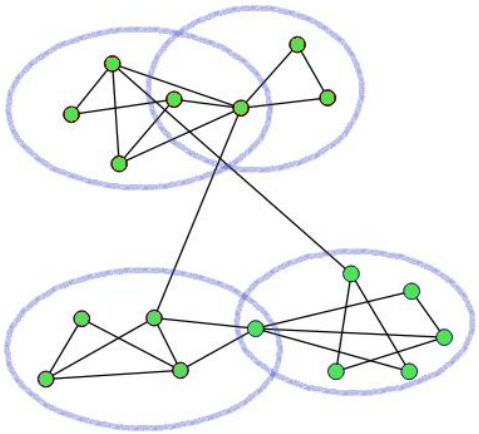

Рисунок 1.2 – Граф, на котором выделены пересекающиеся сообщества.

Коммуникационная сеть может быть представлена как граф взаимодействующих объектов. Сеть контактов абонентов мобильной связи, которая обычно обозначается термином «биллинг», может представлять из себя как «классический» биллинг, включающий звонки и SMS-сообщения абонентов, так и соединения в рамках мобильного Интернета. Анализ биллинга является актуальной задачей информационных технологий и задач информационной безопасности [5, 6, 7]. Задачи обеспечения информационной безопасности информационно-коммуникационных сетей рассматриваются в [8, 9].

Одной из задач анализа биллинга является использование его результатов для раскрытия преступлений [6]. Целью такого анализа соединений между абонентами в следственной ситуации, когда фигуранты дела неизвестны, является как геопозиционирование абонентов, так и очерчивание круга абонентов, которые потенциально могут иметь отношение к совершенному преступлению. Для решения таких задач полезно выделять неявные сообщества на графе фактических коммуникаций абонентов.

Биологические сетевые структуры могут представлять собой, например, пищевые сети, метаболические сети, белок-белковые сети взаимодействия [10]. Анализ биомолекулярных сетей актуален для понимания молекулярного механизма биологических систем, в диагностике, лечении и разработке лекарств для сложных заболеваний или расстройств [11].

Важно так же отметить применение сетевого анализа как набора инструментов для решения задач реинжиниринга программного обеспечения при проектировании информационных систем [12, 13].

Встречаются и иные задачи, для которых применяется анализ структуры графа взаимодействующих объектов, в том числе включающий в себя выделение неявных сообществ. Например, для оценки социально-экономических моделей городского хозяйства с целью инфраструктурного и транспортного планирования, разработки политики в области недвижимости и социально-экономического развития [14].

В работе [15] сетевые методы анализа применяются для описания геофизических транспортных процессов (например, океанические или атмосферные циркуляции), продемонстрирована их эффективность в оценке транспортировки и смешивания течений в геофизических расчетах.

В качестве примера использования сетей для маркетинговых целей можно привести работу [16], в которой рассматриваются подходы по проектированию сети клиентов на основе рейтинговых данных участников электронной коммерции.

Сетевые структуры могут быть построены на основе данных с сайтов, позволяющих своим пользователям взаимодействовать друг с другом, например, по профессиональной направленности. В таких случаях выделение неявных сообществ на основе взаимодействия пользователей актуально для достижения разных задач. Так, в [17] приведен пример анализа части сети *LinkedIn* и выявления сообществ среди практиков в сфере социальной работы. Часто сетевые структуры с последующим выделением в них сообществ строятся на основании данных с сайтов размещения вакансий и поиска работников [18].

Всемирную информационно-телекоммуникационную сеть Интернет можно интерпретировать как граф, вершинами которого являются сайты, а ребрами – гиперссылки между ними. Моделирование сети Интернет в общем случае связано с описанием взаимодействия с помощью разреженных графов, что позволяет разным авторам рассматривать в первую очередь модели случайных графов для описания такого взаимодействия [1, 19]. Возможно проводить исследование графов, построенных на основе отдельных тематических фрагментов сети Интернет. Например, в

работах [20, 21] рассматриваются сайты академических учреждений РАН и их разбиение на сообщества.

Еще одной задачей, возникающей при анализе сетевых структур, является поиск часто встречающихся подграфов в заданном графе, что представляет теоретический интерес для задач обнаружения групп атак в социальных сетях [22, 23]. Так же актуальна и постановка схожей задачи поиска часто встречающихся вхождений шаблонов в базах данных, содержащих много небольших графов.

Построение на основе реальных сложных сетей графов взаимодействующих объектов с последующим выделением на них неявных сообществ позволяет провести в том числе и визуальный этап исследования сети. На рисунке 1.3 приведён пример визуализации графа, полученного при импорте данных из социальной сети *ВКонтакте,* на котором выделены неявные непересекающиеся сообщества. На этой иллюстрации хорошо заметно наличие высокой плотности ребер внутри сообществ и низкой плотности между сообществами.

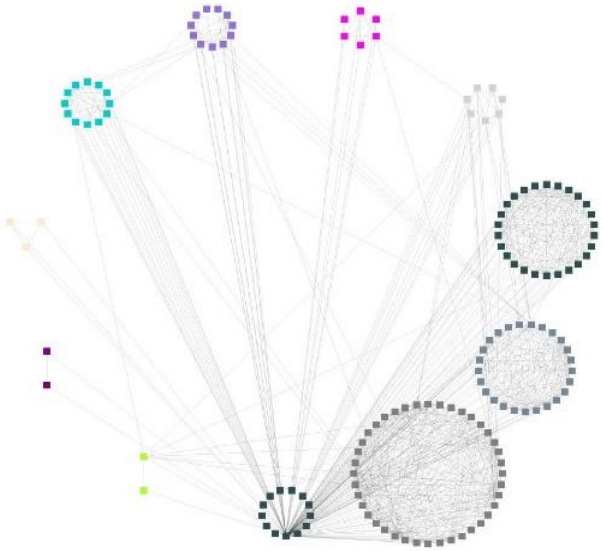

Рисунок 1.3 – Пример изображения сети, разбитой на сообщества.

Выделение сообществ можно рассматривать как базовый, первый этап исследования графа взаимодействующих объектов, для логического завершения которого желательно удобное визуальное представление результатов. Последующий

анализ позволяет перейти к численному и качественному исследованию самих сообществ, а также, в случае сетей коммуникации, к анализу информационного воздействия на группы людей.

1.2 Задачи анализа графов социальных сетей

Одним из самых актуальных направлений исследований сетевых систем является анализ социальных сетей, в том числе онлайн социальных сетей [3, 4, 24-34]. При построении графов тут в качестве вершин обычно рассматриваются аккаунты пользователей. Ребра и вес на них (при необходимости) определяются, как правило, действиями пользователей каждой конкретной социальной сети: отношениями «дружбы», «подписки», простановкой «лайков» или иными подобными средствами коммуникации. В результате получается граф взаимодействующих объектов. Структура построенных на основе реальных данных графов сложных сетей, к которым относятся и графы социальных сетей, имеет свои особенности. Для таких графов характерны в том числе следующие свойства: маленький диаметр графа (эффект «малого мира»), высокие значения кластерного коэффициента (эффект «транзитивности»), медленно спадающее распределение вершин по их степеням («большой хвост распределения»), структура неявных сообществ [1, 2, 3, 33].

Замечено, что ассимптотическая зависимость распределения вершин для больших степеней («в хвосте») близка к степенному закону. То есть у небольшого числа вершин в таких сетях высокие показатели их степени. Поэтому иногда в литературе такие сети называются безмасштабными (scale-free networks), так как распределение сохраняется и в локальных фрагментах – при «изменении масштаба». Это, в свою очередь, связано с тем, что степенная функция является единственным решением соответствующего функционального уравнения [33, 35].

Графы социальных сетей обладают содержательной структурой неявных сообществ, представляющих группы пользователей, объединенных либо активной коммуникацией друг с другом, либо по набору дополнительных признаков (напри-

мер, это могут быть однокурсники, подписчики определенных тематических сайтов). Тут возникают задачи выделения неявных сообществ, выявления лидеров мнений, исследований в части распространения информации по сети.

Построение моделей распространения информации и информационного влияния в социальных сетях рассматривалось в [26, 27]. В этих работах предложены методы описания информационных потоков и психологических операций для задач информационного противоборства. Обзор инструментария для анализа распространения информации и информационного влияния в социальных сетях, включая выделение неявных сообществ, можно найти в работах [9, 26, 27].

Задачи выявления путей распространения информации, повышения уровня защищенности от информационно-психологического воздействия в социальных сетях, в том числе выявление источников и ретрансляторов информации освещены в [36]. В данной работе авторы приводят методику построения графа распространения информации, но далее построенный так граф предлагается рассматривать методами визуального анализа, что накладывает определенные ограничения на размер обрабатываемых так графов.

Для социологов важной задачей является мониторинг общественных изменений посредством анализа структуры явных и неявных групп пользователей в социальных сетях. Как пример решения таких задач в [37] рассматривается метод, названный его авторами «алгоритмом зерновой кластеризации». Данный метод соединяет в себе этапы экспертного определения целевых групп и их дополнения на основе данных из социальной сети. Выделение сообществ на графах взаимодействующих объектов онлайн социальных сетей дает возможность обнаруживать лидеров мнений и экспертов, осуществлять управление и контроль деятельности групп, представляющих угрозу для национальной безопасности [9].

В научной литературе рассматривается множество классических мер центральности для вершины графа [2, 3, 38, 39], характеризующих степень ее влияния на другие вершины (центральность по посредничеству, центральность по Кацу, Pagerank и т.д.). Возможности применения таких характеристик для анализа графов

социальных сетей представлены в [40] на примере алгоритмов ранжирования вершин. При этом классические меры центральности учитывают различные локальные или глобальные свойства вершины, но не учитывают структуру сообществ графа. Между тем, для целей сетевого анализа часто важно учитывать именно структуру сообществ, поэтому в последнее время внимание исследователей обращено на основанные на сообществах меры центральности (community-aware centrality measures) [41]. Это означает, что центральности вершин считаются после выделения сообществ и учитывают отдельно инцидентные этой вершине ребра внутри сообщества и между сообществами. В обзоре [41] приведены исследования 7 таких мер центральности, правда, стоит отметить, что все рассмотренные меры центральности и эксперименты приведены для невзвешенных графов.

Для графов, полученных при скачивании из сетей передачи данных важными характеристиками вершин и ребер являются их метаданные. Возможна реализация такой модели при построении взвешенного графа, при которой эти метаданные представляются как атрибуты вершин и ребер. Вес ребер и вершин при этом является показателем, связанным с этими атрибутами.

1.3 Методы выделения сообществ на графах

Как уже было сказано ранее, под выделением неявных сообществ на графе подразумевается разбиение графа на подграфы, такое что плотность связей внутри этих подграфов сильно выше плотности связей между ними. Понятие сообщества на графах встречается во множестве работ. При этом можно рассмотреть и задачу выделения на графе пересекающихся сообществ. В этом случае подразумевается наличие общих вершин, принадлежащих сразу двум или более сообществам. Алгоритмы поиска неявных сообществ в указанном смысле являются актуальной темой публикаций последних 20 лет [42-47].

Один из интуитивно понятных способов выделения сообществ на графе – поиск такого его разбиения, при котором максимизируется некоторый функционал,

характеризующий качество этого разбиения. Такой функционал, называемый модулярностью, может сравнивать разбиение графа с некоторой «нулевой гипотезой», заключающейся в том, что ребра распределены случайно, но сохранены некоторые свойства исходного графа. Значение этого функционала будет зависеть от выбранной «нулевой гипотезы», такая была предложена в работе [48], а модулярность, рассматриваемая в ней, носит название авторов (Ньюмана-Гирвана). Данная модель сохраняет исходные степени вершин графа, и при этом предполагает случайное распределение ребер между ними, то есть нет закономерностей в распределении плотности ребер внутри и между выделенными сообществами. Такая модулярность может быть использована для взвешенных графов и является одной из наиболее популярных характеристик качества разбиения графа на сообщества, за последние 20 лет ей было посвящено множество работ [49].

За эти годы были предложены разные подходы по поиску максимума для модулярности, включая и жадный алгоритм, и алгоритм симуляция отжига. Одним из быстрых алгоритмов, основанных на активном использовании модулярности Ньюмана-Гирвана, является агломеративный иерархический алгоритм Louvain [50]. Другой популярный подход к выделению сообществ заключается в имитации условного динамического процесса на графе. С точки зрения структуры сообществ малая плотность ребер между ними влечет за собой более вероятным протекание процесса внутри сообществ, чем переход между ними. Самый популярный алгоритм, реализующий данный подход [51-53], предусматривает сжатие информации о динамическом процессе, проходящем в графе, а именно – о случайном блуждании. Этот динамический алгоритм сводит задачу нахождения наилучшего структурного разбиения графа на сообщества к задаче оптимального сжатия информации о структуре графа. Для вычисления показателя качества заданного разбиения используется энтропия, описывающая среднюю длину кодового слова, взятого для кодирования вершины. Показатель качества полученного разбиения, выраженный через энтропию, может быть легко подсчитан для любого разбиения. Обновление и пересчет этого показателя является быстрой операцией. В дальнейшем были

предложены пути развития этого алгоритма, в том числе и для пересекающихся сообществ [54-55].

Встречаются и иные подходы по выделению сообществ. Например, теоретико-игровой метод в работе [56], где рассматривается выявление академических сообществ российских ученых в сети Интернет. А работа [45] посвящена алгоритму выделения пересекающихся сообществ с помощью рассмотрения полных подграфов заданного размера и его применению в биологических сетях.

В статье 2022 года исследователей, стоящих у истоков текущей теории сложных сетей, в том числе в вопросах выделения сообществ, были указаны некоторые актуальные по мнению авторов аспекты [49]. Как одна из задач указана проблема предела разрешения («resolution limit») [57] – ограничения снизу по размеру выделяемого в большой сети сообщества, что мешает находить маленькие сообщества на крупных графах.

Решение такой проблемы видится в реализации корректного итерационного перехода от анализа исходного графа к анализу отдельных его подграфов.

1.4 Тестирование алгоритмов выделения сообществ

Отсутствие в общем случае строгого определения для сообщества и однозначного решения задачи выделения как пересекающихся, так и непересекающихся сообществ, ставит проблему оценки корректности и эффективности работы алгоритмов и методов выделения сообществ на графах взаимодействующих объектов.

Стандартным тут может быть подход по генерации случайных графов для проведения на них тестирования алгоритмов и оценки полученного разбиения на сообщества. Первая, предложенная в свое время для анализа сетей модель, – модель Эрдёша-Реньи, плохо соответствует реальным сложным сетям, особенно социальным (онлайн) сетям [1, 2, 3, 35]. Модель Эрдёша-Реньи симулирует эффект «малого мира», но остальные свойства реальных сложных сетей в этой модели не реализуются, например, «эффект транзитивности», а биноминальное (пуассоновское для

больших графов) распределение степеней вершин не характерно для графов реальных сетей.

Для описания растущих сетей, которые развиваются со временем и которые нельзя описать моделью Эрдёша-Реньи, рассматриваются различные варианты модели Барабаши-Альберт. Модель Барабаши-Альберт при добавлении каждой новой вершины реализует метод предпочтительного присоединения между новой вершиной и старыми. Он заключается в построении ребер между новой вершиной и старыми с вероятностями, пропорциональными степеням старых вершин. В графах, сгенерированных по модели Барабаши-Альберт, больше вершин с малой степенью, чем в случайных графах; более вероятны вершины с высокой степенью. То есть распределение степеней вершин ближе к степенному закону. При этом модель симулирует и эффект «малого мира». Однако, данная модель не реализует «эффект транзитивности» – при увеличении размера генерируемого графа его кластерный коэффициент существенно падает [33]. Это противоречит свойству реальных сложных сетей.

Возникает запрос на генерацию графов, подходящих под условия сложных сетей. На первых этапах работы над алгоритмами нахождения сообществ на графах небольших размеров использовались составленные простые тесты. Наиболее известный тест был предложен Гирваном и Ньюманом (тест GN) [42] и содержал графы размером 128 вершин, которые разбиваются на сообщества одинаковых размеров. Но вершины у таких сгенерированных графов имели примерно одинаковые степени, а сообщества состояли из одинакового числа вершин. Таким образом, данная модель так же была далека от реальных сложных сетей.

Многие исследователи применяют LFR-модели [58] генерации случайных графов, обладающих структурой сообществ. Получаемые в рамках этой модели графы обладают распределением степеней вершин по степенному закону. При этом размеры встроенных при построении сообществ так же реализуются моделью по степенному закону, что нельзя однозначно отнести к положительным или отрицательным свойствам данной модели.

Логичным развитием разработки моделей генерации графов для тестирования алгоритмов является выбор меры качества для сравнения получаемых алгоритмами разбиений сгенерированного графа на сообщества и исходно встроенного в него разбиения на сообщества. Можно выделить три наиболее распространенных подхода для решения этой задачи [3]. Первый из них основан на сравнении полученных сообществ для пар вершин, то есть подсчете числа таких пар вершин, попавших в единое сообщество при обоих разбиениях и попавших в разные сообщества в обоих разбиениях и попавших в разные сообщества только при одном из разбиений. Наиболее известным из первой группы является индекс Жаккара. Для такого сравнения характерен недостаток, который заключается в том, что расхождения на локальном фрагменте графа могут дать существенное снижение общего результата для всего графа. Второй возможный подход для сравнения двух разбиений состоит в поиске наибольших пересечений пар сообществ из разных разбиений. Но у этого подхода имеются свои недостатки, связанные с небольшими пересечениями сообществ из разных разбиений.

Третий вариант решения данной задачи основан на методах из теории информации и заключается в предположении о том, что для близких разбиений достаточно сообщить небольшое количество информации, чтобы получить из одного другое. Одним из наиболее популярных тут является подсчет NMI – нормированной взаимной информации (Normalized Mutual Information) [59]. Во многих работах последних лет сравнение результатов алгоритмов произведено с помощью NMI, между тем, выявлены и недостатки у этой меры качества, связанные со случайными разбиениями на большое число маленьких сообществ. Поэтому ряд авторов предлагают свои корректировки для NMI [60, 61].

Как было ранее отмечено, реальные сложные сети обладают рядом свойств, которые не во всех моделях генерируемых графов реализуются. Более того, как показывают разные исследования, сложные сети тоже нельзя считать однородными, ибо имеются характерные для того или иного типа сетей (социальные, биологические, технологические) топологические свойства [1, 2, 35]. Кроме этого, стоит отметить, что во многих работах детально рассматриваются случаи невзвешенных

графов, тогда как для социальных сетей важную роль играют атрибуты ребер и вершин [62]. Например, интенсивность взаимодействия между пользователями удобно описывается взвешенными графами. Поэтому важным видится исследование моделей импорта данных из реальных сетей и построение на их основе взвешенных графов взаимодействующих объектов [63-67].

В работе [49] так же поднят как актуальный вопрос об оценке качества выделения сообществ на сгенерированных графах. Авторами отмечено, что сравнение с заранее заданным разбиением на искусственных сетях дает сбой для разреженных графов, что как раз характерно для коммуникационных сетей. В этой связи для оценки качества выделения сообществ на графах взаимодействующих объектов, получаемых при импорте данных, целесообразным видится исследование атрибутивных данных вершин и ребер выделенных сообществ. Такая методика показана в работе [68], где методами компьютерной лингвистики проводится анализ текстов и их характеристик у получившихся сообществ для предлагаемого в статье метода их выделения.

Таким образом, актуальным направлением исследования сетей коммуникационного взаимодействия является импорт данных из реальных сетей с последующим построением взвешенных графов взаимодействующих объектов, выделением на них неявных сообществ, в том числе с использованием методик, основанных на переходе от анализа всего исходного графа к его подграфам. Именно такой подход, сочетающий в себе текстовый анализ атрибутов элементов графа, представлен в разработанных автором методах: «Комбинированный алгоритм», «Метод ядра», «Метод Галактик» [68, 69, 70, 71, 72, 73].

1.5 Выводы по главе 1

1. Приведенный в данной главе обзор исследований показывает, что за последние годы многие ученые по всему миру развили теорию сложных сетей: были исследованы свойства данных сетей, построены различные их модели, разработаны алгоритмы выделения неявных сообществ на графах таких сетей. Вместе с тем, для

анализа реальных сетей коммуникационного взаимодействия во многих случаях ключевыми проблемами остаются построение отражающих их свойства моделей, необходимость рассмотрения отдельных фрагментов соответствующих графов в силу найденных ограничений, выделение ключевых вершин, методики оценки эффективности выделения сообществ на графе, эффективное хранение и визуализация таких графов.

2. В силу указанного актуальными являются следующие решенные далее в диссертации задачи: разработка моделей построения взвешенного графа для сетей коммуникационного взаимодействия и реализация этих моделей; построение методов и алгоритмов для выделения пересекающихся и непересекающихся сообществ на графе, позволяющих анализировать отдельные компоненты исходной сети; разработка методики для оценки качества выявления сообществ, основанной на дополнительных данных; разработка модели для эффективного хранения импортированных из существующих коммуникационных сетей графов.

ГЛАВА 2 ПОСТРОЕНИЕ ГРАФА ВЗАИМОДЕЙСТВУЮЩИХ ОБЪЕКТОВ

В данной главе представлена разработанная математическая модель построения графов взаимодействующих объектов, варианты ее адаптации для разных коммуникационных сетей [63, 64, 66, 67, 71, 72, 73, 74, 75, 76]. Личный вклад автора составляют сама модель, ее вариации и методика их применения для формирования взвешенного графа информационного взаимодействия для разных сетей.

2.1 Построение графов для сетей коммуникационного взаимодействия

Для сети коммуникационного взаимодействия рассмотрим модель построения соответствующего графа. Вершины графа будут соответствовать узлам этой сети. Ребра, инцидентные вершинам, строятся на основе зафиксированных в сети факторов взаимодействия между узлами, соответствующими вершинам. Для учета каждого из таких факторов модель подразумевает в зависимости от конкретной сети использование функций, характеризующих степень (интенсивность или близость) каждого из соответствующих взаимодействий. Под факторами взаимодействия могут пониматься как активные действия объектов сети за фиксированный заранее промежуток времени, так и статические данные, присущие всем объектам данной сети, доступные в процессе импорта данных из нее. Для таких сетей важно учитывать, что вершины, как правило, обладают некоторой информацией, которую будем представляется в виде их атрибутов, в том числе и текстовых.

Пусть в процессе импорта данных получена информация о множестве вершин V. Тогда для этого множества можно построить гипотетическое множество всех возможных ребер, инцидентных парам разных вершин из V, обозначим это множество E, а его элементы как $e \in E$. Получен полный (простой, неориентированный) граф $G(V, E)$, в котором изначально ребра не имеют весов. Далее, пусть в коммуникационной сети выявлено N факторов взаимодействия ее объектов. Тогда

для построения графа взаимодействующих объектов введем на множестве E следующую весовую функцию:

$$w(e) = \sum_k^N W_k \cdot \delta_e^k, \quad (2.1)$$

где W_k – весовой коэффициент для k-го фактора взаимодействия;

δ_e^k – функция, принимающая нулевое значение в случае отсутствия k-го фактора взаимодействия у ребра e и натуральные значения в иных случаях в зависимости от степени интенсивности взаимодействия, заложенной в конкретной вариации модели.

В случае, если для какого-то ребра e так определенная весовая функция принимает нулевое значение $w(e) = 0$, данное ребро удаляется из графа. В итоге получаем новое множество ребер $\tilde{E}$. Важно отметить, что исходный импорт множества вершин V должен быть устроен таким образом, чтобы каждый его элемент $v \in V$ имел хотя бы одно ненулевое ребро с остальными вершинами из V. Что гарантирует сохранение одной компоненты связности в графе после удаления ребер с нулевыми значениями весовой функции. По итогам такой предобработки получаем граф взаимодействующих объектов $G(V, \tilde{E})$.

Формирование графов взаимодействующих объектов обеспечивается разработанными процедурами импорта данных из соответствующих коммуникационных сетей с последующим построением взвешенного графа на основе выбранных факторов взаимодействия, весовой функции и набора атрибутов вершин. В построенном связном графе вершины соответствуют объектам, а ребра и вес на них описывают интенсивность или характер их взаимодействия. Таким образом, между каждыми двумя вершинами графа определяется обратное взвешенное расстояние.

Для построения графа взаимодействующих объектов необходимо обеспечить импорт данных из соответствующей сети с последующим построением взвешенного графа на основе выбранных факторов взаимодействия и весовой функции. В рамках методологии, соответствующей подходам по хранению графов, описанному в главе 7, будем использовать следующий алгоритм. Изначально определяется стартовый объект v_0, который войдет на первом шаге в множество V. Это мо-

жет быть и не один объект, а множество таких объектов $V_0 \subset V$, но тогда необходимо будет производить последующие шаги импорта до тех пор, пока формируемый граф не станет связным. Так же возможен вариант, при котором изначально определяется какой-то атрибут в сети, по которому находятся объекты для формирования V_0. Например, если рассматривается социальная сеть *ВКонтакте* то таким стартовым атрибутом может быть некоторая публично доступная запись (пост), с которой имели взаимодействия узлы сети (пользователи), поэтому у них имеется данный атрибут. По этому атрибуту находятся соответствующие вершины для множества V_0.

Далее, начиная с V_0 поиском в ширину по сети на заданную изначально глубину d строится $V_0 \subset V_1 \subset V_2 \subset \cdots \subset V_d = V$. На шаге i для построения V_i находятся объекты, имевшие какой-то из факторов взаимодействия с уже найденными на шаге $i-1$ элементами множества V_{i-1}. То есть на каждом шаге i находятся объекты, которыми дополняется для получения итогового $V_0 \subset V_1 \subset V_2 \subset \cdots \subset V_d = V$ множества вершин.

Для конкретных коммуникационных сетей заранее фиксируются факторы взаимодействия, по которым происходит такой поиск в ширину. Причем, таких факторов для осуществления поиска по сети может быть меньше, чем заданное N, по которому впоследствии будет происходить подсчет значений для весовой функции. После окончания процедуры формирования множества V строится множество E, вычисляются значения $w(e)$ для всех элементов $e \in E$, производится предобработка и по итогам получаем граф взаимодействующих объектов $G(V, \tilde{E})$. Метаданные импортируются в процессе формирования множества вершин и сохраняются как атрибуты в зависимости от необходимого набора типов таких атрибутов.

С точки зрения прикладной реализации процесс выглядит следующим образом. Основой для взаимодействия с коммуникационной сетью являются запросы, как правило, это *http*-запросы, которые возвращают данные в формате *JSON* [77]. В результате импорта данных из сети посредством программного обеспечения формируется специальный *XML*-подобный файл графа в унифицированном формате разметки — *AVS* (*Analytics and Visualization System for graphs*).

Файл данного формата содержит список вершин и ребер, а также сведения о них, которые были получены из коммуникационной сети. Сведения о каждом из этих объектов представлены в файле как их атрибуты. Ключевая характеристика такого формата заключается в представлении данных об объектах, которые являются большими объемами текстовой информации. Вместо хранения этих объемных данных непосредственно в *AVS*-файле, соответствующем графу, в разметке значение соответствующего атрибута указывает на внешний файл, где и находится сам текст, также может быть указана и конкретная часть подобного файла с текстом. Это позволяет отдельно обрабатывать тексты и при необходимости снижать объем данных путем сжатия. Подробнее строение формата описано в Приложении 1.

Данный подход реализуется в том числе посредством распределения данных при их импорте из реальных сетей. В файле формата *AVS* сохраняются данные о вершинах и ребрах графа, также там указываются значения свойств малого размера и ссылки на объемную информацию об объектах. Для хранения текстов может быть использован *YAML*-формат [78, 79].

В последующих разделах текущей главы дано описание вариантов модели построения графов взаимодействующих объектов для ряда коммуникационных сетей: для социальной сети *ВКонтакте* [63], сети коротких сообщений *Twitter* [64] и сети *Telegram*-каналов одноименной сети мгновенного обмена сообщениями [66]. Каждая из этих сетей использует свой синтаксис запросов и формат возвращения данных.

2.2 ВКонтакте

Для классической социальной сети *ВКонтакте* в качестве факторов взаимодействия используется совокупность из статических данных, имеющихся у объектов (пользователей) сети, и импортированных данных об имевших место действиях объектов (пользователей) в отношении друг друга за промежуток времени T.

Для классических социальных сетей целесообразно рассматривать графы «друзей», где ребра строятся на основании имеющегося между пользователями отношения взаимной подписки или «дружбы». В рамках рассматриваемой модели возможно получить и такие графы, определив лишь один статический фактор взаимодействия как отношение «дружбы» между объектами, такие графы недостаточно отвечают задаче анализа информационного воздействия на объекты сети.

Обход сети для импорта данных, как и было сказано ранее, может начинаться либо с вершины (или множества вершин V_0), либо с записи в сети, поста (или некоторого множества постов). В случае сети *ВКонтакте* V_0 строится начиная с пользователей, которыми представлены вершины. Либо пользователи (вершины) импортируются и составляют V_0 как имевшие взаимодействие с исходно заданными постами. Каждый следующий шаг для построения V_i производится алгоритмом обхода в ширину и предполагает добавление вершин за счет выявленных факторов взаимодействия по выбранному заранее набору этих факторов, например отношения «дружбы» пользователей или записей в публичных постах друг друга. В конце строятся ребра между всеми парами полученных вершин и вычисляются их веса на основании формулы (2.1).

Социальная сеть *ВКонтакте* предоставляет прикладной программный интерфейс (API) [80], который позволяет получать информацию из базы данных vk.com. Для осуществления *http*-запросов к данным сети *ВКонтакте* как правило используется сторонняя открытая библиотека *Libcurl* [81]. С ее помощью создаются объекты и сессии, результаты обращения к которым потом данные парсятся и записываются в формат *JSON*. Для обработки *JSON* можно использовать, например, библиотеку *hlohmann json* [77]. Далее выполняются запросы с соответствующими параметрами, результаты которых последовательно записываются в объектах, созданных в структуре данных модуля импорта. После импорта данных на следующем этапе для подсчета веса ребер и обратного расстояния между вершинами используются весовые коэффициенты выбранных факторов взаимодействия.

Как базовый для сети *ВКонтакте* рассмотрим набор из $N = 11$ факторов. Первые 6 из них являются статическими:

1. $\delta_e^1 = \delta_e^{\text{страна}}$ принимает ненулевое значение, равное единице, в случае если у двух инцидентных ребру e вершин указана одинаковая страна.

2. $\delta_e^2 = \delta_e^{\text{город}}$ принимает ненулевое значение, равное единице, в случае если у двух инцидентных ребру e вершин указан одинаковый текущий город.

3. $\delta_e^3 = \delta_e^{\text{школа}}$ принимает ненулевое значение, равное единице, в случае если у двух инцидентных ребру e вершин указана одна и та же школа. Тут возможно использование дополнительных словарей, классифицирующих школы.

4. $\delta_e^4 = \delta_e^{\text{университет}}$ принимает ненулевое значение, равное единице, в случае если у двух инцидентных ребру e вершин указан один и тот же университет. Актуальность любых баз вузов зависит от исходной задачи. Встроенная в *ВКонтакте* база может подходить для базовых задач. Но для каких-то более специализированных на студенческом контингенте задач при построении таких графов может быть актуально создание дополнительного словаря, связывающего разные университеты и их филиалы.

5. $\delta_e^5 = \delta_e^{\text{факультет}}$ принимает ненулевое значение, равное единице, в случае если у двух инцидентных ребру e вершин указан одинаковый факультет. Возможно дополнительное условие, что $\delta_e^{\text{университет}} = 1$. Но в ряде случаев при построении специализированных графов имеет смысл и не учитывать совпадение университета.

6. $\delta_e^6 = \delta_e^{\text{общ.друзья}}$ принимает значение, равное количеству общих друзей у двух инцидентных ребру e вершин.

Следующие 5 факторов относятся к активным действиям пользователей:

7. $\delta_e^7 = \delta_e^{\text{запись}}$ принимает значение, равное количеству записей, оставленных на страницах друг друга пользователями, соответствующими двум инцидентным ребру e вершинам. При этом сами тексты записей импортируются как атрибуты вершин с использованием *YAML*-формата.

8. $\delta_e^8 = \delta_e^{\text{лайк.запись}}$ принимает значение, равное количеству отметок «Нравится» (далее – «лайк»), оставленных под записями друг друга пользователями,

соответствующими двум инцидентным ребру e вершинам. Возможна вариация модели, где учитывается только факт хотя бы одного лайка. Тогда $\delta_e^{\text{лайк.запись}}$ принимает только два значения – ноль и единицу.

9. $\delta_e^9 = \delta_e^{\text{комм.запись}}$ принимает значение, равное количеству комментариев, оставленных на постах друг друга пользователями, соответствующими двум инцидентным ребру e вершинам. При этом сами тексты комментариев импортируются как атрибуты вершин с использованием *YAML*-формата.
10. $\delta_e^{10} = \delta_e^{\text{лайк.фото}}$ определяется аналогично $\delta_e^{\text{лайк.запись}}$.
11. $\delta_e^{11} = \delta_e^{\text{комм.фото}}$ определяется аналогично $\delta_e^{\text{комм.запись}}$.

Таким образом, на основании формулы (2.1) для социальной сети *ВКонтакте* и выбранных факторов взаимодействия получаем следующее выражение для весовой функции в рамках представленной вариации основной модели:

$$w(e) = \sum_{k=1}^{11} W_{\text{k}} \cdot \delta_e^k \qquad (2.2)$$

При заданных значениях весов W_{k} для каждого из выбранных факторов взаимодействия в рамках рассматриваемой для сети *ВКонтакте* вариации модели получаются взвешенные неориентированные графы. Выделим следующие их виды в зависимости от заданных весов и от того, какие из указанных факторов взаимодействия учитываются при обходе в ширину и построении множества V.

Базовый подход состоит в построении графа для оценки близости профилей пользователей, их взаимного сходства. Тут идея в приоритетном учете именно схожести статических характеристик вершин без учета при построении V взаимодействия пользователей с материалами друг друга.

Граф симпатии пользователей – множество V строится без учета взаимодействия пользователей с записями, но с учетом взаимодействия с фотографиями.

Граф информационного взаимодействия пользователей – множество V строится с учетом взаимодействия пользователей с записями друг друга, но без учета взаимодействия с фотографиями.

Граф чистого информационного взаимодействия пользователей – множество V строится с учетом взаимодействия пользователей с записями и фотографиями друг друга, но при этом $W_{\mathrm{k}} = 0$ для $k = 1, 2, 3, 4, 5$. Нацелен на выделение активно взаимодействующих с постами в сети пользователей и информационным воздействием, поэтому тут целесообразна минимизация связей общего характера.

Рассмотрим пример построения графов этих четырех видов, полученных при импорте данных, начиная с одного и того же исходного множества вершин V_0. Как стартовый в мае 2020 года был взят один популярный среди студентов пост в сети, на основе которого построены графы G_1, G_2, G_3 и G_4, соответствующие четырем описанным выше вариациям модели. Данные по полученным графам приведены в таблице 2.1.

Таблица 2.1 – Графы разных вариантов модели, импортированные из *ВКонтакте*

	Число вершин графа	Число ребер графа
G_1	272	1035
G_2	8299	244785
G_3	721	4042
G_4	680	428

Значения весов W_{k} для графов G_1, G_2, G_3 были выбраны следующим образом: $W_1 = W_2 = W_4 = 1$, и $W_3 = W_5 = 4$. Далее с использованием программного обеспечения, описанного в главе 7, получена визуализация для этих графов и выделенных на них сообществ в рамках методики, предложенной в главе 5.

Внутри сообществ, выделенных на графе общего сходства пользователей G_1, часто видна ключевая вершина, связывающая большинство пользователей сообщества. Как правило, это наиболее активный участник сети, сочетающий в себе социальные характеристики и знакомый с другими членами этого сообщества. При этом, исходя из построения графа, в данном случае нельзя сделать утверждение, что эта ключевая вершина распространяет информацию и оказывает информационное воздействие на остальных в своем сообществе (рисунок 2.1).

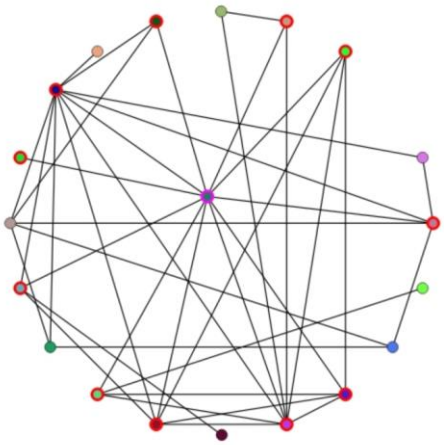

Рисунок 2.1 – Внутреннее устройство одного из выделенных на графе G_1 сообществ. Ключевая вершина перемещена в центр для наглядности. Смежные с ней вершины выделены красным цветом.

Посмотрим теперь на пример того, как выглядят изнутри некоторые сообщества на графе G_3 информационного взаимодействия пользователей (рисунок 2.2) и графа G_4 чистого информационного взаимодействия пользователей (рисунок 2.3). Для графа G_4 явно видно, что рассмотренная ключевая вершина может являться источником информационного воздействия.

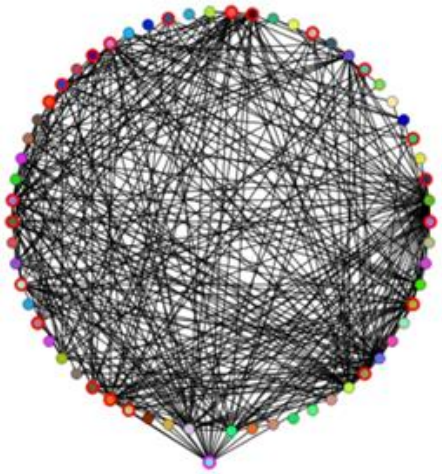

Рисунок 2.2 – Внутреннее устройство одного из выделенных сообществ на графе G_3, ключевая вершина сдвинута вниз для наглядности. Смежные с ней вершины выделены красным.

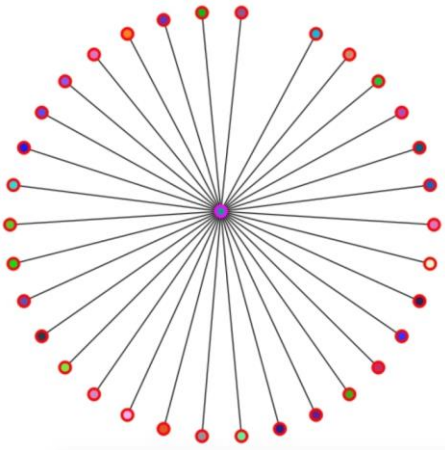

Рисунок 2.3 – Внутреннее устройство одного из выделенных сообществ на графе G_4, ключевая вершина сдвинута в центр наглядности. Смежные с ней вершины выделены красным.

Помимо указанных выше видов графов в рамках модели возможно построение и графа другого вида. Смешанный граф пользователей – множество V строится с учетом всех выбранных типов взаимодействия.

Таким образом, в данном разделе представлены пять видов графов, относящихся к вариации модели для построения взвешенных графов взаимодействующих объектов, которые можно использовать при импорте данных из социальной сети *ВКонтакте*. Данные модели были апробированы на реальных данных этой социальной сети.

2.3 Twitter

Рассмотрим вариацию модели для сети коротких сообщений *Twitter*. Ныне эта сеть поменяла свое название на *X* и продолжает менять функционал, но в работе рассматривается сеть до этих глобальных изменений. Вначале опишем ее устройство, чтобы было понятно, как строится эта вариация модели.

Пользователи этой сети публикуют в своей личной ленте короткие сообщения, на которые могут реагировать другие пользователи. Эти короткие сообщения называются «постами». Пользователи могут не только положительно оценить отметкой

«нравится» чужой пост, но и процитировать в своей ленте чужое сообщение – сделать «ретвит». Поэтому в данной сети для построения согласно модели (2.1) графа выберем следующие факторы взаимодействия: подписки друг на друга, отметки «лайк» («нравится») под постами других пользователей, комментарий поста другого пользователя, «ретвит» поста другого пользователя.

С учетом выбранных факторов, согласно модели, формируется граф взаимодействующих объектов, множество вершин которого V соответствует аккаунтам пользователей (ими могут быть как частные лица, так и организации, компании и т.п.). Ребра, как и в общем случае, строятся на основе выбранных факторов взаимодействия.

Определим $(F,\ L,\ C,\ R)$-модель информационного взаимодействия в сети *Twitter* как взвешенный граф $G(V,E)$, у которого на множестве ребер E весовая функция $w(e)$ с неотрицательными значениями задана следующим образом:

$$w(e) = F \cdot \delta_e^F + L \cdot \delta_e^L + C \cdot \delta_e^C + R \cdot \delta_e^R, \qquad (2.3)$$

где:

$$\delta_e^F = \begin{cases} 2, \text{ если оба инцидентных пользователя подписаны друг на друга} \\ 1, \text{ если хотя бы один из инцидентных пользователей подписан на другого} \\ 0, \text{иначе} \end{cases}$$

$$\delta_e^L = \begin{cases} 2, \text{ если между инцидентными пользователями есть взаимые Like} \\ 1, \text{если есть хотя бы один Like между инцидентными пользователями} \\ 0, \text{иначе} \end{cases}$$

$$\delta_e^C = \begin{cases} 2, \text{ если между инцидентными пользователями есть взаимые Reply} \\ 1, \text{если есть хотя бы один Reply между инцидентными пользователями} \\ 0, \text{иначе} \end{cases}$$

$$\delta_e^R = \begin{cases} 2, \text{ если между инцидентными пользователями есть взаимые Retweet} \\ 1, \text{если есть хотя бы один Retweet между инцидентными пользователями} \\ 0, \text{иначе} \end{cases}$$

а $F,\ L,\ C,\ R$ – веса соответствующих факторов взаимодействия, фиксированные для всего графа. То есть для формулы (2.1) взято $N = 4$ и введены обозначения $W_1 = F$, $W_2 = L$, $W_3 = C$ и $W_4 = R$, которые лучше характеризуют для данной вариации общей модели суть соответствующих факторов.

Под взаимностью в указанных выше факторах взаимодействия понимаются действия от каждого из двух пользователей, но не обязательно эти действия относятся к одному и тому же посту. Это означает, что один пользователь, соответствующий вершине v_1 может поставить «лайк» другому, соответствующему вершине v_2 за первый пост, а пользователь, соответствующий v_2 в свою очередь может поставить *Like* пользователю, соответствующему вершине v_1 за другой пост. И в таком случае для ребра $e = \{v_1, v_2\}$ определяем $\delta_e^L = 2$. Аналогичное правило действует для δ_e^C и δ_e^R.

Граф взаимодействующих объектов для сети *Twitter* строится по результатам импорта данных об имевшем место взаимодействии пользователей с заданными изначально постами и между собой, составляющими множество вершин V_0. Для дальнейшего построения множества V на шаге i берутся посты пользователей, соответствующих вершинам, уже вошедшим в V_{i-1} на предыдущем шаге. После формирования множества V за счет импорта данных по факторам взаимодействия для всех ребер из E вычисляется их вес согласно формуле (2.3). Итоговые результаты импорта для графа $G(V, \tilde{E})$ сохраняются в *AVS*-файле. Одним из атрибутов вершины является объединение всех текстов из постов, оставленных пользователем, соответствующим этой вершине. Далее эти тексты могут быть проанализированы в соответствии с методиками, описанными в главе 6.

Импорт данных из сети *Twitter* реализуется как с помощью *API* [82], так и с помощью извлечения данных со страниц («скрапинг») [83] за счет такого инструмента для автоматизации взаимодействия с веб-браузерами на уровне пользователя как *Selenium WebDriver* [84].

Помимо описанного выше построения (F, L, C, R)-модели рассмотрим еще одну модель, которая может быть использована для построения серии подграфов, характеризующих хронологическое развитие подсети. В данном случае используем для построения ребер не все факторы взаимодействия, которые дают множество вершин. Это может приводить для данной модели к графу, содержащему более одной компоненты связности. Внутри каждой из них поступаем как ранее для всего графа.

Для заданного временного интервала t и изначального множества постов импортируем следующие списки пользователей: список лайкнувших хотя бы один пост из заданного множества; список прокомментировавших хотя бы один пост из заданного множества. Тексты комментариев за временной интервал t также импортируются. Объединение списков дает итоговое множество вершин V будущего графа. Для перехода от графа $G(V, E)$ к взвешенному графу взаимодействующих объектов $G(V, \tilde{E})$ вычисляется вес $w(e_{ij})$ для каждого ребра $e_{ij} \in E$, инцидентного вершинам v_i и v_j следующим образом:

$$w(e_{ij}) = 1 \cdot \delta_e^F + 2 \cdot likes_{ij} + 2 \cdot likes_{ji}, \quad (2.4)$$

где δ_e^F определяется формуле (2.3);

$likes_{ij} = 1$, если за временной интервал t на постах пользователя, соответствующего вершине v_i, есть хотя бы один лайк от пользователя, соответствующего вершине v_j, иначе $likes_{ij} = 0$;

$likes_{ji} = 1$, если за временной интервал t на постах пользователя, соответствующего вершине v_j, есть хотя бы один лайк от пользователя, соответствующего вершине v_i, иначе $likes_{ji} = 0$.

Таким образом для формулы (2.1) в данном случае взято $N = 3$, введены обозначения $\delta_e^2 = likes_{ij}$ и $\delta_e^3 = likes_{ji}$ для большей наглядности. А также взяты значения для весовых коэффициентов $W_1 = 1$ и $W_2 = W_3 = 2$.

Необходимо отметить, что при взаимных подписках и отметке *Like* лишь с одной из сторон вес ребра будет равен 4.

Импортированные тексты комментариев становятся атрибутами вершин графа $G(V, \tilde{E})$. Сформированный так граф взаимодействующих объектов записывается в *AVS*-файл.

Особенностью этой модели является то, что она учитывает лишь наличие взаимодействия между парой вершин и в меньшей степени его интенсивность. Действительно, максимальный вес ребер, равный 6 достигается для пар пользователей со взаимными подписками и как с большим числом взаимных оценок *Like* на постах, так и в случае хотя бы одной пары таких оценок с двух сторон за временной

интервал t. Это позволяет намеренно упростить картину как для выявления объектов сети, так и для анализа ее роста в процессе времени. Примеры таких исследований сети с использованием этой модели и полученных по ней графов, соответствующих разным временных интервалам, но с едиными исходными вершинами, приведены в разделе 6.4.

Таким образом, в данном разделе представлены две вариации модели для построения взвешенных графов взаимодействующих объектов. Эти версии предназначены для формирования графов на основе данных, полученных при обработке информации из *Twitter*. Первая – определяемая задаваемыми параметрами $(F,\ L,\ C,\ R)$-модель, вторая – модель построения хронологических подграфов. Обе модели были апробированы на реальных данных (главы 4 и 6).

2.4 Telegram-каналы

В *Telegram* как сети обмена мгновенными сообщениями существуют публично доступные *Telegram*-каналы. Далее будем называть их просто каналами. Существует возможность ведения открытых каналов, которые доступны для чтения всем пользователям мессенджера и могут быть найдены встроенным в *Telegram* поиском. Так же существует возможность сделать канал закрытым, доступ к таким каналам предоставляется пользователям по запросу от администраторов. Такие каналы обладают в сети уникальными идентификаторами (ID) и текстовым именем, а также гиперссылкой для быстрого перехода и коротким текстовым описанием, доступным для пользователей мессенджера. Канал фактически представлять из себя последовательность текстовых записей (постов) с возможностью добавления к ним файлов, включая графические и аудио/видео. Каждый такой пост может содержать предусмотренные инструментом *Telegram* отсылки к другим каналам («упоминания»), *URL*-ссылки на ресурсы информационно-телекоммуникационной сети Интернет. Дополнительно для редактора канала предусмотрен функционал формального публичного «репоста» записи другого канала – добавления записи, состоящей из цитаты записи другого канала с явным указанием его как источника.

Все это позволяет построить сеть таких каналов и факторы их взаимодействия между собой.

Для построения графа взаимодействующих объектов сети *Telegram*-каналов будем в качестве объектов сети, элементов множества V, рассматривать сами каналы. Связи между объектами представляют собой взаимодействие между ними с точки зрения упоминаний одним каналом других в своих постах («mentioned»), «репостов» записей из других каналов, также будем учитывать совпадающие *URL*-ссылки на сторонние сайты. Эти три фактора взаимодействия и будут использованы для данной коммуникационной сети. Множество всех ребер E будет построено на их основе, взаимодействия учитываются при этом на заданном временном промежутке T. С целью получения данных используется *Telegram API* [85].

Определим модель информационного воздействия каналов. Для этого в рамках формулы (2.1) и указанных выше факторов взаимодействия возьмем такое выражение для весовой функции:

$$w(e) = \sum_{k=1}^{3} W_{\text{k}} \cdot \delta_e^k, \tag{2.5}$$

В формуле (2.5) обозначим W_1 как U, ибо этот фактор будет отвечать за внешние ссылки. Для W_2 дадим обозначение M – явное упоминание других каналов; и вес «репостов» W_3 обозначим за R. Далее, для произвольного ребра $e_{ij} \in E$, соединяющего вершины i и j, введем обозначение $\delta_{e_{ij}}^k = \delta_e^k$. Тогда для e_{ij} можем записать факторы из формулы (2.5) в соответственно: $\delta_e^1 = \delta_{e_{ij}}^U$, $\delta_e^1 = \delta_{e_{ij}}^M$ и $\delta_e^1 = \delta_{e_{ij}}^R$. Тогда подставив эти значения в (2.5) получаем:

$$w\left(e_{ij}\right) = U \cdot \delta_{e_{ij}}^U + M \cdot \delta_{e_{ij}}^M + R \cdot \delta_{e_{ij}}^R , \tag{2.6}$$

Остается определить, как вести подсчет зафиксированных за выбранный период T взаимодействий по данным факторам. Сделаем это следующим образом:

$$\delta_{e_{ij}}^U = \left|URL_i \cap URL_j\right|, \tag{2.7.1}$$

где URL_s – множество уникальных внешних ссылок в записях канала, соответствующего вершине s;

$$\delta_{e_{ij}}^M = Mpost_i^j + Mpost_j^i, \tag{2.7.2}$$

где $Mpost_h^g$ – количество уникальных записей в канале, соответствующем вершине h, где упоминается канал, соответствующий вершине g;

$$\delta_{e_{ij}}^R = REpost_i^j + REpost_j^i, \quad (2.7.3)$$

где $REpost_h^g$ – количество уникальных записей в канале, соответствующем вершине h, где процитирован (сделан «репост» записи) канал, соответствующий вершине g.

Эту вариацию модели (2.1) для сети каналов назовем (U, M, R)-моделью.

На основе графа $G(V, E)$ после вычисления весов ребер для всех пар вершин в соответствии с формулой (2.6) строится взвешенный граф взаимодействующих объектов $G(V, \tilde{E})$, где $\tilde{E}$ – множество ненулевых ребер.

Для построения множества V изначально формируется список исходных каналов – множество V_0, глубина для поиска в ширину d и временной период T. Итеративно запускается d раз процесс импорта данных для получения множества V_i. Для этого для каждого канала из V_{i-1} формируются списки каналов по факторам взаимодействия $\delta_{e_{ij}}^M$ и $\delta_{e_{ij}}^R$. Фактически списки формируются только для добавленных на предыдущем шаге каналов, ибо по добавленным ранее уже поиск произведен. После формирования этих списков каналы из них объединяются с исключением повторений. Полученное объединение добавляется к множеству V_{i-1}, опять же исключая возможные повторения. Процесс повторяется до получения $V_d = V$.

В процессе выполнения скачивания и построения множества V происходит полная обработка постов всех анализируемых каналов, часть полученной информации будет использована при определении весов на ребрах графа, а тексты постов каналов сохраняются как их атрибуты в формате *YAML* [78].

Взвешенный граф взаимодействующих объектов $G(V, \tilde{E})$ получается после вычисления весов ребер для всех пар вершин, где $\tilde{E}$ – множество ненулевых ребер. Подсчет весов ребер $w(e_{ij})$ на полученном множестве вершин для построения разных графов может быть произведен неоднократно с различными значениями для весов U, M, R у факторов взаимодействия.

Построенный граф $G(V, \tilde{E})$ сохраняется в формате *AVS*, описанном в Приложении 2. При этом сами тексты хранятся отдельно, что позволяет использовать их для оценки качества выделенных на графе сообществ. Данная методика подробнее описана в главах 5 и 6.

Таким образом, в данном разделе представлена параметрическая (U, M, R)-модель для построения взвешенных графов взаимодействующих объектов сети *Telegram*-каналов. Модель была апробирована на реальных данных, примеры чего приведены в главе 5.

2.5 Экспериментальные исследования модели

В рамках представленной модели (2.1) и ее вариации для сети *Telegram*-каналов (2.5) используется функция, зависящая линейно от факторов взаимодействия в сети δ_e^k. Рассмотрим альтернативные варианты подсчета весовой функции на стадии перехода от $G(V, E)$ к $G(V, \tilde{E})$ и формирования взвешенного графа взаимодействующих объектов. Для этого обозначим весовую функцию на ребрах графа $G(V, E)$ как F:

$$w(e_{ij}) = F(\delta_{e_{ij}}^U, \delta_{e_{ij}}^M, \delta_{e_{ij}}^R) \tag{2.8}$$

Далее посмотрим на следующие варианты такой функции. Во-первых, это две линейных функции $F_{1,1}$ и $F_{1,2}$, соответствующие представленной ранее (U, M, R)-модели (2.6) для значений $U = 1$, $M = 2$, $R = 3$ и $U = M = R = 1$ соответственно:

$$F_{1,1} = 1 \cdot \delta_{e_{ij}}^U + 2 \cdot \delta_{e_{ij}}^M + 3 \cdot \delta_{e_{ij}}^R \tag{2.9.1}$$

$$F_{1,2} = 1 \cdot \delta_{e_{ij}}^U + 1 \cdot \delta_{e_{ij}}^M + 1 \cdot \delta_{e_{ij}}^R \tag{2.9.2}$$

Кроме этого, рассмотрим и принципиально иные варианты для функции F, предусматривающие логарифмические и степенные зависимости.

$$F_{2,1} = 1 \cdot \ln(\delta_{e_{ij}}^U + 1) + 1 \cdot \ln(\delta_{e_{ij}}^M + 1) + 1 \cdot \ln(\delta_{e_{ij}}^R + 1) \tag{2.9.3}$$

$$F_{2,2} = 1 \cdot \ln(\delta_{e_{ij}}^U + 1) + 2 \cdot \ln(\delta_{e_{ij}}^M + 1) + 3 \cdot \ln(\delta_{e_{ij}}^R + 1) \tag{2.9.4}$$

$$F_{3,1} = 2^{1 \cdot \frac{\delta^U_{e_{ij}}}{max(\delta^U_e)}} + 2^{1 \cdot \frac{\delta^M_{e_{ij}}}{max(\delta^M_e)}} + 2^{1 \cdot \frac{\delta^R_{e_{ij}}}{max(\delta^R_e)}} - 3 \quad (2.9.5)$$

$$F_{3,2} = 2^{1 \cdot \frac{\delta^U_{e_{ij}}}{max(\delta^U_e)}} + 2^{2 \cdot \frac{\delta^M_{e_{ij}}}{max(\delta^M_e)}} + 2^{3 \cdot \frac{\delta^R_{e_{ij}}}{max(\delta^R_e)}} - 3 \quad (2.9.6)$$

$$F_{3,3} = 1 \cdot 2^{1 \cdot \frac{\delta^U_{e_{ij}}}{max(\delta^U_e)}} + 2 \cdot 2^{1 \cdot \frac{\delta^M_{e_{ij}}}{max(\delta^M_e)}} + 3 \cdot 2^{1 \cdot \frac{\delta^R_{e_{ij}}}{max(\delta^R_e)}} - 6 \quad (2.9.7)$$

Здесь под $max(\delta^k_e)$ понимается максимальное по всему графу $G(V,E)$ значение δ^k_e для k-го фактора взаимодействия. Так же рассмотрим варианты функции F, составленные как комбинации:

$$F_{4,1} = \delta^U_{e_{ij}} + \ln\left(\delta^M_{e_{ij}} + 1\right) \cdot \ln\left(\delta^R_{e_{ij}} + 1\right) \quad (2.9.8)$$

$$F_{4,2} = (\delta^U_{e_{ij}} + 1) \cdot \ln\left(\delta^M_{e_{ij}} + 1\right) + \ln\left(\delta^R_{e_{ij}} + 1\right) \quad (2.9.9)$$

$$F_{4,3} = (\delta^U_{e_{ij}} + 1) \cdot \ln\left(\delta^R_{e_{ij}} + 1\right) + \ln\left(\delta^M_{e_{ij}} + 1\right) \quad (2.9.10)$$

$$F_{4,4} = (\delta^U_{e_{ij}} + 1) \cdot \left(\ln\left(\delta^M_{e_{ij}} + 1\right) + \ln\left(\delta^R_{e_{ij}} + 1\right)\right) \quad (2.9.11)$$

$$F_{4,5} = (\delta^U_{e_{ij}} + 1) \cdot \left(2 \cdot \ln\left(\delta^M_{e_{ij}} + 1\right) + 3 \cdot \ln\left(\delta^R_{e_{ij}} + 1\right)\right) \quad (2.9.12)$$

Далее для этих весовых функций были проведены экспериментальные вычисления на основе данных 15 графов, импортированных из сети *Telegram*-каналов. Построение графа $G(V,E)$ остается в той же логике выбора трех факторов взаимодействия и формирования множеств V и E. Для каждой из 15 сетей построено по 12 графов. И для каждого из полученных графов изучено распределение весов их вершин. Обозначим далее каждый из графов, построенных для функции F как $G^i_F(V,E)$, где $i = 1, \dots, 15$. Вес вершины для всех рассматриваемых графов определяется как сумма весов инцидентных ей ребер.

Во многих работах, освещающих аспекты, связанные со сложными безмасштабными сетями, указывается, что одним из свойств таких сетей является степенное распределение весов вершин [1, 2, 3]. Если быть точнее, то важен «хвост» этого распределения. То есть вероятность того, что случайная вершина имеет степень x задается как:

$$P(x) = Cx^{-\gamma}, \quad (2.10)$$

где C – некоторая константа. При этом показатель γ, как правило, для реальных безмасштабных сетей лежит в диапазоне $[2;\ 3]$. Причем для анализа «хвоста» важно соблюдение этого распределения при $x > x_0$ для некоторого значения $x_0 = x_0(G(V, E))$.

Для каждого из полученных 180 графов был проведен следующий вычислительный эксперимент по приближению и оценке эмпирической функции распределения степеней вершин. Для имеющегося у графа распределения весов его вершин («хвоста» этого распределения начиная с некоторого значения x_0) методом максимального правдоподобия находится значение γ с помощью техники, описанной в работах [86, 87].

В таблице 2.2. показаны значения γ для найденных степенных функций распределения в каждом из 180 случаев. Видно, что $\gamma \in [2;\ 3]$ во многих случаях только для ряда функций: $F_{1,1}$, $F_{1,2}$, $F_{3,1}$, $F_{3,2}$ и $F_{3,3}$. Рассмотрим графы, созданные с их использованием. Вычислим расстояние Колмогорова–Смирнова для сравнения наблюдаемого распределения степеней у вершин с полученной функцией. Среди построенных по этим пяти функциям графов было вычислена статистика Колмогорова:

$$D_k = \sup_{x \geq x_0} \left| f(x) - \hat{f}_k(x) \right|, \tag{2.11}$$

где $f(x)$ – полученная функция, а $\hat{f}_k(x)$ – эмпирическая функция распределения, k – размер соответствующей выборки.

В таблице 2.3 показаны значения D_k для рассматриваемых функций и графов. Отсюда можно видеть, что в случае использования $F_{1,1}$ эмпирическая функция распределения хорошо приближается найденной степенной функцией. Поэтому можно сделать утверждение о том, что построенный с использованием $F_{1,1}$ граф удовлетворяет степенному закону распределения весов вершин, свойственному графам сложных безмасштабных сетей.

Таким образом, обоснована целесообразность использования (U, M, R)-модели с $U = 1$, $M = 2$, $R = 3$ для графов взаимодействующих объектов, построенных при импорте данных из сети *Telegram*-каналов.

Таблица 2.2 – Результаты экспериментальных данных для 15 сетей и 12 вариантов весовой функции F

$G_F^i(V,E)$	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	$i=7$	$i=8$	$i=9$	$i=10$	$i=11$	$i=12$	$i=13$	$i=14$	$i=15$
$F_{1,1}$	2,15	2,07	2,36	2,03	2,53	2,06	2,52	2,61	2,91	2,31	2,29	3,08	2,01	3,15	3,82
$F_{1,2}$	1,81	2,05	2,48	2,20	2,47	2,16	2,61	2,62	2,47	2,35	2,29	2,41	2,06	1,99	2,29
$F_{2,1}$	4,25	2,11	3,07	2,82	3,42	7,04	3,41	3,50	3,53	2,25	2,55	3,19	4,20	4,13	2,89
$F_{2,2}$	3,62	2,32	2,87	2,65	3,01	3,59	3,41	3,79	3,63	2,70	2,47	3,75	4,43	2,41	2,72
$F_{3,1}$	1,82	2,37	2,69	4,33	2,42	2,31	2,21	2,44	2,81	2,11	2,24	2,18	2,35	2,02	2,80
$F_{3,2}$	1,87	2,11	2,33	2,62	2,06	2,14	2,24	2,31	2,22	2,30	2,19	2,44	2,22	2,79	2,67
$F_{3,3}$	2,15	2,20	2,59	2,85	2,07	2,14	2,29	2,51	2,39	2,27	2,29	2,13	2,24	2,88	2,72
$F_{4,1}$	2,16	2,62	2,79	1,79	2,62	2,40	3,03	2,83	2,63	4,24	2,39	2,14	2,55	2,45	2,76
$F_{4,2}$	1,85	2,07	2,25	2,11	2,09	2,14	3,24	2,74	2,84	4,10	2,25	2,78	2,38	2,28	2,38
$F_{4,3}$	1,96	2,07	2,31	3,18	2,13	1,95	2,93	2,51	3,29	2,38	2,33	3,08	3,04	2,58	3,71
$F_{4,4}$	1,73	1,91	2,04	2,02	1,98	1,99	2,93	2,30	2,87	2,20	2,16	2,24	2,39	2,21	2,31
$F_{4,5}$	2,06	1,92	1,96	2,04	2,02	1,91	3,26	2,36	2,90	2,41	2,18	2,27	2,40	2,22	2,34

Таблица 2.3 – Значения D_k для рассматриваемых графов

$G_F^i(V,E)$	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$	$i=7$	$i=8$	$i=9$	$i=10$	$i=11$	$i=12$	$i=13$	$i=14$	$i=15$
$F_{1,1}$	0,1105	0,0646	0,0458	0,0649	0,0660	0,0670	0,0553	0,0535	0,0742	0,0509	0,0554	0,0479	0,0552	0,0554	0,0475
$F_{1,2}$	0,0894	0,0655	0,0593	0,0627	0,0584	0,0779	0,0552	0,0609	0,0752	0,0474	0,0517	0,0592	0,0643	0,0448	0,0524
$F_{3,1}$	0,0962	0,0657	0,0446	0,0748	0,0718	0,0460	0,0597	0,0550	0,0741	0,0650	0,0446	0,0572	0,0527	0,0453	0,0396
$F_{3,2}$	0,1146	0,0569	0,0547	0,0598	0,0737	0,0543	0,0668	0,0452	0,0509	0,0518	0,0494	0,0663	0,0564	0,0372	0,0501
$F_{3,3}$	0,1192	0,0733	0,0481	0,0635	0,0798	0,0657	0,0637	0,0493	0,0625	0,0598	0,0608	0,0727	0,0549	0,0520	0,0510

2.6 Выводы по главе 2

1. Предложенная в работе модель формирования графа взаимодействующих объектов, применима при импорте данных из соответствующих коммуникационных сетей. При этом ее вариативная составляющая позволяет адаптировать модель под разные сети и меняющиеся со временем их особенности. Описанное построение такого графа как в целом, так и для частных случаев рассматриваемых сетей позволяет применять методику и в перспективе возникающих в будущем аналогичных сетей рассматриваемой сферы.

2. Предложенный в работе подход по использованию неориентированного взвешенного графа с вершинами, обладающими атрибутами, позволяет производить численные подсчеты и применять различные методы для его дальнейшего анализа. С учетом специфики исходных данных, получаемых из сетей коммуникации, это позволяет вести работу с большими текстовыми данными и использовать для их анализа методы компьютерной лингвистики.

3. С учетом проведенных экспериментальных исследований разработанной модели и ее вариаций для конкретных сетей можно констатировать применимость модели для прикладных задач.

4. Основные результаты, представленные в главе 2, опубликованы в работах [63, 64, 66, 67, 71, 72, 73, 74, 75, 76]. В данных работах соискателю принадлежит разработка модели формирования взвешенного графа взаимодействующих объектов для различных коммуникационных сетей, вариации этой модели для конкретных сетей: социальной сети *ВКонтакте*, сети коротких сообщений *Twitter*, сети *Telegram*-каналов, методика применения этой модели и ее вариаций. Соискателем предложен общий подход к формированию графов взаимодействующих объектов.

ГЛАВА 3 КОМБИНИРОВАННЫЙ АЛГОРИТМ ВЫДЕЛЕНИЯ СООБЩЕСТВ

В данной главе рассмотрены свойства классического агломеративного иерархического алгоритма. Показано отсутствие тривиальных сообществ в результатах его применения, а также с помощью явных вычислений продемонстрировано свойство, названное автором «сбор мусора».

Представлены предложенные автором модификации классического алгоритма на основе оценки энтропии сети [62, 90, 91] и модификации классического агломеративного иерархического алгоритма [92, 93, 94]. Автором продемонстрированы результаты применения модификаций, предложены методики применения разработанных алгоритмов [95, 96].

В 3 главе представлен разработанный автором «Комбинированный алгоритм», который позволяет выделять пересекающиеся и вложенные сообщества на графе [69, 88, 89]. В главе описаны и проиллюстрированы примеры с результатами работы «Комбинированного алгоритма», продемонстрированы его особенности, позволяющие находить структуру сообществ, не фиксируемую классическими алгоритмами.

3.1 Модулярность

В Главе 1 было указано, что стандартный подход по выявлению структуры сообществ на графе предполагает использование введенного специального функционала, названного «модулярность» [42, 47, 48]. Так как сообщества предполагают большую плотность внутри, чем между ними, то модулярность оценивает схожий показатель. А именно, с ее помощью вычислительно измеряется разница между фактическим количеством ребер внутри сообществ и ожидаемым количеством ребер, если бы они были распределены случайным образом при сохранении базовых свойств рассматриваемого графа.

Пусть есть граф G, в котором число ребер равно m. За A_{ij} обозначим элементы матрицы смежности графа G. Пусть на этом графе G выделены какие-то сообщества. Тогда за S_i обозначим то сообщество, которое содержит вершину i. Для оценки выделенных сообществ на графе G возьмем некоторый случайный граф, сохраняющий базовые свойства G, и обозначим за P_{ij} ожидаемое условное число ребер между вершинами i и j в таком случайном графе. Определим тогда модулярность следующим образом:

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - P_{ij}) \cdot \delta(S_i, S_j), \tag{3.1}$$

где δ – дельта функция.

Остается вопрос о том, какие именно базовые свойства графа G должны быть сохранены в случайном графе и как тогда определить P_{ij}. Согласно модели Ньюмана-Гирвана [42, 46-48] для каждой вершины i сохраняется ее степень d_i. Вместе с сохранением этого условия ребра распределяются в графе случайным образом. В данной модели P_{ij} вычисляется по следующей формуле:

$$P_{ij} = \frac{d_i \cdot d_j}{2m} \tag{3.2}$$

Тогда модулярность при сохранении тех же обозначений определяется по следующей формуле:

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{d_i \cdot d_j}{2m}) \cdot \delta(S_i, S_j), \tag{3.3}$$

Данная формула используется для случая выделения непересекающихся сообществ. Для простых взвешенных графов формула (3.3) интерпретируется следующим образом. За m берется сумма весов всех ребер графа, а под d_i понимаем сумму весов ребер, смежных с вершиной i.

Некоторые методы нахождения сообществ используют для этого поиск максимума Q. При этом нахождение глобального максимума Q является NP-полной задачей в сильном смысле [97]. Поэтому используются алгоритмы поиска локального максимума модулярности [50, 58, 93, 98, 99, 100]. Одним из таких является алгоритм *Louvain* (в некоторых работах называемый алгоритмом Блонделя [51] по фамилии одного из его авторов). Далее в разделе 3.4 проведен анализ особенностей

указанного алгоритма и возможные его модификации, предложенные в работах [88, 92, 93, 94].

В случае выделения на графе пересекающихся сообществ рассмотрим следующий вариант подсчета модулярности [101]:

$$Q^{overlap} = \frac{1}{2m}\sum_i \sum_{v\in S_i, w\in S_i} \frac{1}{O_v O_w}\left[A_{vw} - \frac{d_v \cdot d_w}{2m}\right], \qquad (3.4)$$

где A_{vw} – элемент матрицы смежности графа G;

m – число ребер в графе G;

d_v – степень вершины v;

O_v – число сообществ, содержащих вершину v;

$S = \{S_i\}$ – множество всех выделенных на графе пересекающихся сообществ.

Далее в этой главе будем под модулярностью для выделения непересекающихся сообществ подразумевать Q, заданную согласно (3.3), а под модулярностью для выделения пересекающихся сообществ $Q^{overlap}$, заданную согласно (3.4).

3.2 Алгоритм на основе случайного блуждания

Как и указано в Главе 1, помимо модулярности существуют иные подходы для поиска сообществ на графах. В данном разделе описан алгоритм Росваля-Бергстрома [52, 53, 55], а в последующем разделе 3.3 описана его модификация, предназначенная для выделения непересекающихся сообществ взвешенного графа взаимодействующих объектов. Модификация была предложена в работах [90, 91] и применялась для анализа информационного воздействия [95, 96].

Вначале опишем базовый алгоритм Росваля-Бергстрома [52, 53, 102]. Данный классический алгоритм, называемый в литературе также *Infomap,* основан на сжатии информации о процессе случайного блуждания в графе за счет его кодирования. Если присваивать самым часто встречающимся в процессе блуждания вершинам самые короткие кодовые слова, а наименее частым – самые длинные, то можно

минимизировать количество информации, необходимой для описания пути случайного блуждания.

Так как плотность связей внутри сообществ высока, то логично исходить из предположения о том, что в процессе случайного блуждания статистически больше времени проводится внутри сообществ, а не при перемещении от сообщества к сообществу [103, 104]. Тогда используя код Хаффмана и двухуровневое разбиение информации, при котором первая составляющая кодирует выделенные сообщества, а вторая часть – вершины внутри них, можно осуществить сжатие информации о случайном блуждании.

Разбиение графа G на непересекающиеся сообщества $S = \{S_i\}$ дает минимизацию верхней границы кодового слова. Будем далее обозначать ее $L(S)$. Поэтому для оценки выделения множества S достаточно будет посчитать $L(S)$ без проведения процедуры кодирования. Так исходная задача поиска сообществ на G сведена к минимизации $L(S)$.

Для вычисления показателя $L(S)$ качества разбиения на сообщества используем энтропию, описывающую среднюю длину кодового слова. Пусть граф G состоит из n вершин. Тогда для случайной величины X, принимающей n значений с вероятностями p_j, где $j = 1, \dots, n$ оценка минимальной длины кодового слова согласно теории информации Шеннона [105] определяется энтропией X как:

$$H(X) = -\sum_{j=1}^{n} p_j \cdot log\, p_j \tag{3.5}$$

Для заданного найденного $S = \{S_i\}$, где $i = 1, \dots, m$ обозначим $|S_i| = n_i$. Определим случайную величину Y, принимающую натуральные значения от 1 до m, и набор случайных величин Z^i, принимающих натуральные значения от 1 до n_i соответственно. Тогда для $L(S)$ будет выполнено следующее:

$$L(S) = \sum_{j=1}^{m} q_j\, H(Y) + \sum_{i=1}^{m} p_i H(Z^i) \tag{3.6}$$

q_i – вероятность покинуть сообщество S_i;

$p_i = \sum_{\alpha \in S_i} p_\alpha + q_i$ – вероятность остаться в сообществе S_i;

p_α – вероятность посетить вершину α.

При этом имеем следующее выражение для нижней границы средней длины кодового слова, кодирующей сообщество, через энтропию переходов между ними:

$$H(Y) = -\sum_{i=1}^{m} \frac{q_i}{\sum_{j=1}^{m} q_j} log\left(\frac{q_i}{\sum_{j=1}^{m} q_j}\right), \quad (3.7)$$

Также для нижней границы средней длины кодового слова внутри сообщества имеем:

$$H(Z^i) = \frac{q_i}{q_i + \sum_{\beta \in i} p_\beta} log\left(\frac{q_i}{q_i + \sum_{\beta \in i} p_\beta}\right) - \sum_{\alpha \in i} \frac{p_\alpha}{q_i + \sum_{\beta \in i} p_\beta} log\left(\frac{p_\alpha}{q_i + \sum_{\beta \in i} p_\beta}\right) \quad (3.8)$$

Подставляя в формулу (3.6) выражения по формулам (3.7) и (3.8) получаем для $L(S)$ следующее выражение:

$$L(S) = \left(\sum_{i=1}^{m} q_i\right) log\left(\sum_{i=1}^{m} q_i\right) - 2\sum_{i=1}^{m} q_i \, log(q_i) - \sum_{\alpha=1}^{n} p_\alpha \, log(p_\alpha) + \sum_{i=1}^{m} \left(q_i + \sum_{\alpha \in S_i} p_\alpha\right) log\left(q_i + \sum_{\alpha \in S_i} p_\alpha\right), \quad (3.9)$$

Вычисление $L(S)$ по формуле (3.9) производится быстро благодаря хранению результатов промежуточных вычислений для отдельных слагаемых.

Для взвешенных графов можно дополнительно учесть значение w_α – веса вершины α. Если вес ребра e, смежного с вершиной α, обозначить как w_e, то w_α определим следующим образом:

$$w_\alpha = \sum_{e \in E_\alpha} w_e \ , \quad (3.10)$$

где E_α – множество всех ребер, смежных с α.

Тогда для веса сообщества S_i получаем следующее выражение:

$$w(S_i) = \sum_{\alpha \in S_i} w_\alpha \quad (3.11)$$

Рассмотрим E_{S_i} – множество ребер, у которых одна из вершин лежит в сообществе S_i, а вторая вершина не принадлежит сообществу S_i. Тогда за $w^{ex}(S_i)$ – обозначим суммарный «вес выхода» из S_i:

$$w^{ex}(S_i) = \sum_{l \in E_{S_i}} w_l \ , \quad (3.12)$$

Наконец, обозначим суммарный вес всех таких ребер, соединяющих сообщества:

$$w^{exit} = \frac{\sum_{i=1}^{m} w^{ex}(S_i)}{2} \quad (3.13)$$

Формулы (3.10) – (3.12) применим в формуле (3.9) и получим выражение для показателя качества разбиения $L(S)$:

$$L(S) = w^{exit}\, log\left(w^{exit}\right) - 2\sum_{i=1}^{m} w^{ex}(S_i)\, log(w^{ex}(S_i)) - \sum_{\alpha=1}^{n} w_\alpha\, log(w_\alpha) + \sum_{i=1}^{m}(w^{ex}(S_i) + w(S_i))\, log(w^{ex}(S_i) + w(S_i)) \quad (3.14)$$

Для поиска минимума $L(S)$ можно использовать, например, жадный алгоритм.

3.3 Итерационный алгоритм с модифицированными весами

Рассмотрим теперь модификацию этого алгоритма пользуясь тем, что при построении графов взаимодействующих объектов, импортированных из коммуникационных сетей, по модели, описанной в Главе 2, вершинам приписываются атрибуты, содержащие текстовые данные.

Пусть для вершин построенного графа G имеется множество AT типов существенных для выделения сообществ показателей или атрибутов. Это не означает, что каждая вершина имеет все значения этих атрибутов. Для двух произвольных вершин α и β можно определить $A_{common}(\alpha, \beta)$ – множество тех атрибутов, которые имеются в наличии у обеих рассматриваемых вершин. Тогда его подмножество $A_{equal}(\alpha, \beta)$ представляет собой все те атрибуты, значения по которым у α и β совпадают.

Теперь можно определить $\mu_{AT}(\alpha, \beta) = 1 - \frac{|A_{equal}(\alpha,\beta)|}{|A_{common}(\alpha,\beta)|}$ – функцию, характеризующую близость двух вершин α и β. Очевидно, что ее значения лежат на отрезке $[0; 1]$. При этом нулевое значение она принимает только для случая, когда значения всех общих атрибутов α и β у совпадают.

Тогда вес, определенный по формуле (3.11) для сообщества S_i зададим следующим образом:

$$w(S_i) = \left(\sum_{\alpha,\beta \in S_i} \frac{1-\mu_{AT}(\alpha,\beta)}{|S_i|^2}\right) \cdot \sum_{\alpha \in S_i} w_\alpha \quad (3.15)$$

Используем полученное выражение для $w(S_i)$ в формуле (3.14) для итерационного вычисления $L(S)$ для графов взаимодействующих объектов. В реальных сетях коммуникации у объектов, как правило, имеется достаточно большое число атрибутов.

В результате приходим к следующему итерационному алгоритму вычисления $L(S)$ и конструирования множества сообществ $S = \{S_i\}$.

Алгоритм 3.1.

Шаг 1. Подсчет $L(S)$ для исходного состояния, когда каждая вершина лежит в своем собственном сообществе.

Шаг 2. Формирование множества сообществ $S' = \{S_i\}$ по частоте вершин во время случайного блуждания.

Шаг 3. Пересчет $L(S')$ для нового S'. Если $L(S') < L(S)$, то сохраняем S' как S, переход к шагу 2. Иначе переход к шагу 4.

Шаг 4. На графе выделено множество $S = \{S_i\}$.

Вычислительные эксперименты применения алгоритма выделения сообществ на основе случайного блуждания проводились сгенерированных графах, а также на графах реальных коммуникационных сетей.

Тестирование алгоритма на случайно сгенерированных графах *LFR*-модели [59, 106, 107] проводилось следующим образом.

Обозначим количество вершин за N, а степень произвольной вершины j за k_j. Фиксированные константные параметры *LFR*-модели, используемые при генерации обозначим как $c_{1,2}$ и $\tau_{1,2}$. В *LFR*-модели исходно при создании графов заложена структура сообществ, при этом их размеры n_i распределены по степенному закону:

$$P(n_i = x) = c_1 \cdot x^{-\tau_1} \tag{3.16.1}$$

Аналогичное распределение и у степеней вершин в *LFR*-модели:

$$P(k_j = y) = c_2 \cdot y^{-\tau_2} \tag{3.16.2}$$

В ходе тестирования быстродействия алгоритма использовались графы, сгенерированные при следующих параметрах: $N = 10000$, $\max\limits_{j} n_j = 1000$, $\min\limits_{j} n_j = 100$, $\max\limits_{i} k_i = 40$. При этом вероятность построения ребра между двумя вершинами из разных сообществ была взята равной $\frac{1}{10}$. В тестовых графах варьировалась средняя степень вершин, что влияло на число ребер.

Суть исследования заключалась в рассмотрении зависимости времени выполнения алгоритма от $< k >$ на сгенерированных при таких параметрах графах. Измерения скорости выделения сообществ для графов с различными степенями вершин показаны на рисунке 3.1. Полученная зависимость может быть приближена линейной функцией.

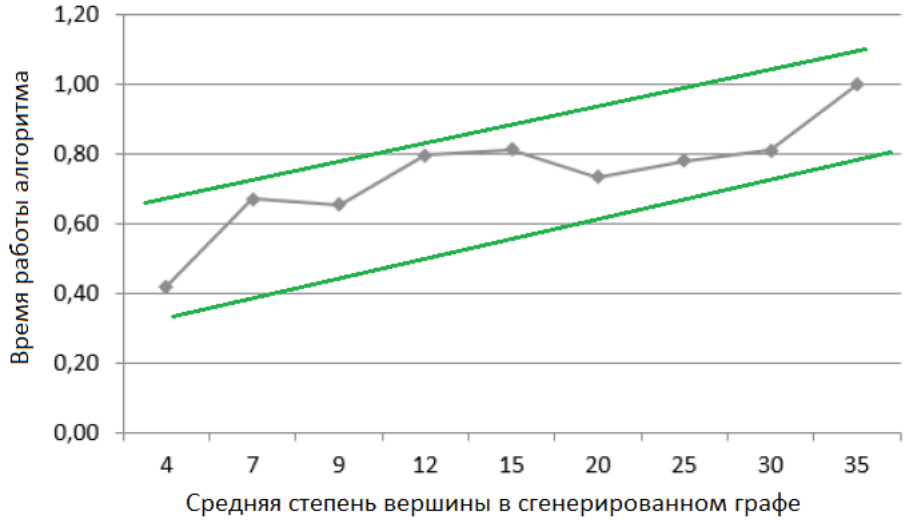

Рисунок 3.1 – Время работы (в секундах) алгоритма на сгенерированных графах с разными значениями $< k >$

Таким образом, экспериментально установлена линейная зависимость времени выполнения алгоритма от средней степени вершины графа. Важным тут является то, что это искусственно сгенерированные графы, которые как было сказано в Главах 1 и 2 отличаются от графов, полученных из реальных сетей.

Алгоритм был применен и на графах, полученных при импорте реальных данных из социальной сети *ВКонтакте* в соответствии с моделью, описанной в главе 2, где в качестве фактора взаимодействия при построении графа используется

только взаимная подписка пользователей. Тогда как остальные атрибуты используются уже в процессе самого алгоритма. Например, возьмем два импортированных так графа G_1 и G_2, представленных в таблице 3.1.

Таблица 3.1 – Показатели графов, на которых выделены сообщества

	Число вершин графа	Число ребер графа
G_1	2 000	43 004
G_2	480	493

Покажем эти графы вначале со случайным размещением до выделения сообществ (рисунки 3.2 и 3.4). А также после выделения алгоритмом сообществ (рисунки 3.3 и 3.5). Тут представление графов дано с помощью метода физических аналогий.

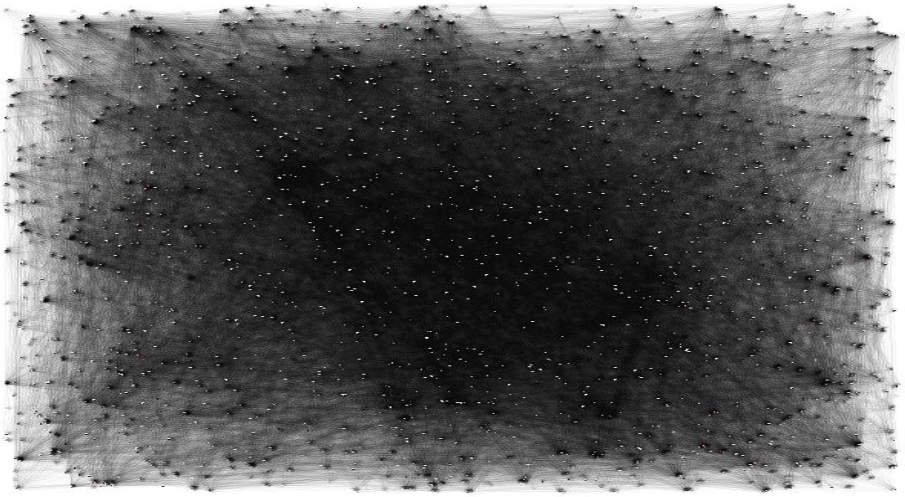

Рисунок 3.2 – Случайное размещение для графа G_1

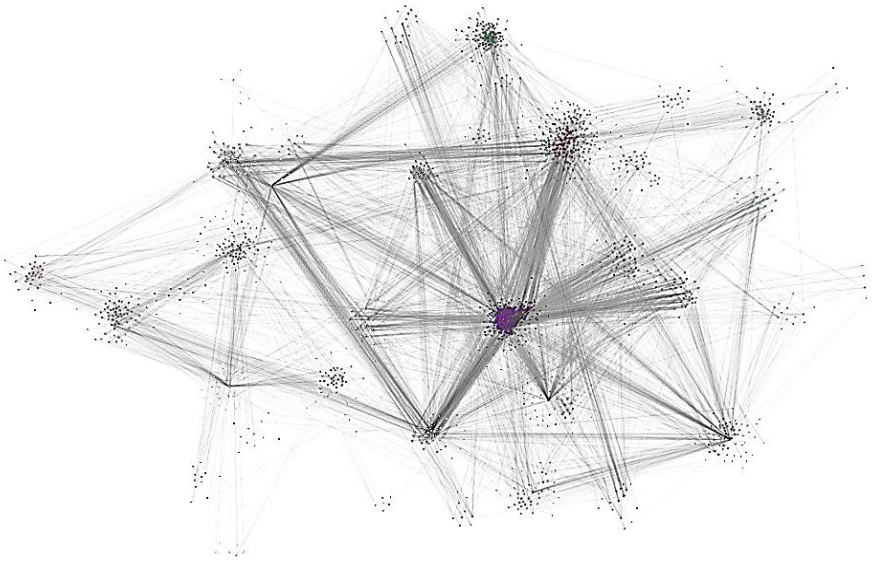

Рисунок 3.3 – Граф G_1 разбит на сообщества

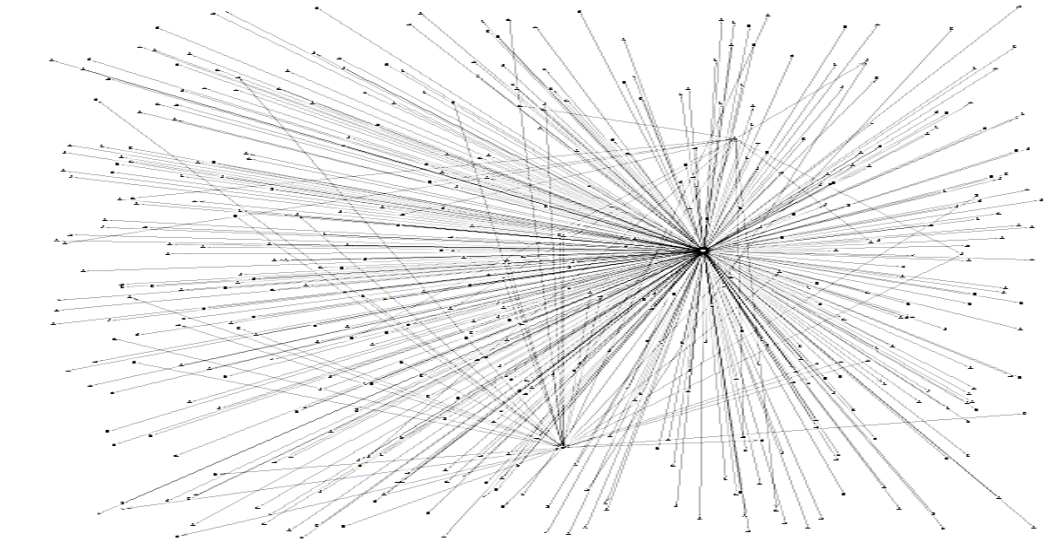

Рисунок 3.4 – Случайное размещение для графа G_2

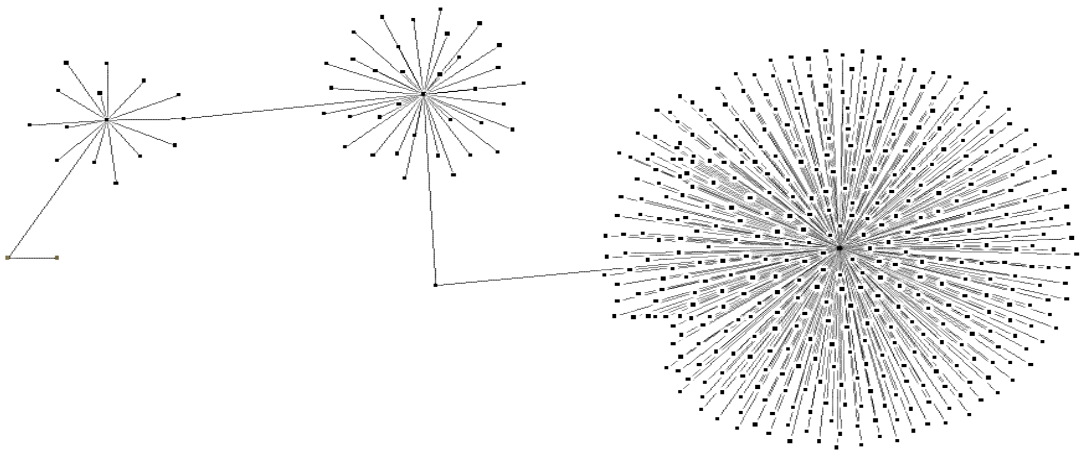

Рисунок 3.5 – Граф G_2 разбит на сообщества

3.4 Модификации алгоритма Louvain

В данном разделе рассмотрим подробнее алгоритм *Louvain* (или как иногда указывается – алгоритм Блонделя по фамилии одного из его авторов) и его модификации. Алгоритм состоит из двух повторяющихся итерационно этапов. Прежде чем представить модификацию этого классического агломеративного иерархиче-

ского алгоритма опишем его для введения соответствующих обозначений. Рассматривается взвешенный граф G, стандартно обозначаем за V множество его вершин. Сами вершины в этом разделе для простоты обозначений будем указывать одной строчной буквой, например i. Алгоритм находит такой набор непересекающихся сообществ, на котором достигается локальный максимум модулярности Q определенной по аналогии с (3.3).

Общая схема алгоритма *Louvain* состоит в следующем. Изначально каждая вершина графа G лежит в своем собственном сообществе. Первый этап заключается в том, чтобы среди вершин j, соседних с вершиной i, посмотреть на возможный прирост модулярности ΔQ, который может иметь место при удалении вершины i из своего сообщества и добавления ее в сообщество вершины j. Вершина i переносится в то сообщество, где достигается максимальный положительный прирост ΔQ. Если положительного прироста не существует, то вершина i остается на исходном месте. Данный процесс повторяется итерационно и последовательно для всех вершин графа G до тех пор, пока возможно положительное ΔQ. На втором этапе алгоритма из найденных сообществ строится новый граф, за вершины которого берутся эти сообщества, определенные на первом этапе. После к новому графу применяется первый этап алгоритма. Алгоритм *Louvain* можно записать следующим образом:

Алгоритм 3.2.

Шаг 0. Присвоить каждой вершине $i \in V$ собственное сообщество: S_i.

Шаг 1. Для каждой вершины i:

Для каждого ее соседа j подсчитать изменение модулярности при перемещении вершины i из текущего сообщества S_i в сообщество S_j и найти максимальный положительный прирост: $\max_j \Delta Q$.

Вершина i переносится в сообщество, дающее максимальный положительный прирост модулярности.

Повторять шаг 1 до тех пор, пока возможно увеличить Q.

В случае, если значение модулярности на втором шаге изменялось, перейти к шагу 2, иначе – выход.

Шаг 2. Создать новый граф: вершины – объединение вершин каждого сообщества. Вес ребер определять через сумму весов ребер между соответствующими сообществами в исходном графе, а вес петли новой вершины через сумму весов ребер внутри этого сообщества.

Запустить на полученном новом графе шаг 1.

Так как суть алгоритма 3.2 в том, что для каждой вершины необходимо подсчитать возможное изменение модулярности при переносе ее из своего сообщества в сообщество каждой из соседних с ней вершин. Изменение модулярности в таком случае складывается из двух компонент: изменение от удаления вершины i из ее текущего сообщества S_i и изменение от добавления вершины в новое сообщество S_j. Изменение модулярности при добавлении вершины i в произвольное сообщество C рассчитывается следующим образом:

$$\Delta Q = \frac{k_i^C}{m} - \frac{\Sigma_{tot}^C \cdot d_i}{2m^2} \tag{3.17}$$

где k_i^C – сумма весов ребер, инцидентных i и сообществу C;

m – сумма весов ребер графа;

Σ_{tot}^C – сумма степеней вершин, принадлежащих сообществу C;

d_i – степень вершины i.

Особенности результатов применения алгоритма 3.2 связаны с некоторыми свойствами модулярности. Запишем на основании (3.17) условие для переноса вершины из тривиального сообщества в сообщество C:

$$k_i^C - \frac{\Sigma_{tot}^C \cdot d_i}{2m} > 0 \tag{3.18}$$

Как показано в работах [72, 92] на простом связном невзвешенном графе выделение сообществ, максимизирующее функционал модулярности согласно алгоритму 3.2, не содержит тривиальных сообществ. В частности, получается, что листовая вершина в графе всегда объединяется с сообществом ее единственного соседа, так как иначе эта вершина лежала бы в тривиальном сообществе. А наличие тривиальных сообществ уменьшает значение модулярности. Из этого следует, что

при построении согласно модели из главы 2 эгографа от одной стартовой вершины все листовые вершины графа будут лежать в одном сообществе с исходной вершиной.

Более того, при выборе, к какому из уже сформированных сообществ добавить вершину, алгоритм 3.2 неявно основывается на суммарном весе ребер, инцидентных вершинам этих сообществ. Поэтому не редка ситуация, когда какая-то из вершин, имеющая большую степень, относится алгоритмом не к тому сообществу, с которым у нее больше всего общих ребер. Это приводит к тому, что вершины, инцидентные большому числу листов, которые в силу отсутствия тривиальных сообществ объединяются с ними в одном сообществе, попадают в сообщества с малым суммарным весом. Эти свойства приводят к разбиению, которое можно назвать «сбором мусора». Что в зависимости от исходной задачи анализа сети может быть неудачным разбиением.

Возможной методикой выделения сообществ на графе с целью борьбы со свойством «сбора мусора» является учет для вершин графа их степени во всей исходной сети. Результаты такого подхода приведены в следующем разделе данной главы.

Другим важным свойством модулярности является адаптация ее под размер сети. Для начала отметим, что при объединении двух сообществ i и j в новое сообщество изменение модулярности ΔQ_{C} рассчитывается по формуле:

$$\Delta Q_{\mathrm{C}} = \frac{k_i^j}{m} - \frac{\Sigma_{tot}^i \cdot \Sigma_{tot}^j}{2m^2}, \tag{3.19}$$

где k_i^j – число ребер между сообществами;

Σ_{tot}^i – степень сообщества i;

Σ_{tot}^j – степень сообщества j;

Отсюда следует, что при увеличении размеров графа, объединение малых сообществ будет увеличивать модулярность разбиения. Более того, из формулы (3.19) видно, что при рассмотрении двух связанных сообществ, степень каждого из которых меньше, чем $\sqrt{2m}$, их объединение приведет к увеличению модулярности. В

этом и заключается «предел разрешения» модулярности [3, 57] – одно из ее свойств, которое часто рассматривается как недостаток: в максимальном по модулярности разбиении невзвешенного неориентированного графа не может существовать двух связанных сообществ, степень каждого из которых меньше, чем $\sqrt{2\mathrm{m}}$.

Одним из возможных решений для выделения малых по размеру сообществ является параметризация модулярности [92]:

$$Q(\alpha) = \frac{1}{2m}\sum_{ij}(A_{ij} - \alpha\frac{d_i \cdot d_j}{2m}) \cdot \delta(S_i, S_j), \tag{3.20}$$

где параметр $0 < \alpha \leq 1$.

Тогда $\Delta Q_{\mathrm{C}}(\alpha)$ будет рассчитываться по формуле:

$$\Delta Q_{\mathrm{C}}(\alpha) = \frac{k_i^j}{m} - \alpha\frac{\Sigma_{tot}^i \cdot \Sigma_{tot}^j}{2m^2}. \tag{3.21}$$

Следовательно, минимальный размер двух связанных сообществ при использовании параметризованной модулярности будет равен $\sqrt{2m/\alpha}$, а значит, получится выделить сообщества меньшего размера. Поэтому еще одной методикой по выделению сообществ на графе является учет параметризированной модулярности вместо обычной. Пример такого подхода приведен в следующем разделе главы 3.

Отметим, что разбиение, получаемое алгоритмом 3.2 на заданном графе, зачастую не обладает абсолютно максимальной модулярностью для этого графа, но указанные свойства разбиения при этом выполняются. Это связанно с тем, что отсутствие тривиальных сообществ и существование предела разрешения опираются на величину прироста модулярности, расчет которой является основным шагом работы алгоритма.

Опишем задачу выбора, к какому из уже сформированных сообществ добавить вершину. Рассмотрим ситуацию, когда для вершины i выбирается сообщество, в которое ее можно добавить. Возможное приращение модулярности при добавлении вершины i в сообщество S_j характеризуется значением показателя, полученного в [92]:

$$s_i^j = k_{i,in}^j - \frac{\Sigma_{tot}^j \cdot k_i}{2m}, \quad (3.22)$$

где k_i – вес вершины i;

Σ_{tot}^j – сумма весов ребер, лежащих внутри сообщества S_j;

$k_{i,in}^j$ – сумма весов ребер, инцидентных вершине i и сообществу S_j;

m – сумма весов всех ребер графа.

В случае если s_i^j положительно для некоторой вершины i и некоторого ее сообщества-соседа S_j, то включение этой вершины увеличит значение модулярности на величину s_i^j/m. Сообщество-сосед S_j вершины i, для которого значение s_i^j будет максимальным, будет выбрано алгоритмом данном шаге. Отсюда получаем некоторое свойство этого алгоритма, которое заключается в том, что выбор, к какому из уже сформированных сообществ добавить вершину, определяется суммарным весом ребер, инцидентных вершинам этих сообществ. Сравним значения показателя s_i^j, чтобы определить, от чего зависит выбор алгоритма.

Рассмотрим возможные приращения модулярности при добавлении вершины i в сообщества S_1 и S_2. Это определяется согласно (3.22) значениями s_i^1 и s_i^2 соответственно:

$$s_i^1 = k_{i,in}^1 - \frac{\Sigma_{tot}^1 \cdot k_i}{2m} \quad (3.23.1)$$

$$s_i^2 = k_{i,in}^2 - \frac{\Sigma_{tot}^2 \cdot k_i}{2m} \quad (3.23.2)$$

Тогда получаем для их разности следующее соотношение:

$$s_i^2 - s_i^1 = \left(k_{i,in}^2 - k_{i,in}^1\right) + \frac{k_i}{2m}\left(\Sigma_{tot}^1 - \Sigma_{tot}^2\right) \quad (3.24)$$

Из (3.24) видно, что при равенстве $k_{i,in}^2 = k_{i,in}^1$ суммарных весов связей между вершиной i и обоими сообществами для присоединения ее к S_2 необходимо выполнение неравенства $\Sigma_{tot}^1 > \Sigma_{tot}^2$. Таком образом, в данном случае вершина будет присоединена к сообществу с меньшим весом.

Далее, рассмотрим более общий случай, когда $k_{i,in}^2 \neq k_{i,in}^1$. Оценим суммарные веса сообществ через некоторые константы λ_j для S_j:

$$\Sigma_{tot}^{j} = \lambda_j m \tag{3.25}$$

Условием того, что алгоритм выберет сообщество S_2, будет выполнение неравенства $(s_i^2 - s_i^1) > 0$, что с учетом (3.24) и (3.25) дает следующее соотношение:

$$k_{i,in}^1 - k_{i,in}^2 < \frac{k_i}{2}(\lambda_1 - \lambda_2) \tag{3.26}$$

Отсюда видно, что в случае, даже если число связей у i -ой вершины с S_1 больше, чем с S_2, то при большом суммарном весе ребер S_1 алгоритм 3.2. отправит i -ую вершину в сообщество S_2. Поэтому не редка ситуация, когда какая-то из вершин с большой степенью относится алгоритмом не к тому сообществу, с которым у нее больше всего инцидентных ребер. В силу указанного ранее имеем, что вершины, у которых много соседей-листов, а значит лежащих с ними в одном сообществе, попадают на следующем этапе в сообщества с малым суммарным весом. В этом и заключается «сбор мусора».

Стоит отметить, что алгоритмы выделения сообществ зачастую имеют дело с некоторым подграфом социальной сети, размер которой может достигать миллиардов вершин. Результатом импорта данных является подграф G, соответствующий подсети, для части вершин которого не были получены их соседи. Степень этих вершин в подграфе, как правило, значительно меньше исходной. Зачастую, подобные вершины исключаются из рассмотрения ввиду их не информативности. Полученный таким образом граф обозначим G'.

Рассмотрим некоторую вершину графа. Пусть d_G — ее степень в G, а $d_{G'}$ — степень в G'. Предположим, что $d_{G'} = 1,\ d_G >> d_{G'}$, то есть вершина имеет единственного соседа в G' и при этом ее реальная степень во всей исходной сети достаточно велика. Алгоритм 3.2. всегда добавляет листовую вершину к сообществу соседа. В данной ситуации рассматриваемая вершина слабо связана с графом G' и ее добавление в сообщество соседа может быть нежелательным для задачи исследования исходной сети. Отсюда возникает потребность учета d_G при работе с графом G'. Одним из решений является использование d_G в качестве веса вершины при расчете модулярности.

3.5 Тесты алгоритма и его модификаций

Для сравнения структуры сообществ, заложенных в генерируемом по *LFR*-модели [59, 106, 107] графе, с полученными в результате работы алгоритма разбиением использовалась нормализованная взаимная информация *NMI (Normalized Mutual Information,*), предложенная в качестве меры сходства разбиений графа в [58]. Тестовые *LFR*-графы были сгенерированы со следующим числом вершин: от 1000 до 15000 с шагом 1000. Максимальный размер сообщества – 50 вершин. Усреднение производилось на 10 графах.

Рассматриваемый алгоритм является иерархическим, следовательно, в разбиениях больших графов крупные сообщества, расположенные на последнем уровне иерархии, объединяют несколько более малых, полученных на более низких уровнях иерархии. Последний уровень иерархии разбиения алгоритма не содержит сообществ меньше определенного размера, даже если последние слабо связаны с остальным графом. Однако, данные сообщества могут быть обнаружены на более низких уровнях иерархии. Как видно на рисунке 3.6 самостоятельное сообщество, отмеченное синей окружностью, на втором уровне иерархии объединяется с центральным сообществом (большое сообщество внизу рисунка).

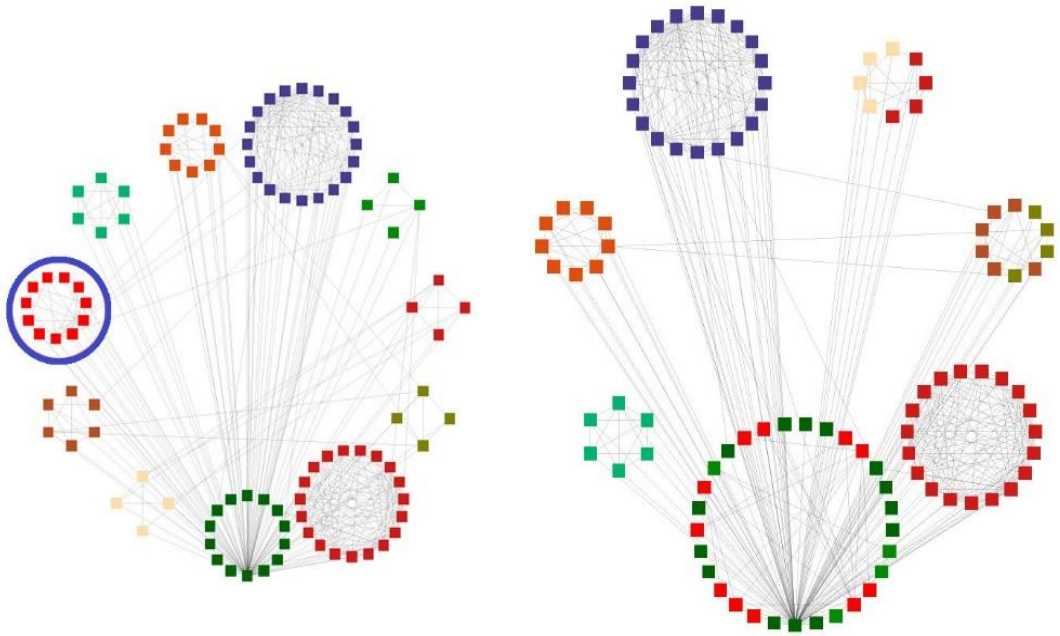

Рисунок 3.6 – Первый (слева) и второй (справа) уровни иерархии разбиения на сообщества.

Красное сообщество с левой части рисунка 3.6 объединяется в правой его части с зеленым сообществом. Объединенное сообщество расположено внизу правой части рисунка.

На *LFR*-моделях работа с более низкими уровнями иерархии алгоритма *Louvain* показала (рисунок 3.7) наилучшие результаты при малой вариативности весов вершин и сообществ, что, однако не находит своего отражения в реальных сложных сетях.

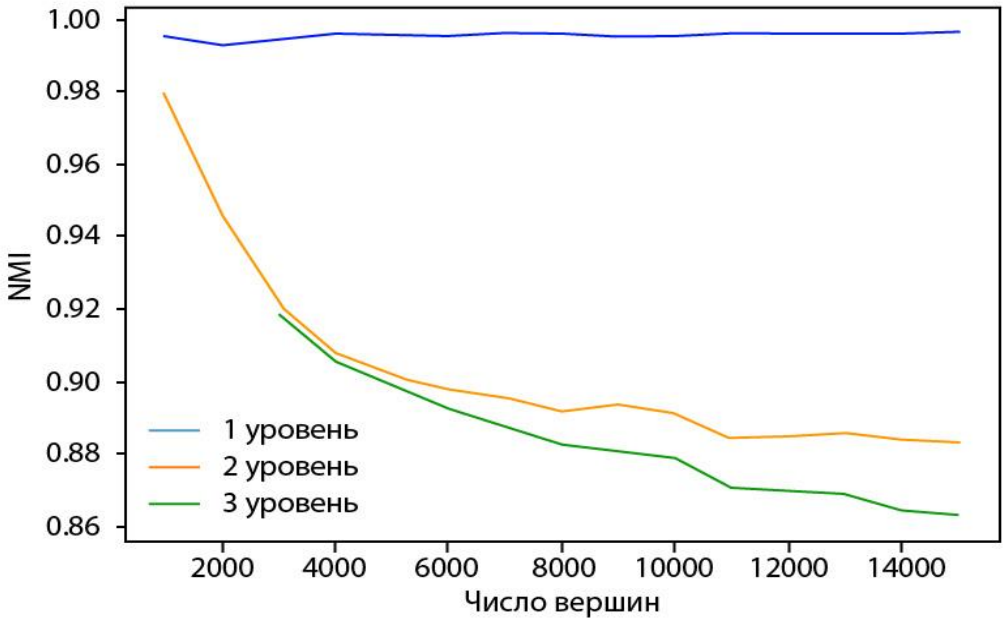

Рисунок 3.7 – Значения *NMI* на различных уровнях иерархии

Одной из методик, позволяющих уменьшить негативное влияние фактора под названием «сбор мусора», при работе с графами, полученными при импорте подсети, является учет степеней вершин графа в соответствии с исходной сетью вместо степеней вершин полученного графа подсети [92]. Для тестирования данной модификации были рассмотрены подграфы сгенерированных *LFR*-моделей путем обхода в ширину от случайной вершины. Сравнение вариантов, с учетом и без учета реальных степеней вершин, показывает преимущество рассматриваемого варианта (рисунок 3.8). Что подтверждает целесообразность использования данной модификации в некоторых случаях.

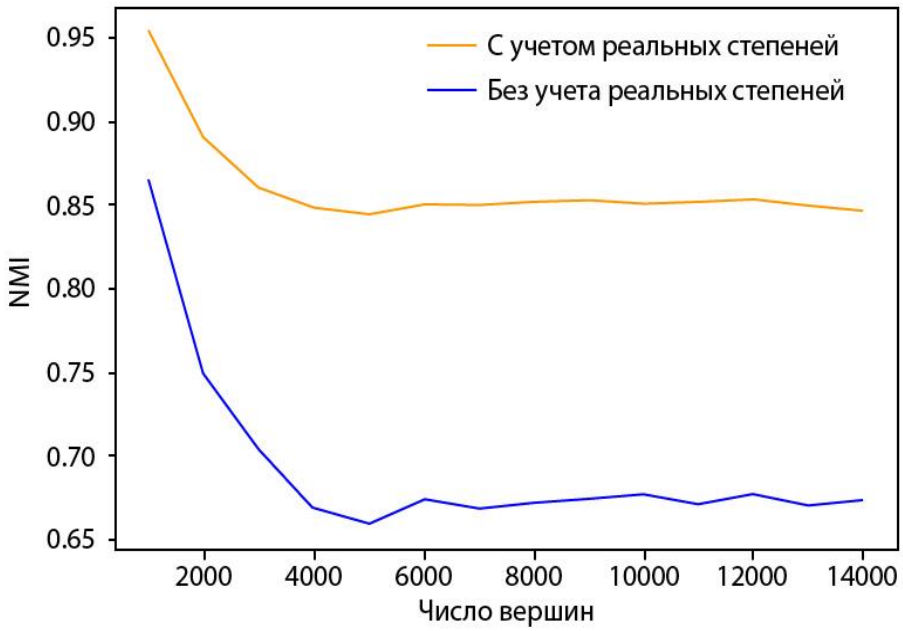

Рисунок 3.8 – Сравнение модификации, учитывающей реальный вес вершины с оригинальным алгоритмом

В рамках тестирования модификации (3.20) на реальных данных в качестве графа взаимодействующих объектов был рассмотрен эгограф друзей пользователя сети *Instagram*[1]. На графе выделили сообщества без масштабирующего коэффициента (рисунок 3.9).

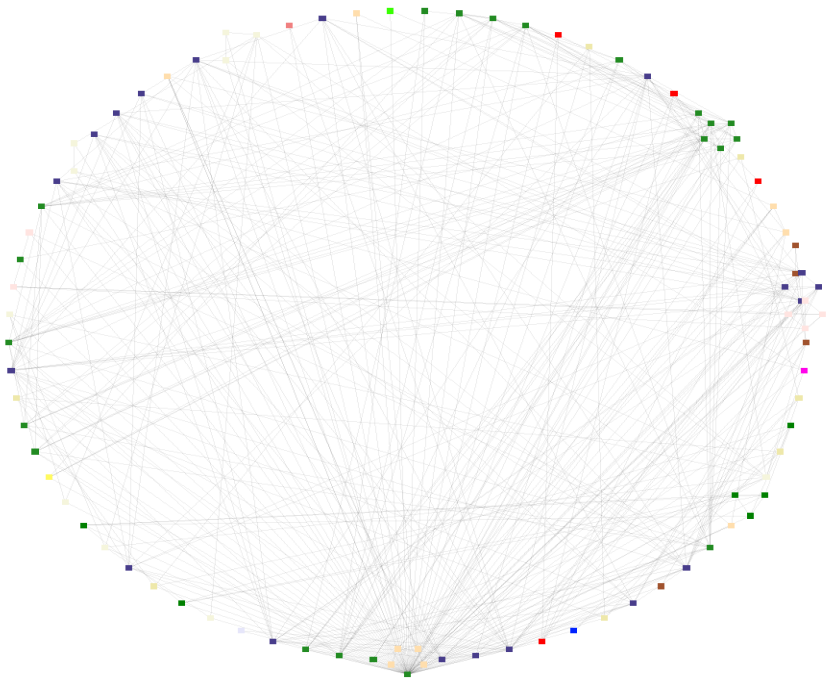

Рисунок 3.9 – Результат работы алгоритма с учетом реальных весов без масштабирования

[1] Принадлежит компании Meta, которая признана экстремистской и запрещена в Российской Федерации

Как видно из рисунка, большая часть вершин лежит в тривиальных сообществах. Это связано с относительно высокой степенью вершин при малых размерах графа. В данном случае для получения более содержательных сообществ можно искусственно ослаблять критерий прироста модулярности. Для этого воспользуемся введенной в предыдущем разделе параметризованной модулярностью (3.20). Уменьшая параметр α, получим различные по своей структуре наборы сообществ на графе. Так, при $\alpha = 0.05$ получаем разбиение, показанное на рисунке 3.10. Данный пример иллюстрирует то, что на практике при использовании модификации, учитывающей реальные степени вершин из исходной сети, целесообразно применять параметризованную модификацию модулярности.

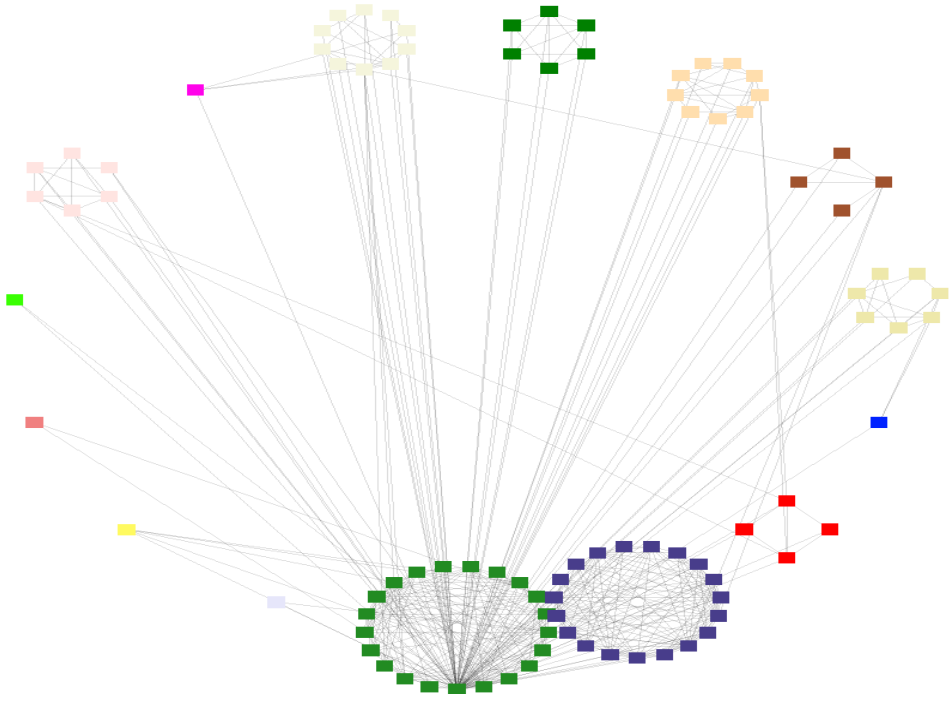

Рисунок 3.10 – Результат работы алгоритма с учетом реальных весов при $\alpha = 0.05$

При рассмотрении графа всей сети объединение вершины со смежными ей листьями в одно сообщество оправдано, ведь вершина-сосед является единственным объектом взаимодействия листовой вершины. В силу большого размера графа прибегают к анализу некоторых его подграфов. Обычным методом получения подграфа является поиск в ширину от заданной вершины. В то же время при рассмот-

рении подграфов сетей, в особенности социальных, достаточно часто возникает ситуация, когда некоторая вершина имеет множество соседей-листов. В результате, на первом уровне иерархии листовые вершины будут объединены в сообщество с этой вершиной. Отметим, что степень такого сообщества относительно мала, так как оно содержит в большинстве своем листовые вершины. Вследствие этого, на следующем уровне иерархии разбиения данное сообщество в силу предела разрешения чаще всего объединяется с другими сообществами или с вершиной, имеющей высокую степени. При рассмотрении графа пользователя сообщество образуется вокруг его вершины. Но наличие таких сообществ не отражает структуры всей сети: большая часть вершин будет связана лишь с вершиной пользователя.

3.6 Экспериментальные исследования

На основе описанной в главе 2 модели были построены графы взаимодействующих объектов, на основании данных, импортированных из социальной сети *ВКонтакте*. При построении каждого из этих графов во время импорта данных в качестве исходной вершины было взято по одному (разному) объекту исходной сети – пользователю. Фактором взаимодействия во время импорта данных было взято отношение дружбы. Получены следующие графы:

граф G_1, который состоит из 175 вершин и 1733 ребер;

граф G_2, который состоит из 155 вершин и 1335 ребер.

Чтобы явно проиллюстрировать описанные в главе 3 свойства алгоритма 3.2 на одном и том же графе применялся еще и алгоритм 3.1., основанный на случайном блуждании. Для наглядности результатов использовалась визуализация получаемых сообществ программным обеспечением, описанным в главе 7 данной работы.

В качестве иллюстрации свойств «сбора мусора» и объединения в одно сообщество маленьких групп при работе алгоритма 3.2 с каждым из графов G_1 и G_2 был проведен следующий эксперимент. Вначале применялся алгоритм 3.1, вершины

окрашивались в соответствии с полученными сообществами, после чего с сохранением раскраски вершин к исходному графу применялся алгоритм 3.2.

Как видно на рисунках 3.11 и 3.12 алгоритм 3.2 собрал в одно из сообществ все листы графа G_1, а также мелкие группы, которые выявил алгоритм, основанный на случайном блуждании. При этом, в силу переноса исходной вершины, большое сообщество разделилось на три примерно равных по размерам.

Аналогичная картина наблюдалась при работе алгоритмов с графом G_2 – на рисунках 3.13 и 3.14 видно, что алгоритм 3.2 собрал в одно сообщество с исходной вершиной все листы и малые группы. Дополнительная ценность данных двух экспериментов в том, что структуры графов G_1 и G_2 существенно различаются. У пользователя, соответствующего исходной вершине в графе G_2, имеются три примерно одинаковых по размерам группы общения, притом слабо связанных между собой. Тогда как у исходной вершины графа G_1 имеется одна большая группа общения, в которой можно выделить три разных подгруппы, что и сделано алгоритмом 3.2. При этом в обоих случаях алгоритм 3.2 «собрал мусор», что иллюстрирует описанное в текущей главе, в разделе 3.3, свойство.

Продемонстрируем как в данных примерах происходит выбор, к какому из уже сформированных сообществ добавить вершину. Суммарный вес ребер, инцидентных сообществу, наряду с весом вершины, для которой ведется подсчет возможного прироста модулярности, на очередном шаге алгоритма серьезно определяют формирование сообществ. Причем, большие значения этих параметров уменьшают возможный прирост модулярности. Это влечет за собой возможность присоединения вершин с большой степенью к сообществам, отличным от тех, с которыми у них имеется максимальное число общих связей.

Обратимся к графу G_1 и посмотрим на одну из вершин v_{73} под соответствующим условным номером. На графе G_1 выделены пять сообществ следующих размеров: $|S_1| = 45$, $|S_2| = 36$, $|S_3| = 36$, $|S_4| = 30$, $|S_5| = 28$. При этом вершина v_{73} отнесена к сообществу S_5. На рисунке 3.15 показан граф G_1 с выделенными на нем сообществами и вершиной v_{73}, перенесенной для наглядности из сообщества S_5 в центр рисунка.

Посмотрим на неравенство (3.26) и проверим, что действительно условие выбора сообщества S_5 для v_{73} выполнено. Вес вершины $k_{73} = 88$. Количество ребер этой вершины, смежных с сообществами равно соответственно: $k^1_{73,in} = 5$, $k^2_{73,in} = 29$, $k^3_{73,in} = 27$, $k^4_{73,in} = 1$, $k^5_{73,in} = 26$. Получаем, что выполнено:

$$k^2_{73,in} < k^5_{73,in} \tag{3.27}$$

Таким образом, видим, что вершина v_{73} не принадлежит в результате выполнения алгоритма тому сообществу, с которым имеет максимальное число связей.

Общий вес сообществ, выделенных на графе G_1 следующий: $\Sigma^1_{tot} = 494$, $\Sigma^2_{tot} = 1031$, $\Sigma^3_{tot} = 880$, $\Sigma^4_{tot} = 496$, $\Sigma^5_{tot} = 597$. Далее, учитывая (3.25), получаем таблицу 3.2. Откуда явно видно, что для всех остальных четырех сообществ выполнено неравенство (3.26), указывающее на принадлежность вершины v_{73} к сообществу S_5.

Таким образом, полученные в разделе 3.3 текущей главы свойства и рекомендации подтверждаются экспериментальными исследованиями. Учет описанных свойств получаемых при выделении сообществ на графах взаимодействующих объектов может быть использован в зависимости от исходных задач, стоящих при исследовании сети.

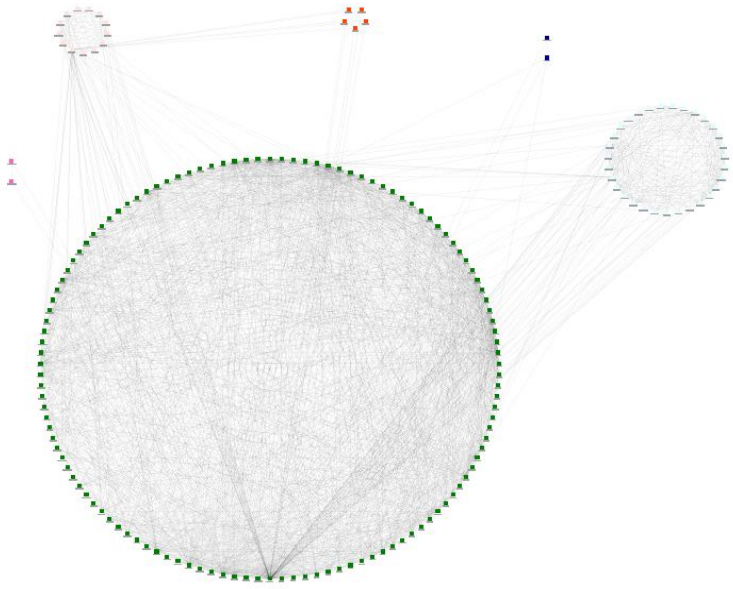

Рисунок 3.11 – Разбиение графа G_1 алгоритмом 3.1

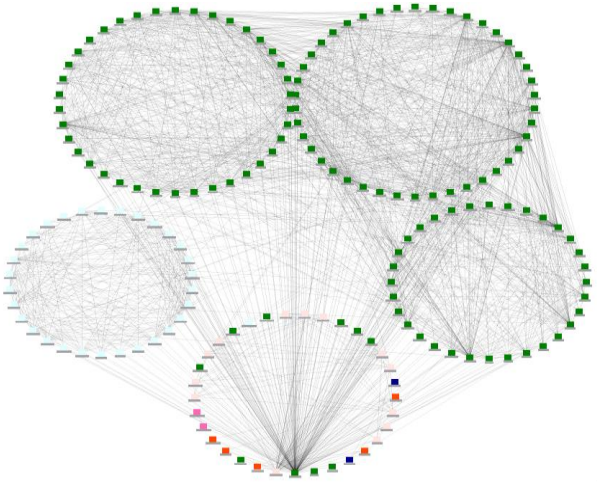

Рисунок 3.12 – Разбиение графа G_1 алгоритмом 3.2

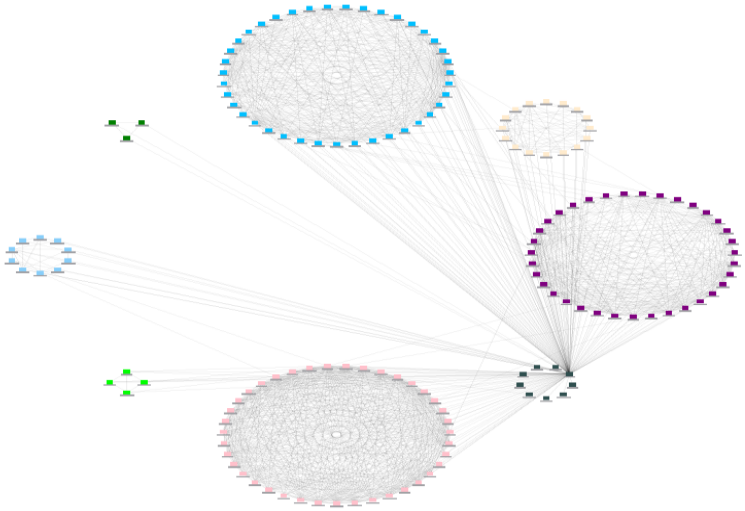

Рисунок 3.13 – Разбиение графа G_2 алгоритмом 3.1

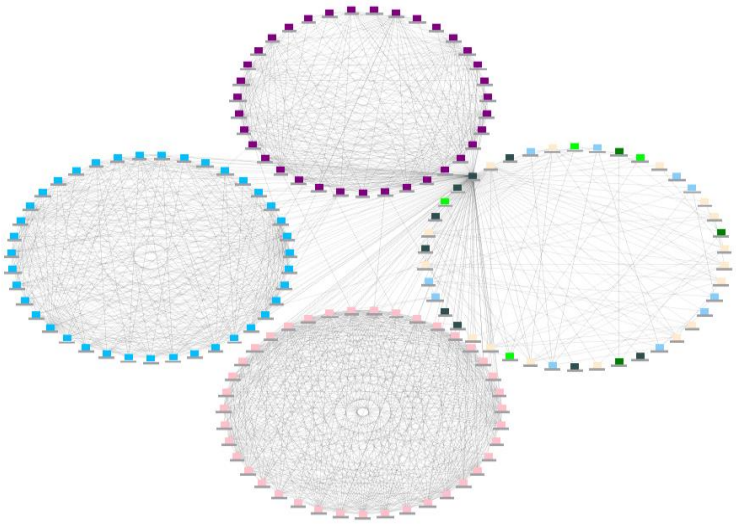

Рисунок 3.14 – Разбиение графа G_2 алгоритмом 3.2

Таблица 3.2 – Различия в приростах модулярности

$\frac{k_{73}}{2}(\lambda_1 - \lambda_5)$	$-2{,}6$	$k^1_{73,in} - k^5_{73,in}$	-21
$\frac{k_{73}}{2}(\lambda_2 - \lambda_5)$	11	$k^2_{73,in} - k^5_{73,in}$	3
$\frac{k_{73}}{2}(\lambda_3 - \lambda_5)$	7	$k^3_{73,in} - k^5_{73,in}$	1
$\frac{k_{73}}{2}(\lambda_4 - \lambda_5)$	$-2{,}5$	$k^4_{73,in} - k^5_{73,in}$	-25

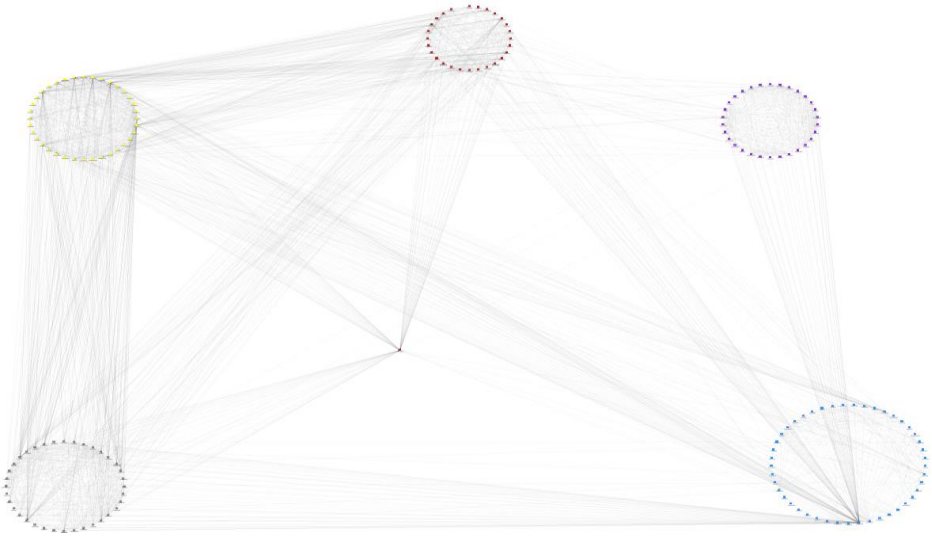

Рисунок 3.15 – Вершина v_{73} и ее связи с сообществами G_1

3.7 Комбинированный алгоритм

Перейдем теперь к вопросу выделения пересекающихся сообществ. Его возможно производить в несколько этапов, в том числе комбинируя с выделением непересекающихся сообществ. Эта идея лежит в основе Комбинированного алгоритма, описанного далее.

Построим алгоритм для выявления пересекающихся сообществ, основанный на последовательном применении алгоритмов *Louvain* [50] и *CPM* [45, 106]. Сам по

себе исходно алгоритм *Clique Percolation Method* (для краткости далее будем обозначать его *CPM*), для заданного наперед k ищет пересекающиеся сообщества посредством выделения полных подграфов размера k, называемых в литературе k-кликами. Для этого *CPM* необходимо изначально выделить все k-клики на заданном графе. Это является NP-трудной задачей, поэтому применение данного алгоритма напрямую имеет очевидные недостатки. Две k-клики называются смежными, если содержат $k - 1$ общую вершину. Сообществом на графе для *CPM* тогда можно назвать максимальный набор смежных k-клик.

На рисунке 3.16 представлен пример графа, на котором описанным выше образом выделены два сообщества. Это вершины синего и зеленого цветов, которые имеют пересечение, состоящее из двух вершин красного цвета. Обе эти вершины принадлежат каждому из выделенных сообществ. В данном случае взято значение $k = 4$ и сообщества найдены для этого размера клик согласно алгоритму *CPM*.

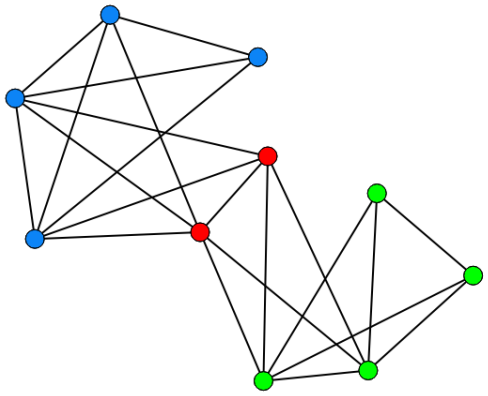

Рисунок 3.16 – Пример двух сообществ для $k = 4$

Комбинированный алгоритм построен так, чтобы вначале выделять на графе сообщества без пересечений между ними, а затем в случае выполнения определенных условий внутри этих сообществ находятся пересекающиеся сообщества меньшего размера с помощью алгоритма. Далее на третьем этапе в случае наличия после второго этапа оставшихся вершин и маленьких сообществ для них производятся действия согласно заданным изначально параметрам алгоритма. Подробнее опишем этапы работы Комбинированного алгоритма.

На первом этапе для графа G одним из классических алгоритмов ищется множество непересекающихся сообществ $S = \{S_0, S_1, S_2, \dots, S_r\}$, где $(r+1)$ – число сообществ, выделенных на этом этапе. Соответственно, на множестве S достигается локальный максимум Q_{max} для модулярности Q, определенной по формуле (3.3). В результате любая вершина графа G лежит в каком-то одном сообществе C_i, а каждое такое сообщество входит в S:

$$Q_{max} = \max_{S:\, C_i \in S\ \forall i} Q \tag{3.28}$$

Далее на втором этапе внутри каждого из сообществ $S_k \in S$, $k = 0, \dots, r$ рассматривается вопрос дополнительного выделения сообществ внутри S_k. Идея состоит в том, чтобы проверить следующее условие: часть вершин из S_k, имеющих высокую центральность по посредничеству, не превышает изначально установленного значения, которое определяется величиной $|S_k|$. Формально это условие записывается следующим образом:

$$\frac{\sum_{v \in S_k} \alpha(v)}{n_{S_k}} \le \beta(n_{S_k}), \quad \alpha(v) = \begin{cases} c_B(v), & c_B(v) \ge \tilde{\alpha} \\ 0, & c_B(v) < \tilde{\alpha} \end{cases} \tag{3.29}$$

где $n_{S_k} = |S_k|$; $c_B(v)$ – центральность по посредничеству вершины v;
$\tilde{\alpha}$ – пороговое значение, заданное исходно на полуинтервале $[0; 1)$;
$\beta(n_{S_k})$ – задан изначально на полуинтервале $(0; 1]$.

Здесь $c_B(v)$ для вершины $v \in S_k$ вычисляется классическим способом, но не по всему графу, а только внутри выделенного на первом этапе сообщества S_k:

$$c_B(v) = \sum_{\substack{s \ne t \\ s,t \in S_k \setminus \{v\}}} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{3.30}$$

где $v -$ вершина, для которой вычисляется $c_B(v)$; $\sigma_{st} -$ число кратчайших путей между s и t внутри S_k; $\sigma_{st}(v) -$ число кратчайших путей между s и t внутри S_k, содержащих v.

Ясно, что в силу $c_B(v) \le 1$ всегда будет выполнено $\sum_{v \in S_k} \alpha(v) \le n_{S_k}$.

Для тех S_k, для которых условие (3.29) выполнено, на втором этапе с помощью *CPM* ищется множество пересекающихся сообществ $S_k^{overlap} = \{S_k^0, S_k^1, S_k^2, \dots, S_k^{r_k}\}$,

где $(r_k + 1)$ – количество выделенных на этом шаге сообществ внутри S_k. На найденных множествах $S_k^{overlap}$ достигается локальный максимум модулярности:

$$Q_{S_k,max}^{overlap} = \max_{S_k^{overlap}} Q^{overlap} \quad (3.31)$$

После нахождения $S_k^{overlap}$ для S_k возможна ситуация, что какие-то из вершин, лежащих в S_k не вошли ни в одну из клик. Необходимо будет дополнительно определить их принадлежность к сообществам. Также возможна ситуация, когда число вершин в каких-то из сообществ, выделенных на первом этапе, менее размера клики. Для них также необходимо дополнительно определить порядок действий.

Для достижения этой цели на третьем этапе введем дополнительные параметры. Первый параметр i_{opt} определяет действия с вершинами, не вошедшими в новые сообществ на втором этапе. В таблице 3.3.а указаны возможные значения параметра i_{opt}.

Для определения действий с сообществами, полученными до второго этапа, но внутри которых число вершин менее размера выбранной клики, используется параметр m_{opt}. В таблице 3.3.б указаны возможные значения параметра m_{opt}.

В соответствии с выбранными при запуске алгоритма значениями для i_{opt} и m_{opt} формируется итоговый набор сообществ, выделенных на исходном графе. Третий этап позволяет предотвратить объединение многих листов графа или небольших сообществ в крупные лишенные смысла сообщества. Так данный метод предоставляет возможности для контроля в рамках получаемого разбиения эффекта «сбора мусора».

На рисунке 3.17 приведена возможная картина результата работы Комбинированного алгоритма внутри одного из сообществ, полученных на первом его этапе и обозначенного красным контуром. Зелеными контурами на рисунке 3.17 обозначены сообщества, выделяемые на втором и третьем этапах. Вершины, лежащие пересекающихся сообществах второго этапа, обозначены красным цветом. В данном случае используются: $k = 4$, $i_{opt} = i2c$, а также $m_{opt} = None$.

Выделение сообществ, аналогичное показанному, невозможно достичь при использовании отдельно алгоритмов *Louvain* и *CPM*. При этом, такого вида разбиения позволяют визуально анализировать отдельные фрагменты сети, что немаловажно при разных задачах операторов, работающих с подобными данными.

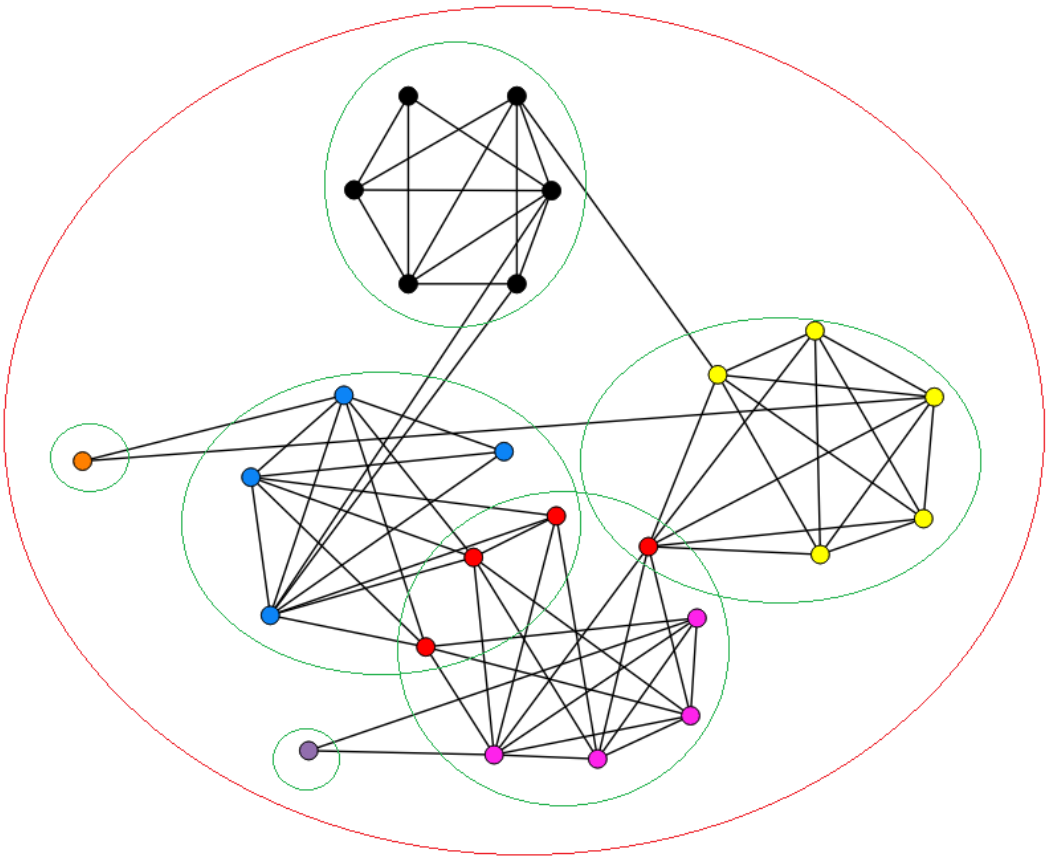

Рисунок 3.17 – Результат Комбинированного алгоритма

В контексте перехода между первым и вторым этапами необходимо подчеркнуть, что сравнительно высокое значение центральности по посредничеству указывает на вершины, которые могут служить связующими элементами между разными сообществами. Иными словами, удаление таких вершин из графа может привести к увеличению компонент связности. Неравенство (3.29) служит порогом для определения необходимости по дальнейшему разбиению выделенного на первом этапе сообщества на непересекающиеся с помощью *Louvain*. Ибо в случае, если велика доля вершин с высоким значением $c_B(v)$, то структура выделенного на первом этапе сообщества содержит части, требующие повторного разделения с помощью первого этапа Комбинированного алгоритма. Поэтому условие для перехода на

второй этап определяется долей вершин, у которых данное значение превосходит определённый порог $\tilde{\alpha}$.

Таблица 3.3.а – Возможные значения для i_{opt}

None	Отсеченные вершины не принадлежат ни одному сообществу
i2c	Каждая отсеченная вершина выделяется в индивидуальное тривиальное сообщество внутри того сообщества, в котором она находилась после первого этапа
i2r	Каждая отсеченная вершина выделяется в индивидуальное тривиальное сообщество, не имеющее пересечений с другими
ia2c	Множество из всех отсеченных вершин, ранее находящихся в одном сообществе после первого этапа, образует внутри него свое сообщество
ia2r	Все отсеченные вершины образуют единое сообщество, не имеющее пересечений с другими

Таблица 3.3.б – Возможные значения для m_{opt}

None	Структура сообществ не изменяется
mdel	Маленькие сообщества расформировываются, при этом, если вершина принадлежит другим сообществам, то она в них остается. В противном случае она будет принадлежать только своему тривиальному сообществу
m2c	Все маленькие сообщества, находящиеся в общем для них сообществе (то есть вложенные в него), объединяются в нем в одно общее сообщество, не имеющее вложенных сообществ
m2r	Все маленькие сообщества образуют одно общее сообщество, не имеющее пересечений с другими, даже если вершины принадлежали и другим сообществам

Разработанный алгоритм включает в себя как комбинацию двух классических алгоритмов (для выделения пересекающихся и не пересекающихся сообществ), так

и дополнительную параметризацию на этапе обработки результатов между их применением. Что позволяет убирать из рассмотрения малозначимые элементы сети (провести процедуру «сбора мусора») или выделять их в отдельные сообщества в зависимости от задач оператора.

Сформулируем итоговое представление Комбинированного алгоритма по основным его шагам. Алгоритм итерационно производит выделение сообществ на графе по описанным шагам и с учетом сформулированных выше принципов и в соответствии с заданными параметрами: размером клик k, показателями $\tilde{\alpha}$ и $\beta(n_{S_k})$ для формулы (3.29), значениями i_{opt} и m_{opt}.

Алгоритм 3.3.

Шаг 1. Выделение для G множества непересекающихся сообществ $S = \{S_k\}$.

Шаг 2. Для каждого S_k, выделенного на шаге 1 проверяется условие (3.29). В случае, если условие для S_k не выполнено, перейти к шагу 3. Если условие выполнено, перейти к шагу 4.

Шаг 3. Выделить на S_k непересекающиеся сообщества. Если остается S_k в исходном виде, то перейти к шагу 4. Если сообщества на S_k выделены, то дополнить множество S и перейти к шагу 2.

Шаг 4. Выделить на S_k множество пересекающихся сообществ $S_k^{overlap}$.

Шаг 5. Произвести действия с отдельными вершинами и малыми сообществами внутри полученных на прошлых шагах в соответствии с заданными изначально параметрами i_{opt} и m_{opt}.

3.8 Применение Комбинированного алгоритма

Для оценки получаемых Комбинированным алгоритмом результатов проведены вычислительные эксперименты его работы на наборе графов и сравнены с

рядом других алгоритмов. В том числе как с алгоритмами, выделяющими сообщества непересекающиеся: *Louvain*, *Girvan-Newman* [42], *Fast Greedy* [43], так и с выделяющими пересекающиеся: *CPM*, *CONGA* [107, 108].

Для проведения сравнений результатов их работы с результатами Комбинированного алгоритма использован индекс ω (*Omega Index*) [109, 111]. Индекс согласия ω позволяет сравнивать, насколько схожи два разбиения и может быть применен в том числе и для сравнения пересекающихся разбиений. Он основан на подсчете доли пар вершин, которые для двух заданных разбиений C_1 и C_2 либо оказались в общем сообществе, либо оказались в разных сообществах в обоих случаях. Обозначим за k_s количество сообществ, выделенных C_s – некоторым разбиением графа. Пусть всего у нас N пар вершин, которые могут быть распределены по сообществам.

Тогда ожидаемое согласие вычисляется следующим образом:

$$\omega_{expected}(C_1, C_2) = \frac{1}{N^2}\sum_{j=0}^{min(k_1,k_2)} N_j(C_1) \cdot N_j(C_2) \tag{3.32}$$

где $N_j(C_s) -$ количество пар вершин, которые ровно j раз были отнесены в одно и то же сообщество при разбиении C_s.

Наблюдаемое же согласие определяется соответственно:

$$\omega_{observed}(C_1, C_2) = \frac{1}{N}\sum_{j=0}^{min(k_1,k_2)} A_j \tag{3.33}$$

где $A_j -$ количество пар вершин, согласованных (встречающихся вместе или одновременно не встречающихся) в двух разбиениях ровно j раз.

Тогда ω задается формулой:

$$\omega(C_1, C_2) = \frac{\omega_{observed}(C_1,C_2) - \omega_{expected}(C_1,C_2)}{1 - \omega_{expected}(C_1,C_2)} \tag{3.34}$$

Значения индекса ω лежат на отрезке $[-1,\ 1]$, а равенство $\omega(C_1, C_2) = 1$ достигается только в случае абсолютного совпадения двух разбиений.

Для ряда стандартных алгоритмов и рассматриваемого Комбинированного метода на графах, импортированных из социальной сети *ВКонтакте* по модели, описанной в главе 2, были проведены вычислительные эксперименты. Сами графы приведены в таблице 3.4.

Таблица 3.4 – Графы, полученные при импорте из сети *ВКонтакте*

	Число вершин	Число ребер	Средняя степень вершин в графе
G_1	105	792	15,1
G_2	120	1226	20,4
G_3	158	1352	17,1
G_4	158	1596	20,2
G_5	187	1873	20,0
G_6	229	1726	15,1
G_7	439	4143	18,9
G_8	461	4299	18,7

Данные графы являются эгографами, у которых исходные степени вершин, с которых начиналось построение, находятся в диапазоне $(100;\ 500)$.

По итогам работы рассматриваемых алгоритмов были вычислены значения модулярности. Под Q^A в таблице 3.5 обозначено значение соответствующей модулярности для результата работы алгоритма A.

Как показано в таблице 3.5, число выделенных Комбинированным алгоритмом на данном наборе графов сообществ, как правило, больше, чем по результатам работы иных алгоритмов. При этом комбинированный алгоритм применялся со следующими параметрами: $i_{opt} = i2c, m_{opt} = m2r$ и $k = 4$, показавшими себя хорошо для социальной сети *ВКонтакте*.

Дополнительно качественный анализ результатов Комбинированного алгоритма, проведенный рядом непосредственных пользователей сети, по чьим аккаунтам были построены эгографы, показал содержательно эффективный результат именно этого метода в сравнении со стандартными алгоритмами.

Также посчитан ω индекс для сравнения результатов Комбинированного метода с остальными алгоритмами (таблица 3.6). В таблице приведены результаты вычисления ω индекса между Комбинированным и рассматриваемыми алгоритмами.

Таблица 3.5 – Результаты выделения сообществ на графах классическими алгоритмами и **КА**

	$k_{\text{КА}}$	$Q^{\text{КА}}$	$k_{Louvain}$	$Q^{Louvain}$	k_{CPM}	Q^{CPM}	k_{GN}	Q^{GN}	k_{FG}	Q^{FG}	k_{CONGA}	Q^{CONGA}
G_1	38	0,213	4	0,320	4	0,167	17	0,250	3	0,315	3	0,245
G_2	21	0,367	4	0,385	4	−0,013	20	0,329	4	0,371	4	0,136
G_3	28	0,650	4	0,614	5	0,390	15	0,581	5	0,581	5	0,494
G_4	39	0,327	4	0,342	3	0,016	47	0,175	3	0,313	4	0,100
G_5	36	0,048	5	0,369	3	−0,008	27	0,250	5	0,325	110	0,011
G_6	51	0,224	5	0,465	4	−0,029	22	0,408	6	0,393	4	0,273
G_7	90	0,501	7	0,572	8	−0,149	54	0,536	8	0,509	5	0,304
G_8	99	0,349	7	0,569	11	−0,272	53	0,514	6	0,552	8	0,195

Под k_A в таблице указано число сообществ, выделенных алгоритмом A.

За Q^A в таблице обозначено значение соответствующей модулярности для результата работы алгоритма A.

Таблица 3.6 – Подсчет ω для сравнения разбиений алгоритмами

	$\omega(Louvain)$	$\omega(CPM)$	$\omega(GN)$	$\omega(FG)$	$\omega(CONGA)$
G_1	0,0811	−0,0478	0,0209	0,0729	0,0226
G_2	−0,0125	0,0139	0,0152	−0,0087	0,0655
G_3	0,0160	−0,0166	−0,0017	0,0043	0,0149
G_4	0,0459	−0,0077	−0,0098	0,0047	−0,0147
G_5	0,0651	−0,0058	0,0232	0,0133	−0,0055
G_6	0,0915	0,0003	0,0004	0,0159	0,0088
G_7	0,0452	0,0009	0,0061	0,0020	0,0066
G_8	0,0152	0,0137	−0,0070	0,0096	0,0079

Необходимо отметить, что модулярность для результатов работы Комбинированного алгоритма не во всех случаях принимает значение больше, чем у других методов. Как показывают вычисленные значения ω для данных разбиений, имеется существенное отличие между найденными соответствующими алгоритмами сообществами. Это подтверждает, что ими найдены различные локальные максимумы модулярности.

Из таблицы 3.5 видно, что имеется отличие в числе выделенных сообществ. Результат работы Комбинированного алгоритма содержит большее их число, что ожидаемо следует из его строения. Классические алгоритмы не могут давать разбиения некоторых типов, например, показанное на рисунке 3.17. Этим объясняется и результат, показанный в таблице 3.6, а именно, крайне слабое согласование результатов между Комбинированным алгоритмом с одной стороны, и классическими алгоритмами – с другой. Значения ω около нуля показывают слабое согласование.

Совокупность выделяемых с помощью Комбинированного алгоритма сообществ содержит пересекающиеся и вложенные сообщества на исходном графе, что соответствует взаимодействию объектов в реальной природе, в том числе – пользователей в социальных сетях.

Рассмотрим подробнее работу Комбинированного алгоритма на примере графа G_3. На первом этапе алгоритм выделяет на графе G_3 множество из четырех сообществ $S = \{S_0, S_1, S_2, S_3\}$. Для которых выполнено: $|S_0| = 44$, $|S_1| = 41$, $|S_2| =$

37 , $|S_3| = 36$. Визуализация G_3 с выделенными на первом этапе сообществами представлена на рисунке 3.18. Для G_3 как эгографа хорошо видно, что сообщество S_0 сформировано путем «сбора мусора»: содержит вершину, с которой начато импортирование, а также листовые вершины. Еще три сообщества состоят из пользователей, редко связанных с другими сообществами.

Комбинированный алгоритм применяется со следующими значениями для параметров: $i_{opt} = i2c$, $m_{opt} = None$, $k = 4$. По результатам второго и третьего этапов Комбинированного алгоритма на графе выделяются 28 сообществ. Сообщество S_2 с очень высокой плотностью не претерпело изменений по их итогам. Для S_1 и S_3 были нашлись одна и три вершины соответственно, которые выделились в отдельные сообщества, вложенные в исходные S_1 и S_3.

А для сообщества S_0 выделено четыре вложенных в него (содержащих все как пересечение основную вершину эгографа). Помимо этого найдены 14 вершин, для каждой из которых выделено по своему сообществу в соответствии с $i_{opt} = i2c$.

На рисунке 3.19 представлен итог выделения сообществ Комбинированным алгоритмом. Сообщество S_0 показано красной окружностью, а вложенные в него – разными цветами, исходная вершина – помечена красным цветом.

По итогам исследования результатов работы Комбинированного алгоритма на графах из социальной сети *ВКонтакте* были получены следующие данные. По оценке пользователей, чьи эгографы рассматривались, разбиения, полученные классическими алгоритмами, содержательно были не столь показательны и актуальны, содержали большие сообщества, включающие совершенно разных людей. Хотя при этом показатель модулярности у таких разбиений и был иногда выше, чем у результата работы Комбинированного алгоритма.

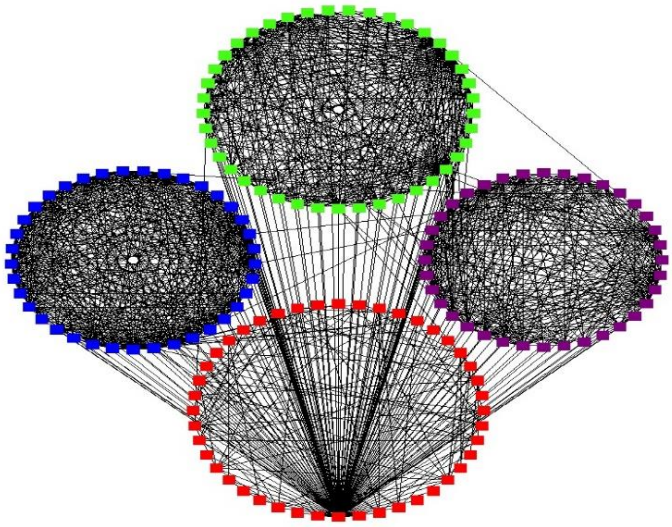

Рисунок 3.18 – Выделенные на графе G_3 четыре сообщества на первом этапе

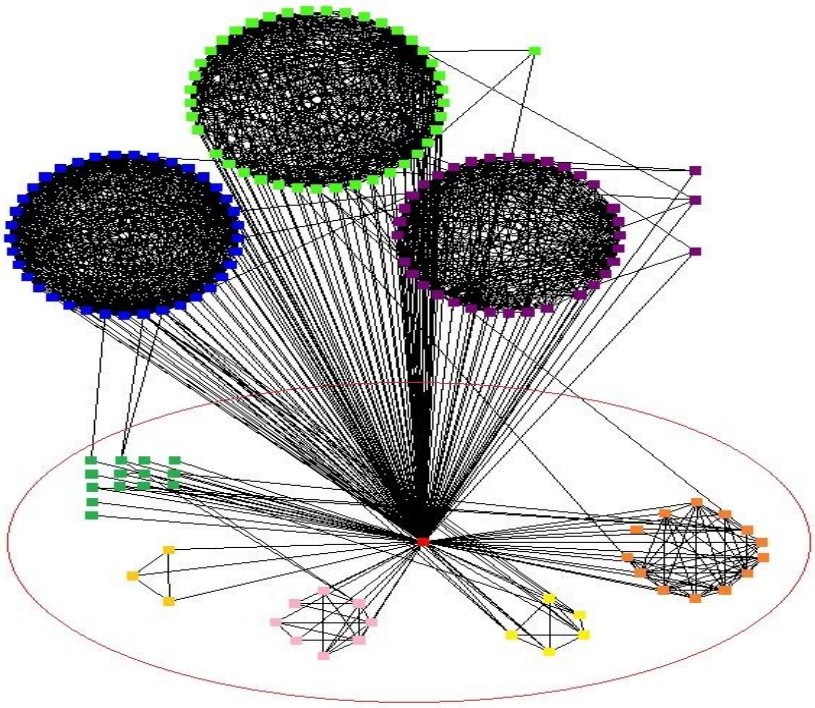

Рисунок 3.19 – Выделенные на графе G_3 итоговые сообщества алгоритмом

Помимо детально рассмотренных выше графов, полученных при импорте из сети *ВКонтакте*, были проведены дополнительные вычислительные эксперименты применения Комбинированного алгоритма на более чем 70 графах взаимодействующих объектов. Рассматривались графы, соответствующих различным сетям: биологическим, транспортным, социальным, сетям цитирования. Для сетей схожей природы получены наборы параметров, наиболее удачно подходящих для

соответствующего типа сетей. Визуализация выполнялась с помощью программном обеспечения, описанного в главе 6.

3.9 Сравнительный анализ элементов профилей пользователей сетей

В современном мире, где распространение информации через глобальную сеть и выявление групп влияния в социальных сетях имеют большое значение, описание участников коммуникации становится ключевым элементом. Такое описание, которое назовём профилем пользователя, представляет собой набор характеристик. Эти характеристики могут включать в себя данные, которые пользователь самостоятельно указал о себе (фактические характеристики), а также информацию, полученную на основе его действий в сети (поведенческие характеристики).

Профили пользователей широко применяются в различных сферах: для идентификации лиц, склонных к противоправным действиям; в банковской отрасли для оценки надёжности потенциальных заёмщиков и анализа возможных рисков для финансовых учреждений; при подборе персонала и оценке кандидатов на вакантные позиции; в маркетинге для создания портрета целевой аудитории товаров и услуг; в рекламе для демонстрации объявлений определённым группам пользователей с заданными характеристиками; злоумышленниками для получения несанкционированного доступа к аккаунтам; сетевыми сервисами для предоставления персональных рекомендаций и прогнозирования связей между пользователями.

В социальных сетях пользователи имеют возможность самостоятельно заполнять свой профиль информацией о себе. Однако часто эти профили остаются неполными. Причины отсутствия некоторых характеристик в профиле могут быть следующие: пользователь не указал данную информацию; пользователь скрыл информацию через настройки конфиденциальности; социальная сеть не предоставляет соответствующего поля для ввода этой характеристики. В связи с этим возникает задача восстановления недостающих характеристик для полного описания цифрового следа пользователя. Выбор подходов для решения этой задачи зависит

от доступных данных для анализа и от того, какие именно характеристики необходимо восстановить. Алгоритмы, показавшие высокую эффективность в предсказании одних характеристик, могут не подходить для предсказания других. Таким образом, выбор подходящего метода должен основываться на объёме и типе исходных данных, а также на том, какие характеристики требуются в итоге. В работе [95] была представлена методика такого анализа на основе выявления скрытых сообществ на графе социальных сетей.

После импорта данных, например, из социальной сети *ВКонтакте* и построения графа взаимодействующих объектов согласно описанной в главе 2 модели возможно построение базового профиля. Помимо данных, указанных пользователем о себе, импортируются данные о его дружеских связях с другими пользователями. Характеристики пользователей при этом учитываются как атрибуты вершин. В рамках этой задачи нас интересует эгограф – построенный от одной вершины анализируемого пользователя граф согласно методике из главы 2.

Первым этапом анализа профилей пользователей и групп общения является выделение на графе неявных сообществ. Для выделения сообществ может быть применен алгоритм 3.1, описанный в разделе 3.2 текущей главы.

Для выделения сообществ и визуального анализа применяется программное обеспечение, описанное в главе 7. Визуальный анализ сообществ в социальных графах позволяет лучше оценить плотность ребер внутри сообщества и характер связей между членами различных сообществ.

Круговое размещение сообществ оказывается удобным для дальнейшего анализа, оно позволяет лучше оценить плотность ребер внутри сообщества и характер связей между членами различных сообществ. На рисунках 3.20, 3.22, 3.24 представлены примеры разбиения реального графа друзей некоторого пользователя социальной сети *ВКонтакте* и его визуализация с выделенными сообществами.

После выделения сообществ определяются степени наполненности профилей и рассчитываются для какой доли пользователей доступен тот или иной атрибут. Это позволяет понять, какие атрибуты и в каком количестве присутствуют у пользователей, какие из них интересны для дальнейшего использования. Наибольшее

покрытие обычно наблюдается у таких характеристик, как пол, страна, город и дата рождения. На следующем шаге осуществляется подготовка выборки пользователей социальной сети и проводится их анализ. Опишем два примера экспериментов на случайных пользователях социальной сети *ВКонтакте*.

Граф $G_{\text{др1}}$, состоящий из 172 вершин и 3026 ребер, с разбиением на 7 значительных сообществ представлен на рисунке 3.20. Города проживания друзей позволяют сделать вывод о городах, в которых человек присутствовал на протяжении жизни. На столбчатой диаграмме можно зафиксировать преобладание конкретного города.

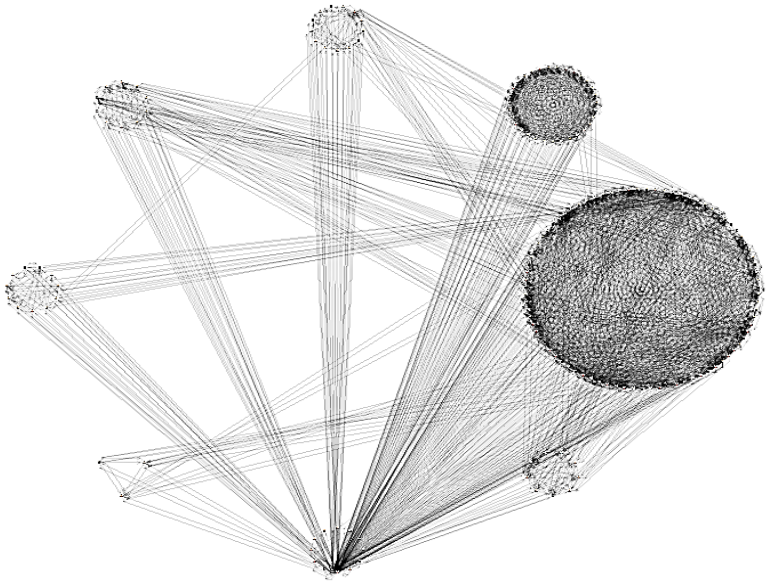

Рисунок 3.20 – Граф $G_{\text{др1}}$

Реже встречается преобладание двух городов, как на рисунке 3.21. Это происходит, как правило, если пользователь переезжал на протяжении жизни или, например, уезжал на обучение в другой город.

Другой граф друзей $G_{\text{др2}}$, в котором выделены 7 значительных (более трех вершин) сообществ, представлен на рисунке 3.22. Для этого выделения на рисунке 3.23 приведены гистограммы распределения по университетам по выделенным группам общения. Очевидно, что распределение по выделенным сообществам соответствует обучению в одном университете, что позволяет в случае отсутствия заполнения атрибута «университет» определить ожидаемую принадлежность к конкретному университету по принадлежности к сообществу.

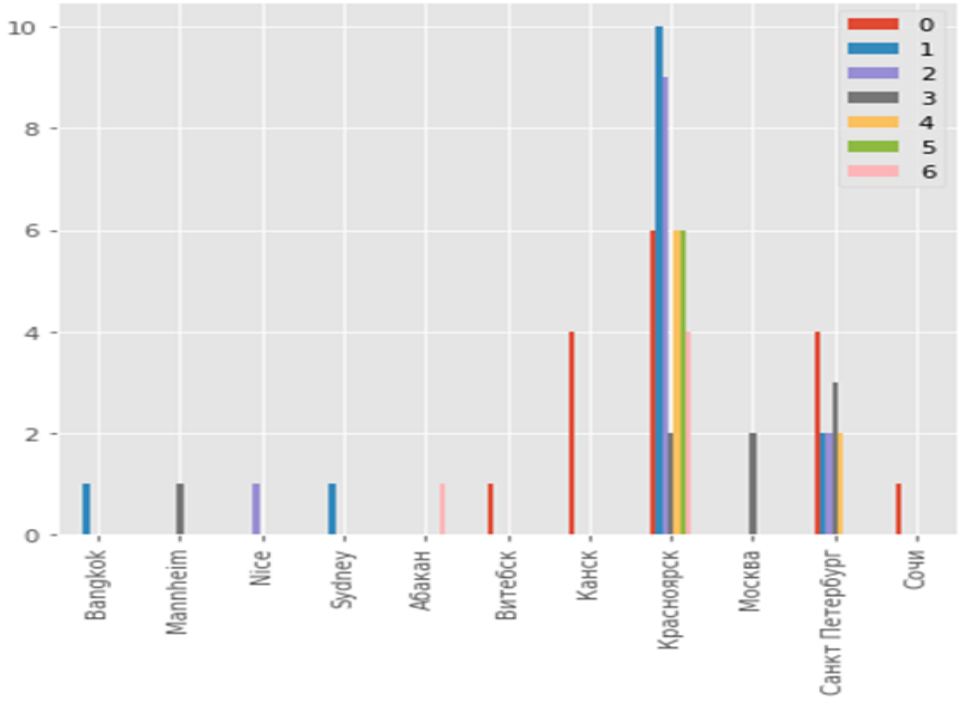

Рисунок 3.21 – Распределение по году рождения по выделенным сообществам на графе $G_{\text{др1}}$

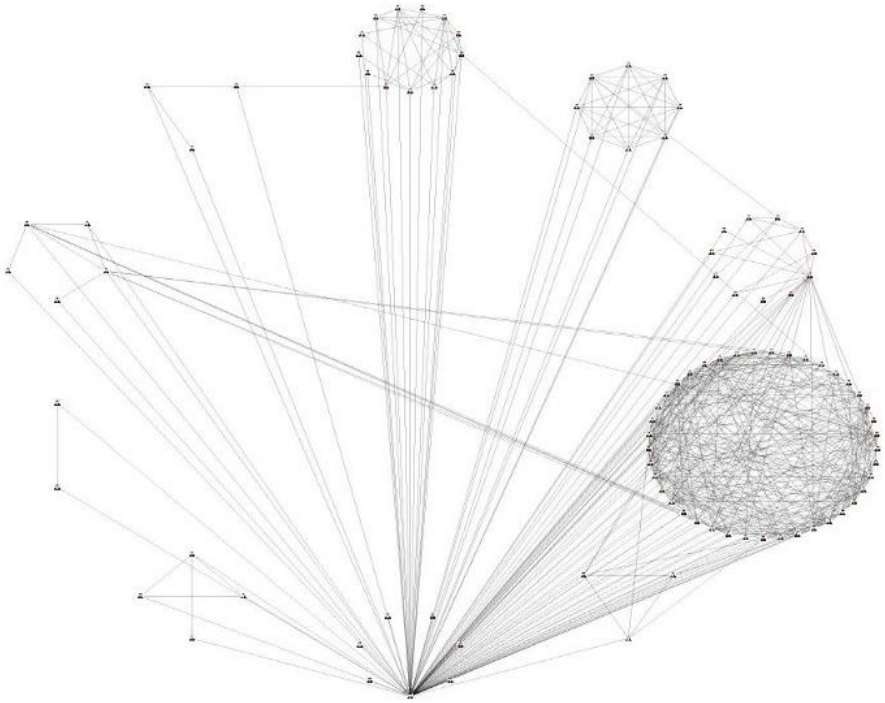

Рисунок 3.22 – Граф $G_{\text{др2}}$

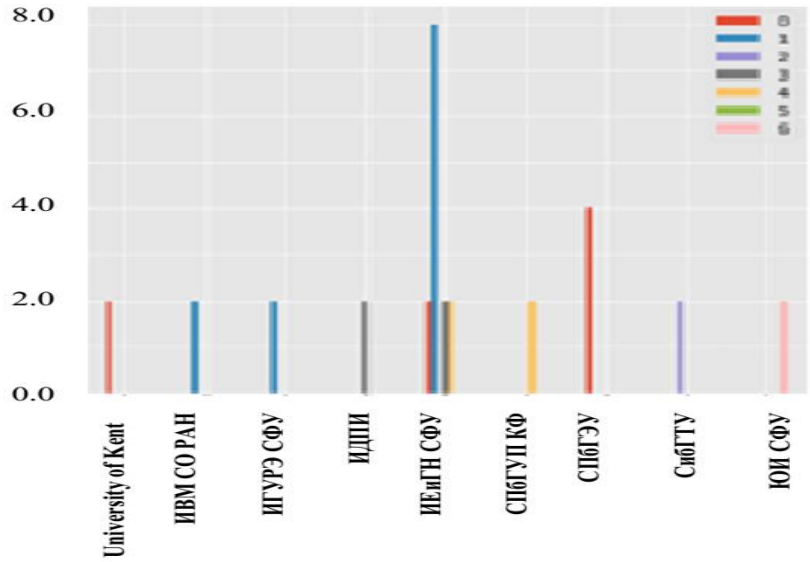

Рисунок 3.23 – Распределение по университетам по выделенным сообществам на графе $G_{\text{др2}}$

Рассмотрим граф $G_{\text{др3}}$, представленный на рисунке 3.24. Выделенная вершина соединена с 302 другими, которые распределены на 15 сообществ. При построении $G_{\text{др3}}$ представляет из себя эгограф, построенный на основании одного фактора взаимодействия – взаимной подписки («дружба» в данной сети). При этом у исходной вершины не известно значение атрибута возраста. Таким образом, решается задача восстановления этого значения с помощью выделенных сообществ. Простое усреднение значений по этому атрибуту у всех смежных вершин не дает разумного решения, ибо обычно имеется серьезный разброс в этих значениях и такой подсчет не обоснован.

После выделения сообществ и анализа полученных результатов с учетом значений других атрибутов можно классифицировать полученные группы пользователей. После чего получить более точную оценку для восстановления отсутствующего значения у исходной вершины.

Вычисленные значения стандартных величин статистического анализа для отдельных сообществ приведены на рисунке 3.25. Отсутствие большого разброса для года рождения показателен для сообществ, отражающих временную группу общения (друзей из школьного класса или университетской группы). На рисунке 3.25 такая ситуация наблюдается для следующих номеров: 0, 3, 12.

Наличие существенного разброса в значениях данного атрибута типично для сообществ, состоящих из людей, связанных родственными узами, либо рабочими отношениями. Для первого случая также может быть проверено совпадение значений для атрибута, отвечающего за фамилию пользователя (при наличии в сети такого). Описанная ситуация с существенным разбросом по параметру наблюдается на рисунке 3.25 для сообществ с номерами 8 и 9.

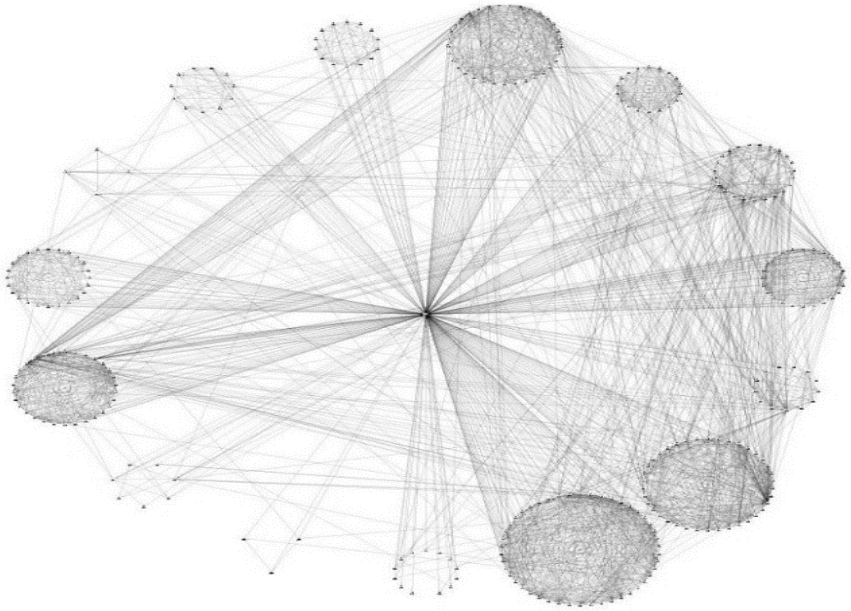

Рисунок 3.24 – Граф $G_{\text{дрз}}$

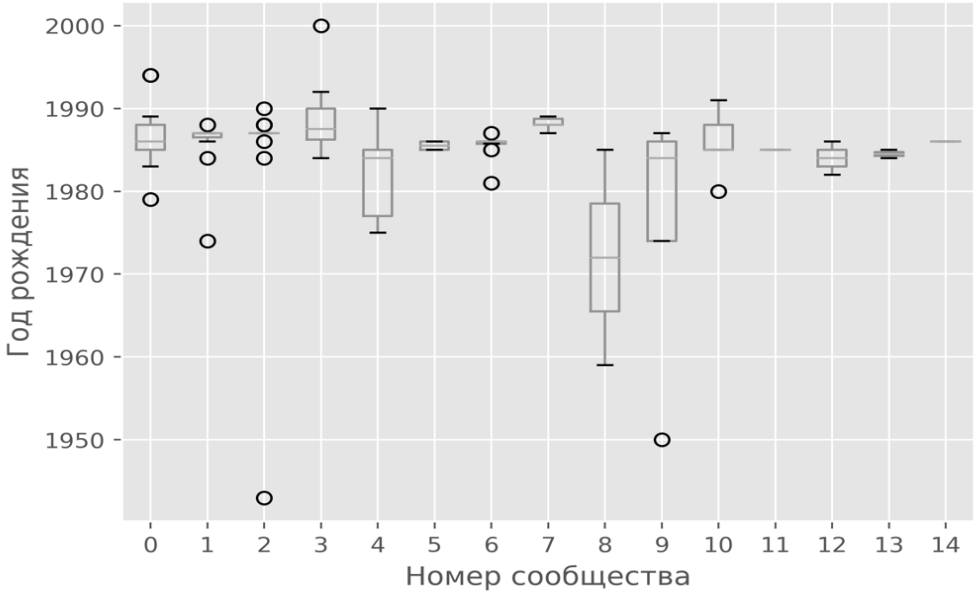

Рисунок 3.25 – Распределение внутри сообществ

Пользователям свойственно вступать в активную временную коммуникацию для получения информация по узкоспециализированной сфере вопросов, актуальных для них в течение некоторого фиксированного или ограниченного по продолжительности времени. Классическими примерами могут быть контакты во время учебы в вузе. Выделение сообществ и анализ таких неявных групп позволяет решать задачи изучения цифровых следов пользователей. Эти задачи и ручная обработка выделенных профилей может применяться в кадровой службе, банковском секторе, органами госбезопасности. Компоненты профилей могут быть использованы для автоматизированной обработки.

Описанная методика выделения основных характеристик пользователей актуальна для коммуникационных сетей и основана на выделении сообществ. Использование предложенного подхода применимо для задач восстановления составляющих профиля пользователя.

Приведенные примеры не только показывают возможности определения неуказанных значений атрибутов членов сообществ, но и показывают достоверность нахождения неявных сообществ алгоритмом 1.3.

3.10 Выводы по главе 3

1. Предложенный и реализованный Комбинированный алгоритм для выделения сообществ на графе взаимодействующих объектов является главным результатом главы и позволяет убирать из рассмотрения малозначимые элементы сети, предусматривает параметрические модификации для формирования разнородных разбиений в зависимости от задач оператора.

2. Представленные экспериментальные результаты работы Комбинированного алгоритма на данных, полученных из реальных сетей коммуникационного взаимодействия, показывают его применимость для решения актуальных задач по анализу соответствующих сетей.

3. Построенный итерационный алгоритм на основе классического алгоритма оценки энтропии сети и модифицированных весов показал высокие результаты как

на наборе случайно сгенерированных графов, так и на графах, полученных из реальных социальных сетей.

4. Представленные модификации классического агломеративного иерархического алгоритма позволяют получать результаты, востребованные на практике.

5. Продемонстрированная в данной главе методика выделения основных характеристик профилей пользователей социальных сетей на основе выявленных сообществ может быть использована для восстановления составляющих профиля пользователя.

6. Основные результаты, представленные в главе 3, опубликованы в работах [62], [69], [88], [89], [90], [91], [92], [93], [94], [95], [96]. В данных работах соискателю принадлежат построение Комбинированного алгоритма, методика его применения и анализ результатов, модификации классического алгоритма на основе оценки энтропии сети и модификации классического агломеративного иерархического алгоритма.

ГЛАВА 4 «МЕТОД ЯДРА» ВЫДЕЛЕНИЯ СООБЩЕСТВ

В данной главе представлен предложенный в работах [70, 73] алгоритм под названием «Метод ядра», дающий комплексный подход к анализу взвешенных графов. Вклад автора заключается в разработке метода и процедуры алгоритма, методики применения и анализу результатов применения.

Предлагаемый метод дает комплексный подход к анализу взвешенных графов. Представлен алгоритм действий аналитика в рамках метода для выделения лидеров мнений и анализа распространения информации в сети.

4.1 Обобщенная схема алгоритма

Под графом $G = G(V, \tilde{E})$ будем понимать взвешенный граф, построенный согласно модели, представленной в главе 2, при этом вес на множестве его ребер задается функцией w по описанным вариациям модели и соответствует степени интенсивности взаимодействия объектов между собой. В данной главе рассматривается выделение непересекающихся сообществ на графе G.

Вес $w(v)$ вершины v тогда определяется как сумма весов всех инцидентных ей ребер. А вес заданного множества вершин определим как сумму весов всех вершин этого множества. Соответственно, для множества выделенных на графе сообществ $S = \{S_i\}$ обозначим вес $w(S_i)$ каждого из них как сумму весов входящих в это сообщество вершин. Введем еще понятие внутреннего веса сообщества $w^*(S_i)$ – суммы весов ребер, обе вершины которых лежат внутри сообщества S_i. Аналогично внутренний вес вершины $w^*(v)$ определим как сумму весов инцидентных ей ребер, лежащих в сообществе данной вершины.

Если рассматривать граф G и применить к нему классические методы выделения непересекающихся сообществ, то картину, как правило, будет искажать большое число вершин-листов, полученных при импорте данных. О проблеме «сбора

мусора» уже было сказано в главе 3. Кроме этого, встречаются и вершины, у которых вес инцидентных им ребер существенно ниже остальных в графе. Как правило, такими будут вершины пользователей, осуществляющих минимальное взаимодействие с остальными. Для графов информационного взаимодействия данные пользователи не представляют особого интереса. Такие вершины так же отнесем условно к листам. В отличие от листов ключевыми являются вершины лидеров мнений, которые имеют большой вес. Выявление таких вершин как раз и является одной из важных задач, ибо вокруг них формируются «тяжелые» сообщества, происходит «притяжение» иных вершин.

Формально вершина v будет называться δ-мусором, если ее вес меньше δ. Тогда множество всего мусора в графе $Junk_\delta(G)$ определим следующим образом:

$$Junk_\delta(G) = \{v \in V \mid w(v) < \delta\} \quad (4.1)$$

Зеркально обратная картина имеет место для вершин с большим весом. Назовем α-звездой, или просто «звездой», такие вершины v графа G, что v имеет вес, больший некоторого заданного для этого графа значения α. Множество звезд $Star_\alpha(G)$ тогда определяется следующим образом:

$$Star_\alpha(G) = \{v \in V \mid w(v) > \alpha\} \quad (4.2)$$

Такие вершины притягивают к себе в сообщества другие вершины, если только те в свою очередь не состоят в сообществах с суммарным существенным весом. Одна из целей Метода ядра состоит в том, чтобы на графе G выделить ключевое множество – «ядро» (сообщество вершин или совокупность таких сообществ), имеющее существенный вес среди остальных сообществ и содержащее звезды.

Пусть на графе G выделено множество сообществ $S = \{S_i\}$. Максимальное среди всех $w(S_i)$ значение назовем весом ядра и обозначим через $w(Core(G))$. При этом значение $w(S_j)$ для некоторых j может быть близко к $w(Core(G))$. Поэтому для заданной степени близости γ определим γ-ядро $Core_\gamma(G)$, как множество вершин из сообществ, которые удовлетворяет следующему соотношению:

$$Core_\gamma(G) = \left\{ v \in V \,\middle|\, v \in S_i : \frac{w(S_i)}{w(Core(G))} > \gamma \right\} \quad (4.3)$$

Исходя из определения получаем $\gamma \in [0; 1)$, причем для выявления множества ключевых вершин графа требуется нахождение значения γ, близкого к 1.

Помимо ядер и звезд на графе часто присутствуют и сообщества меньших размеров, связь внутри которых плотна, а вес достаточно высок, что ядру и иным большим сообществам не удается поглотить данные меньшие сообщества по мере развития сети.

Упростить задачу анализа графа можно проигнорировав не самые активные взаимодействия исходных объектов за счет удаления «мусорных вершин». Сделать это можно двумя способами. Первый: просто удалив мусорные вершины, а затем оставшиеся от них ребра. Но мы воспользуемся другим способом: для всех ребер, имеющих вес менее заданного β будем считать этот вес равным нулю, то есть удалим такие ребра и получим новое множество ребер E'. После чего часть вершин станут изолированными и могут быть удалены из анализируемого графа. Получим V' – новое множество вершин. Обозначим получившийся после этих операций граф как $G'(V', E')$. Назовем его **графом активного информационного взаимодействия** объектов сети. Взяв β равным δ, мы сможем убрать не только $Junk_\delta(G)$**,** но другие малоактивные ребра и вершины из графа. Далее метод ядра применяется уже к $G'(V', E')$.

Определим коэффициент взаимодействия $k_{int}(G')$ как отношение удвоенного числа ребер к квадрату числа вершин в полученном графе G' активного информационного взаимодействия:

$$k_{int}(G') = \frac{2|E'|}{|V'|^2} \quad (4.4)$$

Коэффициент взаимодействия $k_{int}(G)$ аналогично может быть подсчитан и для исходного графа G. Для полного графа имеем $k_{int}(G') = \frac{2|E'|}{|V'|^2} = \frac{2\frac{n(n-1)}{2}}{n^2} = 1 - \frac{1}{n}$, что для больших n близко к 1. Т.е. для больших полных графов $\lim_{n\to\infty} k_{int}(G') = 1$.

В общем случае имеем $k_{int}(G') \in (0;1)$. Но для графов взаимодействующих объектов характерна разреженность. Поэтому высокие значения $k_{int}(G')$ будут свидетельствовать об активной коммуникации объектов соответствующей сети.

Далее введем $k_{S_i}(G')$ – коэффициент плотности сообщества S_i как отношение суммарного веса ребер внутри сообщества к суммарному весу ребер всего графа:

$$k_{S_i}(G') = \frac{w^*(S_i)}{\sum_{e \in E'} w(e)} \tag{4.5}$$

Аналогично определим $k_{Core_\gamma}(G')$ – коэффициент плотности γ-ядра, равный отношению суммарного веса вершин γ-ядра $Core_\gamma(G')$ к удвоенному суммарному весу ребер графа G':

$$k_{Core_\gamma}(G') = \frac{\sum_{v \in Core_\gamma(G')} w(v)}{2\sum_{e \in E'} w(e)} \tag{4.6}$$

Высокие значения $k_{Core_\gamma}(G')$ показывают, что ядро в таком графе играет существенную роль по сравнению с остальными сообществами и вершинами. На основе данных коэффициентов и будет строиться Метод ядра. Представим последовательность действий аналитика для решения основных задач распознавания лидеров мнений и выявления каналов распространения и обмена информации между пользователями. Именно эту методику и будем называть Методом ядра.

Алгоритм 4.1. Метода ядра для анализа взвешенных графов.

1. **Удалить изолированные вершины.** Наличие таковых может быть обусловлено особенностями исходной сети и процесса импорта данных. Получаем граф G.
2. **Подсчитать исходный коэффициент взаимодействия графа** $k_{int}(G)$. Он важен как ориентир для последующих шагов.
3. **Убрать мусор**. Необходимо определить значение β и выполнить операции по удалению ребер. Получаем граф G'. Если $G' = G$ перейти к шагу 5.
4. **Подсчитать обновленный коэффициент взаимодействия графа** $k_{int}(G')$. Рекомендуемый диапазон изменения для графов из социальных сетей лежит в

следующих пределах: $0{,}8 < \frac{k_{int}(G')}{k_{int}(G)} < 0{,}9$. В случае, если коэффициент изменился не в этом диапазоне, рекомендуется вернуться к шагу 3 и сделать его с иным значением β. Диапазон изменений может быть и иным, определяется опытным путем на массиве графов схожей природы.

5. **Применить алгоритм разбиения на сообщества**. Предполагается использование алгоритма, выделяющего непересекающиеся сообщества. Например, можно использовать вариации алгоритмов из предыдущих глав.
6. **Определить звезды.** Выбрать значение α и выделить множество вершин $Star_\alpha(G')$, состоящее из звезд графа. Проверить, что в наиболее крупных сообществах выделены звезды и такие сообщества имеют высокий показатель $k_{S_i}(G')$**.** Если нет, то поменять значение α.
7. **Выделить ядро.** Определить γ и составить $Core_\gamma(G')$.

Таким образом, для графа G' активного информационного взаимодействия после нахождения локального максимума модулярности Q_{max} и соответствующего ему множества непересекающихся сообществ Методом ядра находится значение параметра α, для которого $Star_\alpha(G)$ содержит вершины графа, осуществляющие основное информационное воздействие на остальных. После чего находится множество ключевых вершин графа $Core_\gamma(G')$:

$$\gamma \to \max_{Core_\gamma(G') \supset Star_\alpha(G')} \tag{4.7}$$

Данным алгоритмом в том числе выделяются вершины, осуществляющие наибольшее информационное воздействие на сеть, и наиболее активно взаимодействующие между собой вершины сети. Предложенная методика реализована в специальных программных средствах, реализующих как импорт данных из социальных сетей, так соответствующий инструментарий для анализа графов. Поддержка автоматического выделения сообществ за счет встроенных алгоритмов и визуализация полученного результата, позволяют реализовывать на практике Метод ядра.

В следующих разделах данной главы рассмотрены конкретные примеры реализации метода на графах, построенных при импорте данных согласно вариации модели (2.3).

При проведении анализа графов взаимодействующих объектов возможно рассмотреть мета-граф, вершинами которого являются выделенные на исходном графе сообщества. Ребра в таком мета-графе строятся на основании суммарных весов ребер между парами вершин из разных сообществ, ставших мета-вершинами. Для первой итерации, после первого выделения сообществ на графе, будем для конкретного графа G_s, построенного на основе реальных данных при импорте из s-ой сети, обозначать такой мета-граф как $G_{s,1}$. Тогда с каждой итерацией число вершин будет уменьшаться: каждая мета-вершина в новом графе представляет из себя группу вершин предыдущего графа. Следовательно, мета-вершина при этом сама является некоторым графом, внутри которого полезно применить алгоритм выделения сообществ и проанализировать результат.

После выделения в мета-графе сообществ и анализа их связей между собой, получаем новые мета-вершины. Повторив операцию формирования мета-вершин, получаем граф $G_{s,2}$. Аналогично на i-ом шаге получается граф $G_{s,i}$.

4.2 Примеры использования «Метода ядра»

Рассмотрим следующий пример. Импортированы данные из сети *Twitter* – стартовые 8 популярных постов, тема которых связана с распространением в 2020 году *Covid-19* и мерами, принимаемыми в связи с этим правительством (данные были взяты на дату 27.05.2020). Скачаны также оставленные пользователями под этими постами комментарии, отметки «лайк», поставленные в лентах пользователей «ретвиты». Посты относятся как к официальным аккаунтам властей, так и провокационные сообщения оппозиционных аккаунтов. Импорт данных и построение графа взаимодействующих объектов происходили в соответствии с (F, L, C, R)-моделью (2.3).

Вначале был получен граф с 632 вершинами и 1002 ребрами. Согласно описанному ранее алгоритму Метода ядра произведены следующие действия. Во-первых, была произведена предобработка данных, удалены изолированные вершины и осталось 459 вершин при 1002 ребрах, так получен граф $G_{covid} = G(V, \tilde{E})$. Средний вес ребер и средний вес вершин у полученного графа G_{covid} равны соответственно 2,767 и 1,585, а $k_{int}(G_{covid}) = 0{,}0095$. После переходим к шагу 3 Метода ядра и «убираем мусор»: выбрав значение $\beta = 1$, уберем 249 ребер с весом не превосходящим β и 34 вершины ставшие изолированными после удаления этих ребер. Получаем граф G'_{covid}, для которого коэффициент $k_{int}(G'_{covid}) = 0{,}0083$. Изменения можно оценить следующим образом: $\frac{k_{int}(G'_{covid})}{k_{int}(G_{covid})} = 0{,}87$, что лежит в рекомендованном для *Twitter* диапазоне. Посчитаем суммарный вес ребер: $\sum_{e \epsilon E'} w(e) = 2773$.

Далее выделим на графе G'_{covid} неявные сообщества. Получаем 43 сообщества (рисунок 4.1) из которых восемь состоят из досточно большого числа вершин (таблица 4.1). Значения $k_{S_i}(G')$ и $\max\limits_{v \epsilon S_i} w^*(v)$ принимают относительно высокие по графу значения для $i = 0, 1, 2, 4$. Из чего можно сделать вывод об активном взаимодействии внутри S_i для этих значений i, а также присутствии в них звезд. В сообществах S_5 и S_7 возможно наличие звезд, а сообщества S_3 и S_6 скорее не имеют таковых в своем составе. Но при этом плотность сообщества S_3 – высокая.

Посмотрим на вершины графа G'_{covid} с максимальными весами (таблица 4.2). Средний вес вершины в G'_{covid} равен 12,08. Последний столбец, полученный как вес вершины, деленный на это значение, хорошо показывает звезды – вершины с данным показателем выше 14. Поэтому возьмем $\alpha = 170 > 12{,}08 \cdot 14 = 169{,}12$.

Таким образом, мы нашли $Star_\alpha(G'_{covid})$, и для $\alpha = 170$ это множество состоит из 5 вершин. Сообщество S_0 имеет наибольший вес, поэтому берем $Core(G'_{covid}) = S_0$, и определим $\gamma = 0{,}77$. Тогда в $Core_\gamma(G'_{covid})$ входят S_0, S_1, S_2 и S_4. Посчитаем $k_{Core_\gamma}(G'_{covid}) = \frac{2195}{2 \times 2773} = 0{,}404$. Полученное высокое значение свидетельствует

о правильно найденном ядре. Далее можно посмотреть на строение внутри ключевых сообществ, в том числе входящих в $Core_{\gamma}(G'_{covid})$.

Рассмотрим разбиение на внутренние сообщества в S_1 (рисунок 4.2). Эта мета-вершина состоит из источника информации – вершины-звезды, соответствующей официальному аккаунту СМИ, а также смежных ей вершин. Будем называть такие мета-вершины «созвездиями первого рода», а «планетами» – вершины, смежные вершине-звезде. Таким образом, S_1 является созвездием первого рода, состоящем из одной вершины-звезды, и 48 вершин-планет.

Рассмотрим подробнее $S_0 \in Core_{\gamma}(G'_{covid})$. Выделим в S_0 внутренние сообщества (рисунок 4.3). Визуально видна вершина-звезда – это ярко выраженный лидер мнений, а также смежные с ним пользователи, имевшие взаимодействие с исходными постами. Стоит отметить, что состав многих из этих пользователей неоднозначен: никнеймы базовые, из 15 случайных символов, создаются при регистрации, количество подписчиков нулевое или близко к нулю, фотографии загружены в большинстве своем без лица. Предположительно, такие пользователи являются ботами или фейками соответствующего лидера мнений. Безусловно, тут присутствуют и реальные пользователи, разделяющие взгляды лидера. Они даже могут образовывать свои сообщества, в данном случае их два, но оба они из 2 вершин. Далее будут примеры, где дополнительные сообщества больше.

Будем называть такие мета-вершины «созвездиями второго рода», «звездами» в них – лидеров мнений, а «планетами» – остальные вершины (в общем случае не все из них будут смежными со звездой). Таким образом, в нашей классификации S_0 является созвездием второго рода, состоящем из одной вершины-звезды, с которой связаны 43 вершины-планеты.

Сообщества S_2 и S_4 так же представляют из себя созвездия второго рода (рисунок 4.4 и рисунок 4.5), состоящие из одной звезды, 32 и 43 вершин-планет соответственно. Другие вершины согласно значению α тут звездами не являются.

Сообщество S_3, которое не входит в γ-ядро $Core_{\gamma}(G'_{covid})$, представляет собой «созвездие третьего рода» (рисунок 4.6), звезды здесь нет, но плотность $k_{S_3}(G'_{covid})$ достаточно высокая, вершины еще соревнуются за главенство в этой группе. Это

означает, что при дальнейшем развитии сети с течением времени звезда тут вероятно появится. Устройство сообществ S_5, S_6 и S_7 аналогично одной из трех рассмотренных ранее ситуаций (рисунки 4.7–4.9).

Таким образом, с помощью Метода ядра выделяются лидеры мнений и сообщества, составляющие группы взаимодействия для соответствующих лидеров. Причем необходимо отметить, что при качественном анализе выделены оказались лидеры различных мнений – как «проправительственных», так и показательно «оппозиционных».

Рассмотрим еще один пример. Импортированы данные из сети *Twitter – 7* постов на тему вакцинации от *Covid-19* с помощью *Sputnik V* (на дату 03.02.2021). Посты аккаунта @sputnikvaccine (официальный аккаунт вакцины) и одного активного аккаунта пользователя из ФРГ (*более 21 тыс. подписчиков*). Импорт данных произведен в той же модели: на основе взаимодействия с исходными 7 постами и ранее совершенных действий пользователей генерируется взвешенный граф. Вес на ребре получен как взвешенная сумма, в которой учтены подписки пользователей, их лайки друг другу, комментарии и ретвиты постов друг друга.

При скачивании получился граф с 524 вершинами и 265 ребрами. Он крайне разрежен, даже после удаления изолированных вершин получен граф G_{SptV}, в котором 214 вершин и 265 ребер. Средний вес ребер и средний вес вершин у полученного графа G_{SptV} равны соответственно 1,238 и 2,679, а $k_{int}(G_{SptV}) = 0{,}01157$.

На шаге 3 Метода ядра выберем значение $\beta = 0$, поэтому граф $G'_{SptV} = G_{SptV}$ и $k_{int}(G'_{SptV}) = k_{int}(G_{SptV}) = 0{,}01157$. В данном случае шаг 4 не предполагается. При этом суммарный вес ребер получается следующим: $\sum_{e \epsilon E'} w(e) = 710$.

На графе выделяются 20 сообществ (рисунок 4.11), из них 5 содержат более 3 вершин, детали представлены в Таблице 4.3. Вес первого сообщества выше остальных, следовательно $Core(G'_{SptV}) = S_0$. Коэффициент плотности $k_{S_i}(G'_{SptV})$ и вес $w(S_i)$ существенно меньше у S_3 и S_4. Рисунок 4.11 иллюстрирует то же самое. Для значения $\gamma = 0{,}3$ получим, что $Core_\gamma(G'_{SptV})$ будет состоять из первых трех сообществ: $\frac{w(S_i)}{w(Core(G))} > 0{,}3$ для $i = 0, 1, 2$. При этом высокие значения коэффициента

$\frac{\max_{v \in S_i} w^*(v)}{w(S_i)}$ для $i = 3, 4$ указывают на возможное наличие звезд в этих сообществах. Поэтому необходимо проверить данные сообщества.

Для выделения звезд сначала посмотрим на вершины графа G'_{SptV} с максимальными весами (таблица 4.4). В этом графе значение среднего веса вершины равно 6,63. Последний столбец, полученный как вес вершины, деленный на это значение, указывает на возможные звезды в этом графе. Для графа G'_{SptV} видны два возможных варианта – вершины с данным показателем выше 3,7 или 5 соответственно. Поэтому чтобы определиться со звездами и значением α, нам необходимо на шаге 6 метода исследовать этот вопрос. Для этого перейдем к мета-вершинам и рассмотрим разбиение на сообщества внутри них.

Для определения значения α посмотрим на внутреннее строение сообществ как отдельных графов. Начнем с графа, соответствующего S_0 и состоящего из 90 вершин. После выделения сообществ из S_0 алгоритма получаем 14 сообществ (рисунок 4.11). Посмотрим на те 4 вершины, что претендуют на лидерство (номера 2, 6, 9, 10). Они изображены на рисунках 4.12 - 4.15. Картинки расположены по убыванию степени взвешенной внутренней вершины в S_0.

И если по вершинам Ru***or и so***12 сомнений не возникает, то роль bo***nc и Er***ja не так очевидна. Исследуем роль вершины bo***nc подробнее. Для этого возьмем сообщество $S_{0,0}$ внутри которого она находится в подграфе S_0. Выделим сообщества в $S_{0,0}$ получаем разбиение на три сообщества, где Ru***or и bo***nc играют роли лидеров (рисунки 4.16, 4.17).

Посмотрим на строение сообществ S_1, S_2 и S_3. Заметим, что сообщества S_1 и S_2 являются созвездиями первого рода, а S_3 – третьего рода, там нет звезды. Подробнее устройство сообществ S_1, S_2, S_3 рассмотрено на рисунках 4.18–4.20. Мы проверили, что вершины с номерами 7 и 8 не являются звездами. Поэтому возьмем $\alpha = 6{,}6 \cdot 5 = 33$. Множество $Star_\alpha(G'_{SptV})$ для $\alpha = 33$ состоит из 6 вершин. Тогда согласно (4.3) получаем, что в $Core_\gamma(G'_{SptV})$ входят S_0, S_1, S_2, а $k_{Core_\gamma}(G'_{SptV}) = \frac{365}{2 \times 710} = 0{,}257$.

Таблица 4.1 – Основные показатели сообществ G'_{covid}

S_i	$\lvert S_i \rvert$	$w(S_i)$	$w^*(S_i)$	$k_{S_i}(G')$	$\max\limits_{v \in S_i} w(v)$	$\max\limits_{v \in S_i} w^*(v)$	$\frac{\max\limits_{v \in S_i} w^*(v)}{w^*(S_i)}$
S_0	44	710	382	0,068	445	187	0,489
S_1	49	576	408	0,073	315	204	0,5
S_2	33	537	296	0,053	258	129	0,435
S_3	25	444	288	0,051	60	30	0,104
S_4	44	372	306	0,055	171	128	0,418
S_5	24	340	170	0,03	171	83	0,488
S_6	24	337	180	0,032	64	27	0,15
S_7	22	167	134	0,024	84	61	0,455

Таблица 4.2 – Значения весов вершин графа G'_{covid}

Зашифрованное имя пользователя	Количество вершин, смежных вершиной v	$w(v)$	$w^*(v)$	Усредненный вес вершины
v_***ov	126	445	187	37,08
le***al	84	315	204	26,25
Vi***va	76	258	129	21,5
Mr***ay	65	171	128	14,25
Pr***at	62	171	83	14,25
Ol***13	27	84	61	7
aa***an	38	64	27	5,33
8a***Wn	20	60	30	5
ma***n_	25	57	29	4,75
Se***us	20	49	23	4,08
ru***60	31	42	15	3,5
NпA***36	13	41	18	3,41
dj***ef	12	40	15	3,33

Таблица 4.3 – Основные показатели сообществ G'_{SptV}

S_i	$\lvert S_i \rvert$	$w(S_i)$	$w^*(S_i)$	$k_{S_i}(G')$	$\max\limits_{v \in S_i} w(v)$	$\max\limits_{v \in S_i} w^*(v)$	$\dfrac{\max\limits_{v \in S_i} w^*(v)}{w^*(S_i)}$
S_0	90	625	618	0,435	106	106	0,171
S_1	47	402	402	0,283	115	115	0,286
S_2	33	198	192	0,135	35	35	0,182
S_3	7	78	78	0,054	27	27	0,346
S_4	4	43	40	0,028	16	15	0,375

Таблица 4.4 – Значения весов вершин графа G'_{SptV}

Условный номер вершины	S_i	Зашифрованное имя пользователя	Количество вершин, смежных вершиной v	$w(v)$	$w^*(v)$	Усредненный вес вершины
1	S_1	Ca***ha	26	115	115	17,35
2	S_0	Ru***or	28	106	106	15,99
3	S_1	Vo***co	15	40	40	6,03
4	S_2	pa***r1	13	35	35	5,28
5	S_1	Ca***os	13	35	35	5,28
6	S_0	so***12	10	34	34	5,13
7	S_3	vi***ur	6	27	27	4,07
8	S_2	Ci***K1	11	27	25	4,07
9	S_0	bo***nc	16	25	24	3,77
10	S_0	Er***ja	7	25	25	3,77
11	S_0	Pi***e7	7	21	21	3,17
12	S_3	na***er	4	19	19	2,87
13	S_0	AS***ux	5	17	17	2,56

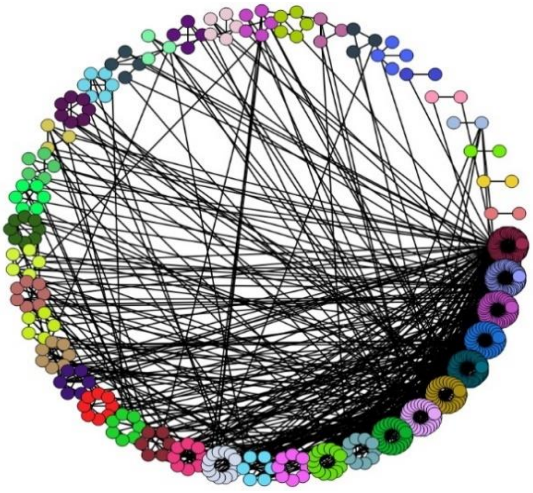

Рисунок 4.1 – Визуализация неявных сообществ на графе G'_{covid}

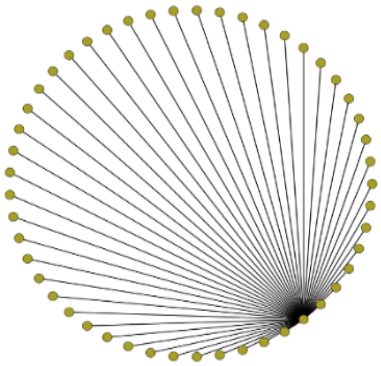

Рисунок 4.2 – Внутренне устройство сообщества S_1

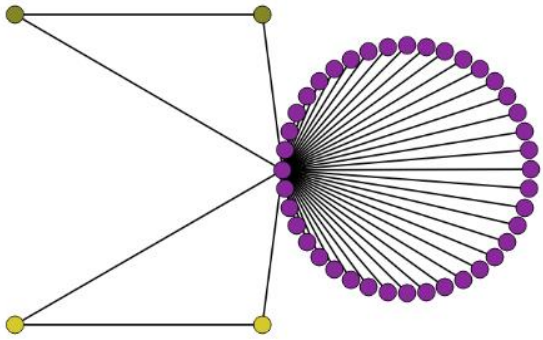

Рисунок 4.3 – Внутреннее устройство сообщества S_0

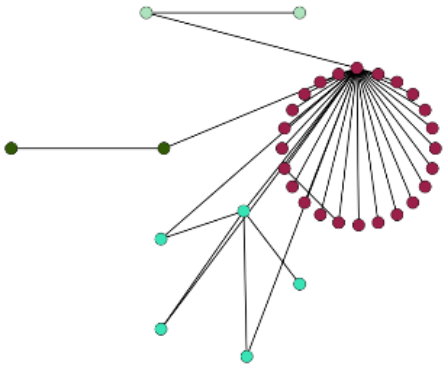

Рисунок 4.4 – Внутреннее устройство сообщества S_2

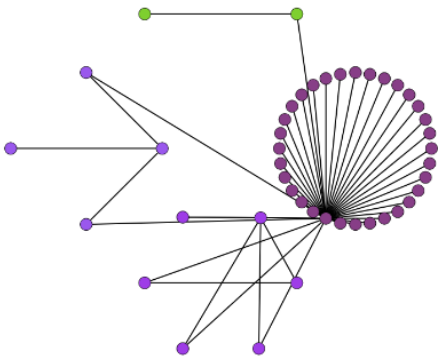

Рисунок 4.5 – Внутреннее устройство сообщества S_4

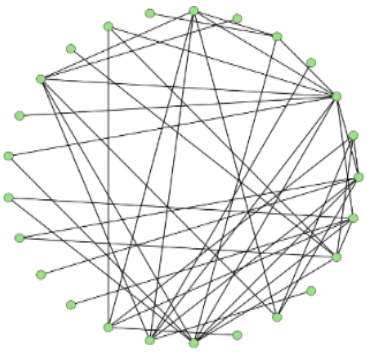

Рисунок 4.6 – Внутреннее устройство сообщества S_3

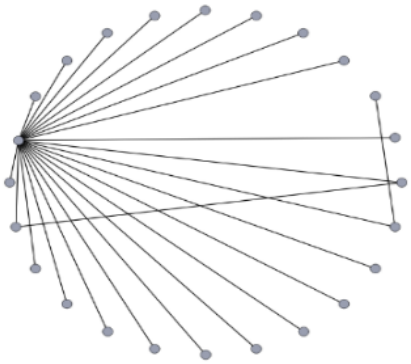

Рисунок 4.7 – Внутреннее устройство сообщества S_5

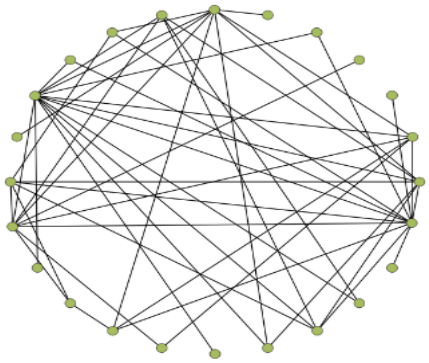

Рисунок 4.8 – Внутреннее устройство сообщества S_6

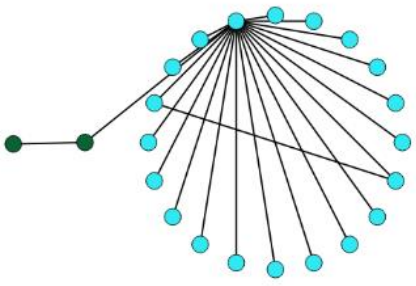

Рисунок 4.9 – Внутреннее устройство сообщества S_7

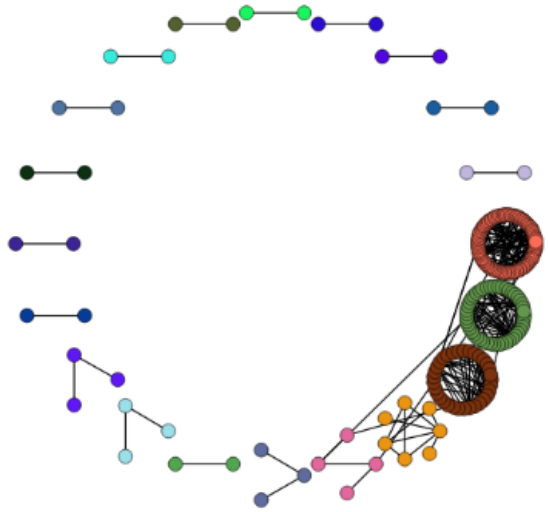

Рисунок 4.10 – Сообщества графа G'_{SptV}

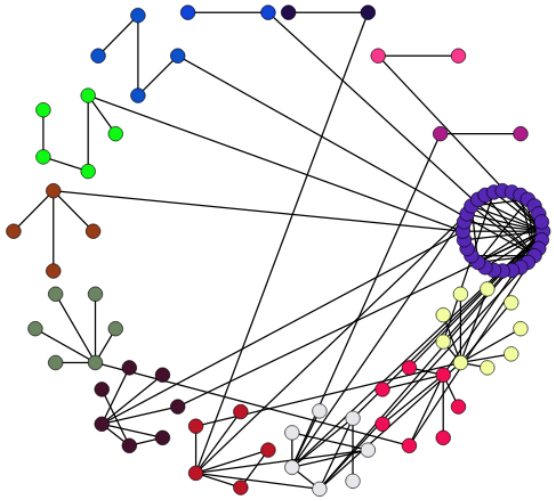

Рисунок 4.11 – Внутреннее устройство сообщества S_0 графа G'_{SptV}

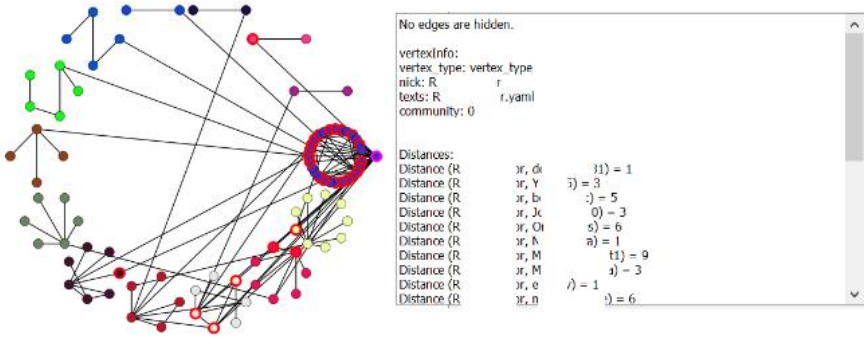

Рисунок 4.12 – Сообщества подграфа S_0, связи вершины Ru***or

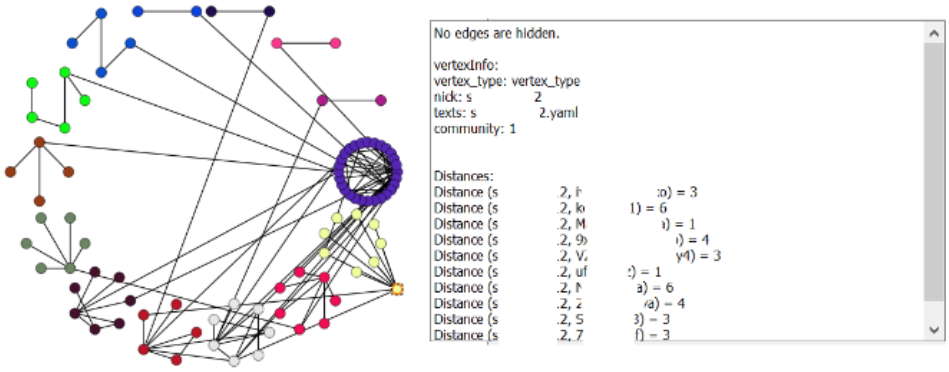

Рисунок 4.13 – Сообщества подграфа S_0, связи вершины so***12

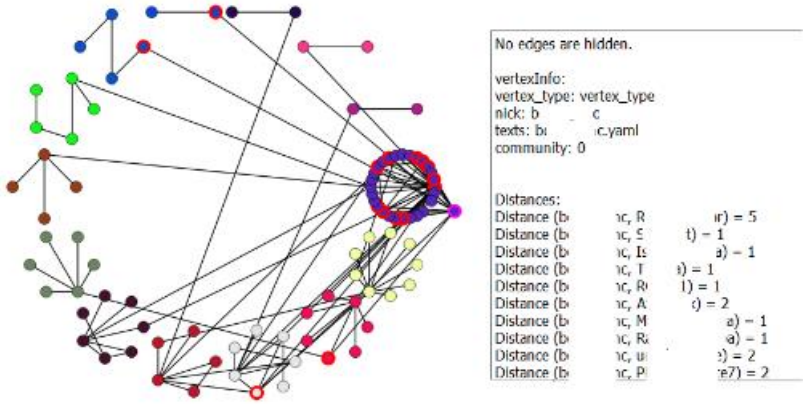

Рисунок 4.14 – Сообщества подграфа S_0, связи вершины bo***nc

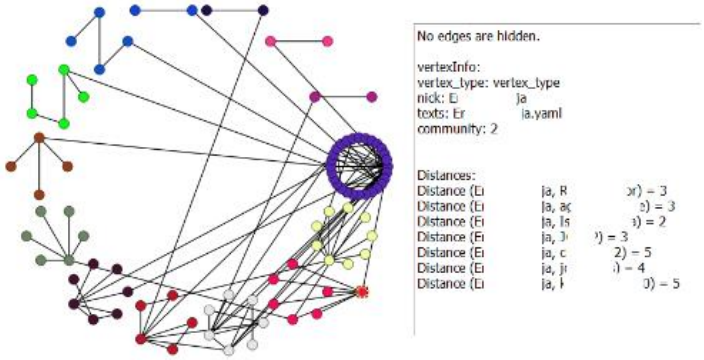

Рисунок 4.15 – Сообщества подграфа S_0, связи вершины Er***ja

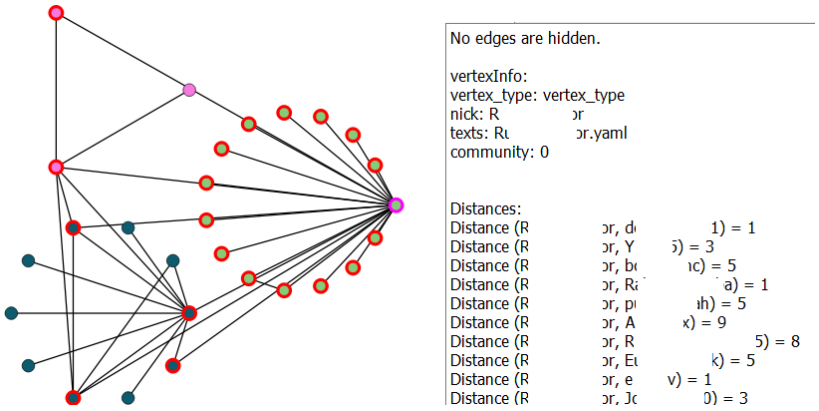

Рисунок 4.16 – Сообщества подграфа, $S_{0,0}$, связи вершины Ru***or

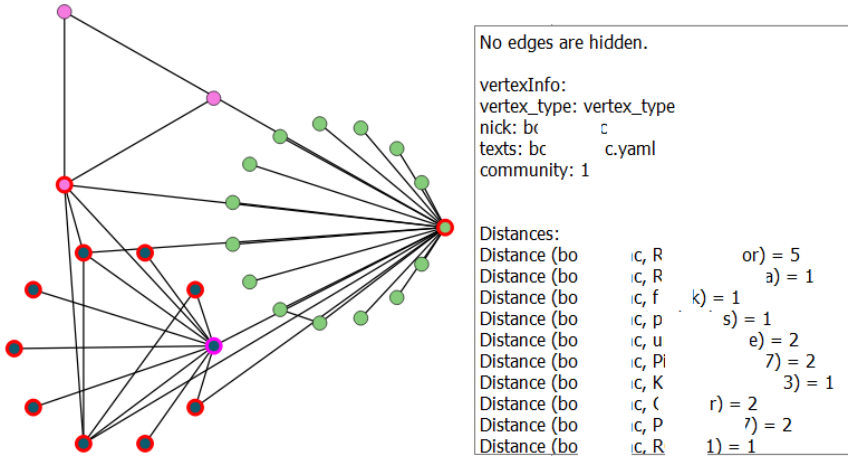

Рисунок 4.17 – Сообщества подграфа, $S_{0,0}$, связи вершины bo***nc

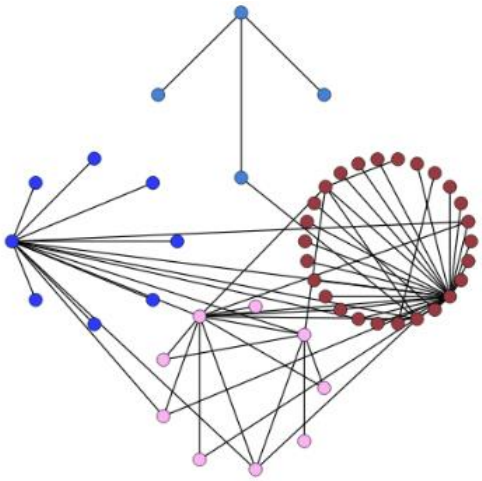

Рисунок 4.18 – Внутреннее устройство сообщества S_1 графа G'_{SptV}

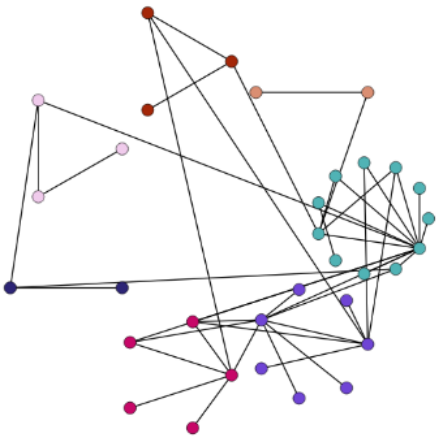

Рисунок 4.19 – Внутреннее устройство сообщества S_2 графа G'_{SptV}

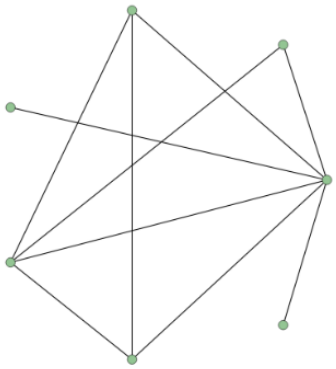

Рисунок 4.20 – Внутреннее устройство сообщества S_3 графа G'_{SptV}

4.3 Исследования текстов сообществ

Рассмотрим описанный в разделе 4.2 первый пример выделения неявных сообществ. Для последующего анализа из 43 сообществ (рисунок 4.1) будем исследовать содержащие более 15 вершин и представленные в таблице 4.1., это сообщества S_i $(i = 0, \dots, 7)$. Как было указано, для $i = 0, 1, 2, 4$ них имеются относительно высокие значения параметров $k_{S_i}(G')$ и $\underset{v \in S_i}{max}\, w^*(v)$, характеризующих взаимодействие внутри сообществ.

В силу особенностей процедуры импорта данных и формата хранения файлов для каждого из S_i можем легко получить тексты, которые являются объединением текстовых данных всех вершин соответствующего сообщества. В этом процессе также удаляются рабочие спецсимволы, относящиеся к пользователям, рассматриваются тесты на русском языке. Автоматизированная обработка полученных естественных текстов позволяет составить соответствующие частотные словари для слов и словосочетаний. Размеры полученных словарей в единицах записей приведены в таблице 4.5. Подробнее этот процесс описан в главе 6 [112, 113].

Вычислена попарная ранговая корреляция для этих словарей и буквосочетаний различной длины, результаты приведены в таблицах 4.6, 4.7, 4.8, 4.9. Как видно из таблицы 4.6, найдены ожидаемо высокие значения корреляции для буквосочетаний в 2 символа. Этот показатель характеризует язык, на котором написаны тексты.

Для буквосочетаний большей длины попарная корреляция существенно уменьшается, показывая низкую согласованность между частотными словарями

текстов у разных сообществ (таблицы 4.7, 4.8, 4.9). В свою очередь данные различия позволяют сделать выводы о различии содержательной направленности текстов и целесообразности выделенных сообществ. Как и представленные в таблицах 4.10 и 4.11 сравнения частотных словарей для псевдооснов и именных групп, где значения корреляции близки к нулю. В таблице 4.12 показаны результаты аналогичного сравнения для глагольных групп, где различия настолько существенны, что находятся около -1. Такие серьезные различия для глагольных групп свидетельствуют о возможности рассматривать их как дифференцирующие признаки сообществ. Получаем, что выделенные сообщества могут быть разделены с точки зрения направленности превалирующих в них текстах на активность пользователей сети.

Помимо сравнения частотных словарей для текстов рассматриваемых сообществ S_i $(i = 0, \dots, 7)$ с применением описанных в разделах 6.1 и 6.3 методов вычислены определенные их статистические характеристики. Общее число таких статистических характеристик превосходит 20, многие из них относятся к психолингвистическим характеристикам. Таблица 4.13 показывает, что стандартные характеристики не дают особых возможностей выделить различия между сообществами.

Возьмем 4 характеристики, приведенные в таблице 4.14 и имеющие в том числе направленность на совершение действий. Далее проведем сравнение не только между текстами выделенных сообществ, но и с некоторыми стандартизированными наборами. Были взяты два больших таких набора: литературные тексты (*lit)* и политические тексты, в том числе с противозаконными призывами (*nt)*.

Из таблицы 4.14 видно, что как раз у S_i для $i = 0, 1, 2, 4$ характеристики, относящиеся к коэффициентам действия отличаются по своим значениям от остальных сообществ и набора политически-новостных текстов. Значения у четырех сообществ ниже, чем у остальных, и подтверждают характер текстов, не нацеленных на побуждение к активным действиям.

Приведенные в таблицах данные и их интерпретация показывают, что для полученных сообществ на рассматриваемом графе разнообразные лингвистические характеристики свидетельствуют о содержательно верном их выделении. Можно

интерпретировать отличие психолингвистических характеристик для разных сообществ, как подтверждение корректности методики нахождения неявных сообществ.

Таблица 4.5 – Размеры текстов и частотных словарей текстов сообществ.

Словари текстов	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Словарь существительных	1359	1771	1125	977	1569	1073	1058	847
Словарь глаголов	826	1023	584	673	933	641	582	532
Словарь прилагательных	464	579	355	337	518	321	319	275
Словарь псевдоосновов	3886	4984	2966	2882	4461	2977	2859	2388
Словарь именных групп	3702	5203	2483	2576	4507	2398	2181	1898
Словарь глагольных групп	1017	1541	655	733	1362	711	647	534

Таблица 4.6 – Сравнение словарей буквосочетания длиной 2

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,97	1						
S_2	0,96	0,97	1					
S_3	0,94	0,95	0,94	1				
S_4	0,95	0,97	0,95	0,94	1			
S_5	0,93	0,93	0,92	0,93	0,94	1		
S_6	0,96	0,95	0,95	0,94	0,95	0,94	1	
S_7	0,94	0,94	0,93	0,93	0,94	0,93	0,93	1

Таблица 4.7 – Сравнение словарей буквосочетания длиной 4

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,46	1						
S_2	0,44	0,43	1					
S_3	0,54	0,45	0,35	1				
S_4	0,57	0,48	0,40	0,56	1			
S_5	0,58	0,42	0,37	0,49	0,54	1		
S_6	0,54	0,41	0,34	0,45	0,49	0,482	1	
S_7	0,56	0,43	0,34	0,52	0,57	0,55	0,44	1

Таблица 4.8 – Сравнение словарей буквосочетания длиной 5

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,47	1						
S_2	0,28	0,30	1					
S_3	0,31	0,34	0,11	1				
S_4	0,35	0,42	0,17	0,31	1			
S_5	0,36	0,35	0,21	0,18	0,24	1		
S_6	0,32	0,28	0,11	0,14	0,15	0,20	1	
S_7	0,27	0,30	0,11	0,21	0,25	0,22	0,10	1

Таблица 4.9 – Сравнение словарей буквосочетания длиной 6

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,25	1						
S_2	0,04	0,10	1					
S_3	0,11	0,12	−0,14	1				
S_4	0,13	0,17	−0,04	0,10	1			
S_5	0,19	0,18	0,01	0,05	0,06	1		
S_6	0,13	0,11	−0,04	−0,03	−0,03	0,067	1	
S_7	0,05	0,03	−0,13	−0,03	−0,005	0,001	−0,08	1

Таблица 4.10 – Сравнение словарей псевдоосновов

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,36	1						
S_2	0,23	0,25	1					
S_3	0,28	0,31	0,21	1				
S_4	0,31	0,31	0,21	0,32	1			
S_5	0,29	0,29	0,21	0,27	0,26	1		
S_6	0,31	0,29	0,23	0,26	0,26	0,28	1	
S_7	0,35	0,32	0,25	0,32	0,33	0,28	0,27	1

Таблица 4.11 – Сравнение по именным группам

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	0,11	1						
S_2	−0,05	0,04	1					
S_3	−0,09	0,02	−0,24	1				
S_4	0,13	0,13	−0,06	0,01	1			
S_5	−0,06	−0,01	−0,17	−0,22	−0,05	1		
S_6	−0,09	−0,01	−0,24	−0,28	−0,07	−0,22	1	
S_7	−0,09	−0,11	−0,25	−0,20	−0,07	−0,25	−0,26	1

Таблица 4.12 – Сравнение по глагольным группам

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	−0,92	1						
S_2	−0,92	−0,59	1					
S_3	−0,93	−0,73	−0,96	1				
S_4	−0,95	−0,89	−0,79	−0,85	1			
S_5	−0,91	−0,71	−0,96	−0,96	−0,84	1		
S_6	−0,88	−0,57	−0,96	−0,95	−0,76	−0,95	1	
S_7	−0,75	−0,16	−0,96	−0,89	−0,46	−0,94	−0,94	1

Таблица 4.13 – Стандартные характеристики

Характеристика	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Средняя длина словоупотреблений в символах	5,09	5,14	5,23	4,92	5,18	5,02	5,04	5,12
Средняя длина предложения в словоупотреблениях	8,50	8,66	8,34	9,46	10,1	8,75	7,72	9,80
Средняя длина именных групп в словоупотреблениях	2,76	2,84	2,74	2,80	2,76	2,72	2,69	2,79
Средняя длина глагольных групп в словоупотреблениях	2,96	3,05	2,95	3,02	2,97	2,93	2,91	2,98

Таблица 4.14 – Психолингвистические факторы наборов текстов

Характеристика	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	*nt*	*lit*
Коэффициент лексического разнообразия 2 (ЛС1)	0,35	0,34	0,40	0,36	0,34	0,40	0,40	0,42	0,16	0,12
Коэффициент действия 1 (КД1)	1,93	1,88	1,73	2,16	1,95	2,15	1,95	2,23	2,59	1,82
Коэффициент действия 2 (КД2)	2,14	2,10	1,92	2,37	2,20	2,38	2,16	2,43	3,00	2,14
Коэффициент опредмеченности действия (КОД)	0,59	0,58	0,49	0,64	0,56	0,64	0,55	0,68	0,61	0,69

4.4 Выводы по главе 4

1. Предложенный и реализованный «Метод ядра» для выделения непересекающихся сообществ на взвешенных графах является главным результатом главы и предусматривает выделение ключевой компоненты на основании вычисляемых в явном виде характеристик графа.

2. Приведенные в главе экспериментальные результаты работы метода на реальных данных из сети *Twitter* демонстрируют его применимость для решения задачи распознавания лидеров мнений.

3. Представленный анализ объединенных текстов из полученных «Методом ядра» сообществ, подтверждает качество работы метода.

4. Основные результаты, представленные в главе 4, опубликованы в следующих работах: [70], [73]. В работе [70] соискателю принадлежит алгоритм Метода ядра и методика его применения, анализ результатов по итогам применения.

ГЛАВА 5 МЕТОД «ГАЛАКТИК» ВЫДЕЛЕНИЯ СООБЩЕСТВ

В данной главе представлен предложенный в работе [68] метод Галактик для выявления сообществ. Метод применяется на взвешенных графах взаимодействующих объектов, построенных согласно модели, описанной в главе 2, и позволяет выделять пересекающиеся сообщества. Представлены примеры его применения на реальных данных, продемонстрирована методика оценки эффективности работы данного метода с помощью методов компьютерной лингвистики, примененных к текстовым атрибутам вершин графа.

5.1 Алгоритм метода «Галактик»

Рассмотрим граф взаимодействующих объектов $G(V, \tilde{E})$, сформированный при импорте данных из сети *Telegram*-каналов согласно описанной в главе 2 процедуре из графа $G(V, E)$. При построении взвешенного графа использовалась (U, M, R)-модель информационного взаимодействия [66], веса ребер построены согласно формуле (2.6). Целесообразность поиска пересекающихся сообществ для таких сетей основана на том, что содержание записей одного и того же канала может иметь несколько различных специализаций. Это, в свою очередь, повлечет за собой цитирование и упоминание других каналов разных направлений. Поэтому при построении метода для подобных коммуникационных сетей важно предусмотреть возможность для вершин принадлежать в итоге более, чем к одному сообществу.

Если изначально выделить на графе сообщества, то каждое из них можно назвать мета-вершиной. При этом, в случае, если были получены индивидуальные сообщества из одной вершины, исключаем их из дальнейшего рассмотрения. Либо эта вершина вошла еще в какое-то из сообществ, либо она считается «мусорной». Каждая мета-вершина, таким образом, состоит из каких-то вершин исходного графа. Далее, если исходно был рассмотрен взвешенный граф, то для двух таких произвольных мета-вершин можно определить новое ребро и вес для него. За счет

этого строится мета-граф. Определим вес ребер в нем равным сумме весов ребер, соединяющих те пары вершин, которые лежат в двух рассматриваемых сообществах. Для вершин исходного графа, принадлежащих сразу обеим мета-вершинам (т.е. лежащим в пересечении сообществ) вес ребер не добавляется и петли на основании этого не строятся. Исходный граф, построенный согласно модели, описанной в разделе 2.4, также является простым взвешенным, поэтому петель у нового мета-графа не будет.

Далее для построенного так мета-графа можно выделить сообщества уже на нем. Это даст возможность объединить вершины исходного графа внутри выделенных мета-сообществ и составить итоговые пересекающиеся сообщества. Подобный подход позволяет получать достаточно вариативные варианты сообществ для каждого из каналов сети.

Согласно модели, описанной по формуле (2.6) строим взвешенный граф взаимодействующих объектов $G(V, \tilde{E})$, где $\tilde{E}$ – множество ненулевых ребер. Явно представим алгоритм для метода Галактик по выделению сообществ.

Алгоритм 5.1. Метод «Галактик»

Шаг 1. На графе $G(V, \tilde{E})$ выделяются пересекающиеся сообщества. Получаем набор сообществ $S_{G(V,\tilde{E})}$.

Шаг 2. Строится множество мета-вершин $\tilde{S}_{G(V,\tilde{E})}$, которое состоит из элементов $S_{G(V,\tilde{E})}$, содержащих более, чем одну исходную вершину.

Шаг 3. Для множества $\tilde{S}_{G(V,\tilde{E})}$ вершин нового мета-графа $\tilde{\tilde{G}}$ вычисляются веса его ребер и строится сам граф $\tilde{\tilde{G}}$.

Шаг 4. На графе $\tilde{\tilde{G}}$ выделяются непересекающиеся сообщества.

По итогам работы алгоритма получаем выделение сообществ для вершин исходного графа. Также для каждого из сообществ с учетом строения файлов, описанного в главе 2, формируется общий текст, основанный на текстовых данных каждой из вершин, в него входящих.

На шаге 1 может быть использован, например, метод *Connected Iterative Scan* (CIS) [114] или *BigClaim* [115]. На шаге 4 для графов можно использовать, например, алгоритм *Louvain* [50] или *Infomap* [53].

Таким образом, на шаге 1 для графа $G(V, \tilde{E})$ и модулярности $Q^{overlap}_{G(V,\tilde{E})}$, определенной по аналогии с (3.4), ищется набор сообществ $S_{G(V,\tilde{E})}$ для локального максимума $Q^{overlap}_{G(V,\tilde{E})}$. Обозначим множество сообществ после 2 шага и «уборки мусора» как $\tilde{S}_{G(V,\tilde{E})} = \{\tilde{S}_0, \ldots, \tilde{S}_{r_2}\}$, где $(r_2 + 1)$ – количество сообществ после выполнения шага 2. На шаге 3 из $\tilde{S}_{G(V,\tilde{E})}$ строится множество мета-вершин для нового графа $\tilde{\tilde{G}}$ и ребра между ними.

На 4 шаге для модулярности $Q_{\tilde{\tilde{G}}}$, определенной по аналогии с (3.3), ищется набор сообществ $\tilde{\tilde{S}}_{\tilde{\tilde{G}}} = \left\{\tilde{\tilde{S}}_0, \ldots, \tilde{\tilde{S}}_{r_3}\right\}$ для локального максимума:

$$Q_{\tilde{\tilde{G}}} \to \max_{\tilde{\tilde{S}}_{\tilde{\tilde{G}}}} \tag{5.1}$$

Тут $r_3 + 1$ – количество сообществ после выполнения этого шага. А так как $\tilde{\tilde{S}}_i = \{\tilde{S}_{i_0}, \ldots, \tilde{S}_{i_r}\}$ для всех i, и при этом $\tilde{S}_{i_k} = \{v_{i_{k1}}, \ldots, v_{i_{kl}}\}$ для всех k, то получаем набор сообществ на исходном графе.

Дополнительной методикой оценки качества для полученного данным алгоритмом выделения сообществ служат описанные в главе 6 процедуры с текстовыми данными.

5.2 Применение метода «Галактик» к реальным данным

Рассмотрим примеры применения метода Галактик. Были импортированы данные пяти *Telegram*-каналов. Для построения графов взаимодействующих объектов использовалась (U, M, R)-модель, в которой с учетом описанного в разделе 2.5 в формуле (2.6) взяты следующие значения весов для факторов взаимодействия: $U = 1,\ M = 2,\ R = 3$. Далее произведен импорт данных по модели, описанной в

главе 2, с разными исходными вершинами различной направленности, разными временными промежутками T и на разную глубину. Получены представленные в таблице 5.1 пять графов G_i для $i = 1, \dots, 5$. За стартовые вершины брались каналы следующих специализаций: финансовая, спортивная, молодежная политика, развлекательная, новостная из сферы образования.

Далее к этим пяти графам последовательно применяется метод Галактик. Ключевые показатели исходных графов и возникающие в процессе метода представлены в таблице 5.1. В соответствии с первым шагом алгоритма выделяются пересекающиеся сообщества. Их количество крайне велико и связано в том числе с разнообразными направлениями контента, представляемого каналами. Зачастую это количество сопоставимо с числом вершин. Что еще раз показывает целесообразность введения мета-сообществ.

Таблица 5.1 – Примеры графов и выделения сообществ методом Галактик

Исходный граф	Кол-во вершин графа $G(V, \tilde{E})$	Кол-во ребер графа $G(V, \tilde{E})$	Кол-во выделенных на шаге 1 сообществ на графе $G(V, \tilde{E})$	Кол-во выделенных на шаге 4 мета-сообществ на графе $\tilde{\tilde{G}}$
G_1	625	6137	430	17
G_2	590	4352	369	20
G_3	773	6611	511	21
G_4	619	2973	258	8
G_5	168	697	112	8

Полученные графы могут быть визуализированы с помощью программного обеспечения, представленного в главе 7. Пример такой визуализации приведен на рисунке 5.1, где изображен граф G_5 до применения к нему метода Галактик. На данном рисунке цветовое выделение вершин не несет в себе дополнительных сведений и выбрано случайным образом. Расстановка вершин на сцене также произвольная.

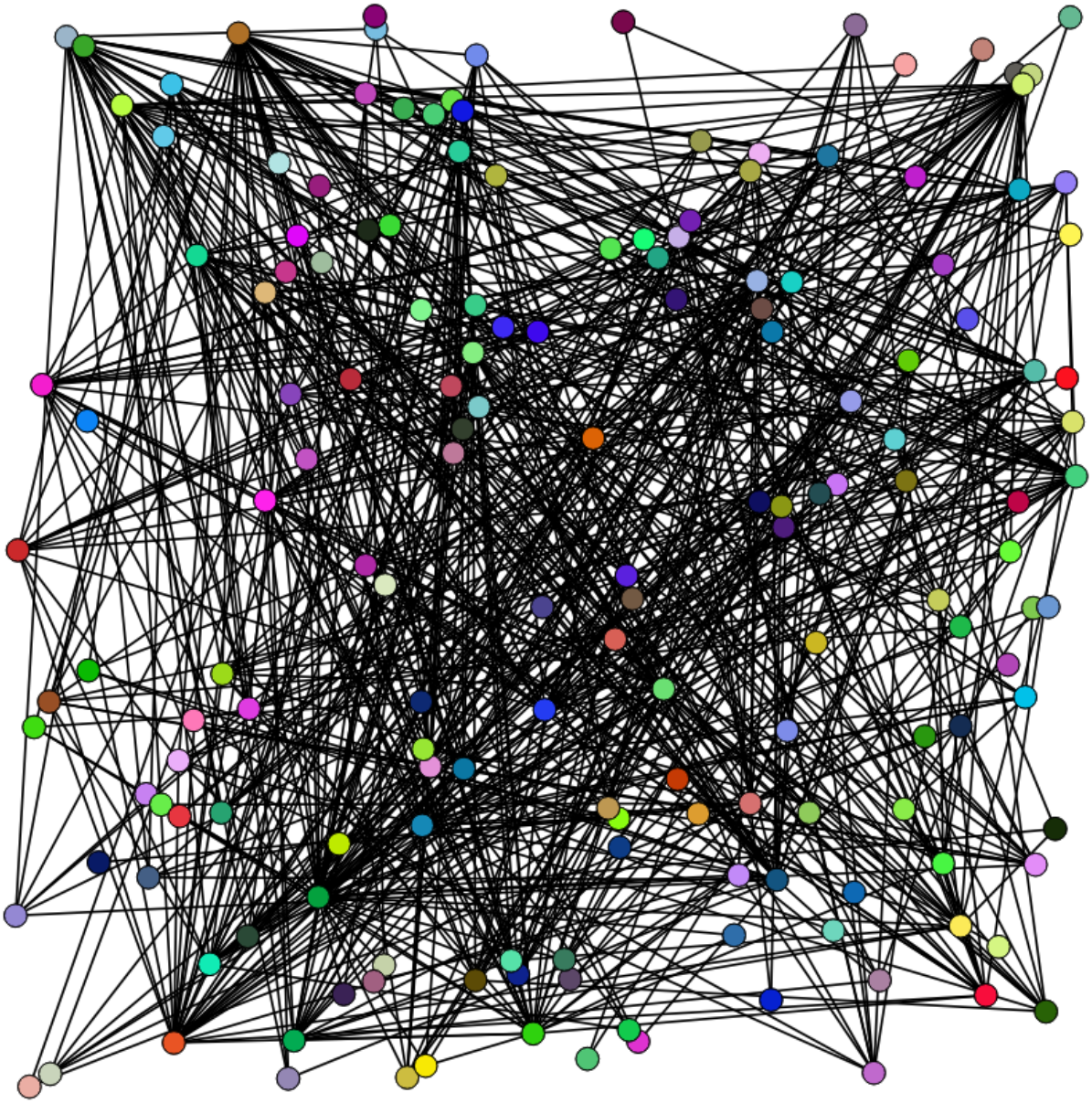

Рисунок 5.1 – Исходный граф G_5 с произвольным расположением вершин

Полученное разбиение графа представлено на рисунок 5.2. Данная визуализация разбиения графа на сообщества реализована в программном комплексе *AVS*, описанном в главе 7. Вершины представлены разноцветными кругами, которые в случае принадлежности к одному сообществу имеют цвет этого сообщества; в случае принадлежности вершины к нескольким сообществам круги разделены на это число секторов, каждый из которых окрашен в цвет соответствующего сообщества. Таким образом, общая визуализация графа осуществлена методом кругового размещения.

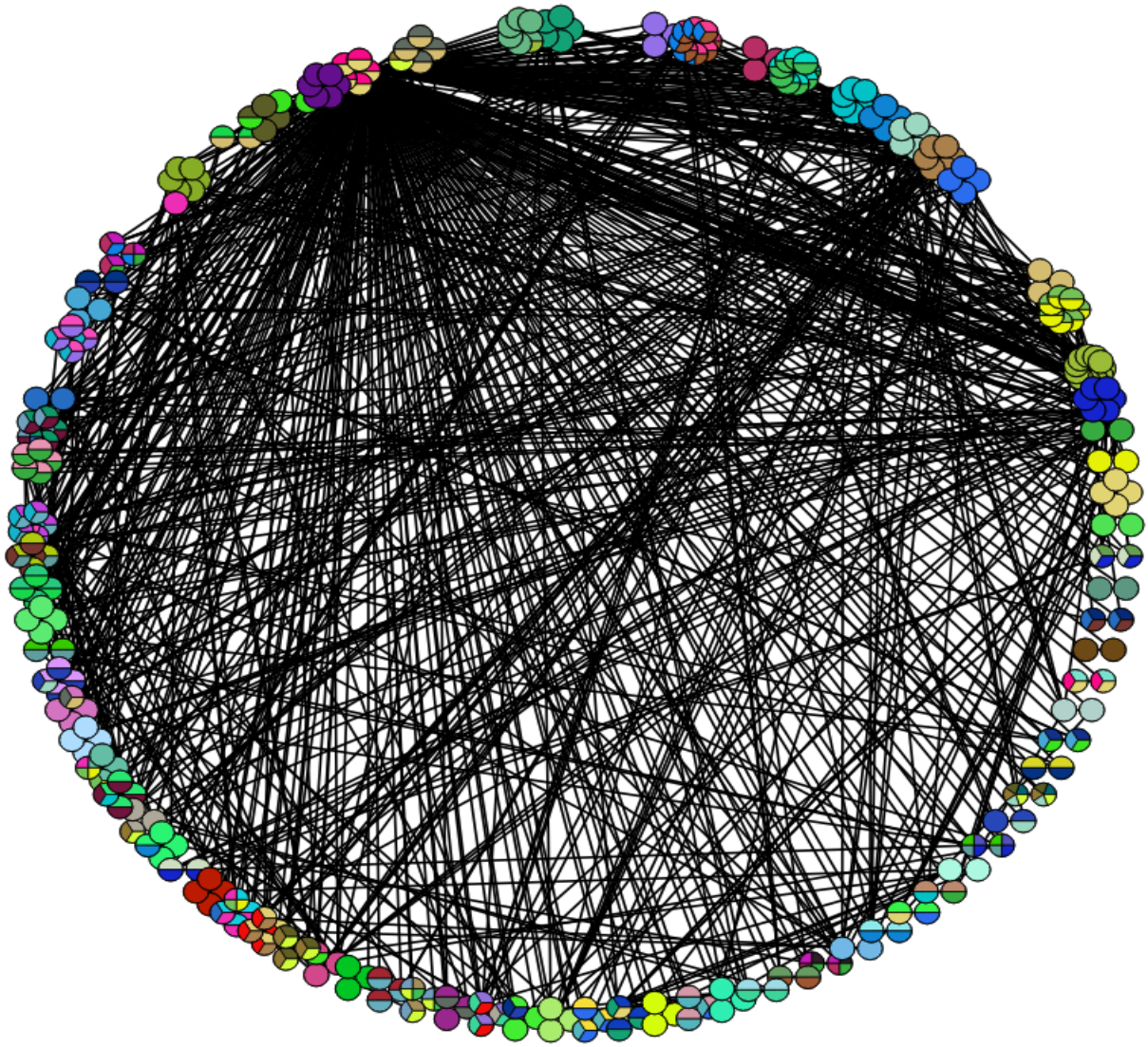

Рисунок 5.2 – Граф G_1 после шага 2

Далее согласно алгоритму производятся 3 и 4 шаги и на $\widetilde{\widetilde{G_1}}$ выделяются 17 мета-сообществ. Количество выделенных мета-сообществ для остальных графов также приведено в таблице. Необходимо отметить некоторые практические результаты, которые следуют из представленных в ней данных. Преобразование графа к мета-сообществам в десятки раз уменьшает количество анализируемых структурных единиц графа. Это представляет возможности для организации эффективного хранения и анализа больших объемов метаданных коммуникационного взаимодействия между объектами. С точки зрения работы с элементами интерфейса получаем существенное сокращение их количества с дополнительной возможностью визуализации по запросу пользователя внутри выделенного сообщества.

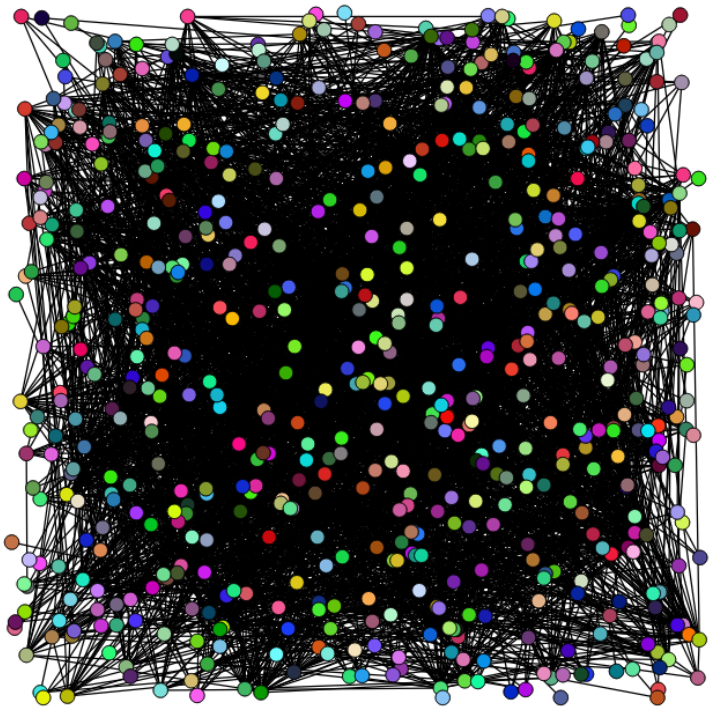

Рисунок 5.3 – Исходный граф G_4 с 619 вершинами

Иллюстрировать это может пример для графа G_4, построенного при старте изначально с вершины, соответствующей развлекательному каналу (рисунок 5.3). По итогам работы алгоритма выделяются 8 мета-сообществ (рисунок 5.4).

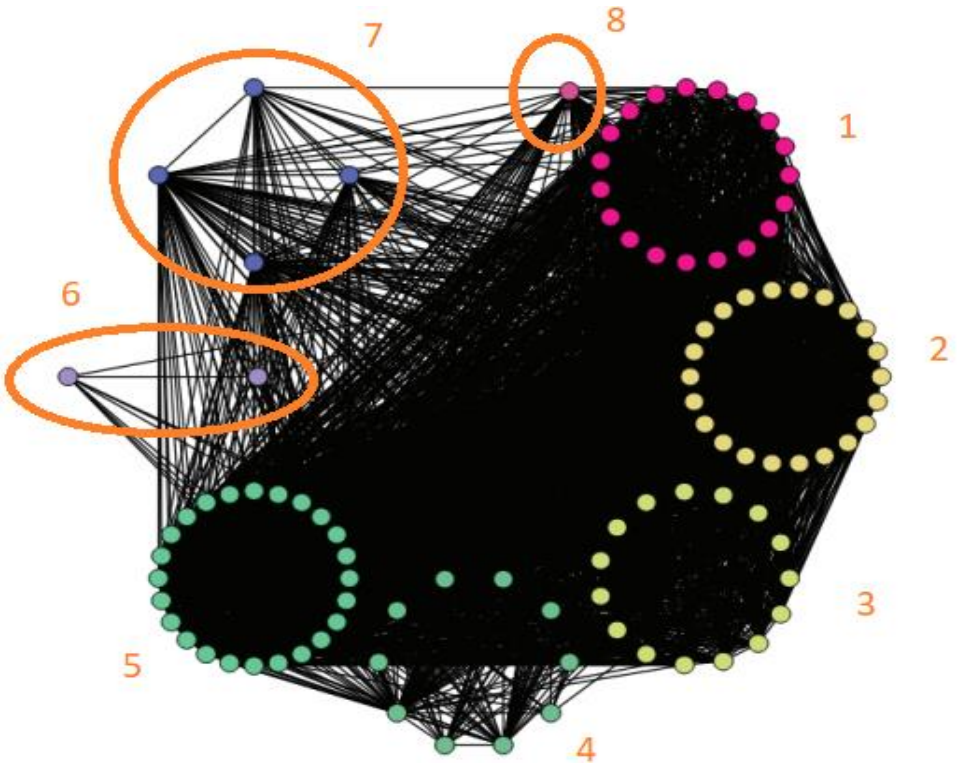

Рисунок 5.4 – Применение метода Галактик к графу G_4: окончательное выделение 8 мета-сообществ

Каждое из 8 выделенных сообществ состоит из мета-вершин, которые представляют собой наборы вершин, отнесенных к различным исходно выделенным сообществам. Пример визуального представления внутри такой мета-вершины, содержащей в свою очередь 19 вершин, представлен на рисунке 5.5. Как и ранее, в случае принадлежности вершины к нескольким сообществам, она разделена на сектора соответствующих цветов.

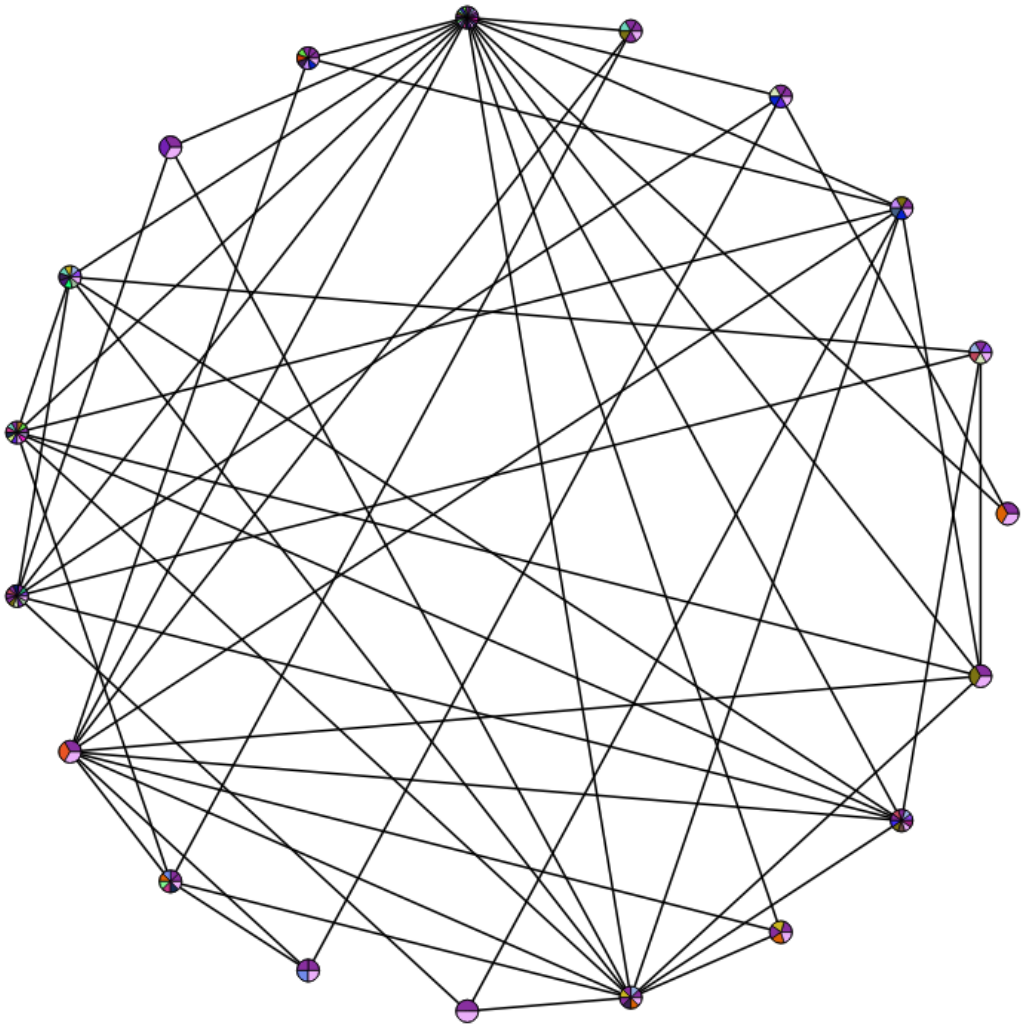

Рисунок 5.5 – Представление внутри одной из мета-вершин G_4 с 19 вершинами

5.3 Обоснование качества выделения сообществ

Для проверки эффективности выделения методом Галактик неявных сообществ на графах взаимодействующих объектов была проведена следующая процедура. Как описано в главе 2, у вершин построенного графа имеются атрибуты, в

том числе текстовые. Такие атрибуты хранятся как отдельные файлы. И в результате импорта данных в рамках (U, M, R)-модели такие файлы содержат тексты постов *Telegram*-каналов соответствующих вершинам. В процессе применения метода Галактик для каждого из выделенных в итоге сообществ сформирован файл, включающий в себя объединение текстов вершин, входящих в это сообщество. Таким образом, каждому из сообществ был поставлен в соответствие «свой» текст. Помимо этого, рассмотрен еще и общий текст, полученный для всего графа. Под анализируемым качеством разбиения далее понимается соответствие полученных сообществ и содержания текстов.

Для анализа качества работы метода Галактик была проведена экспертная оценка тематик выделенных сообществ. В качестве примера посмотрим подробнее на результаты этой оценки для графов G_1 и G_2.

На основе содержания каналов графа G_1 констатировано, что полученные по результатам метода Галактик сообщества в этом графе относятся к следующим направлениям: политические и новостные; про финансы (личные и государственные) и приобретение/ремонт недвижимости; на тему моды и искусства; на разные темы. Распределение по темам приведено в таблице 5.2. В ряде случаев направленность каналов из выделенного сообщества хорошо идентифицируется, но бывают и неоднозначные случаи в силу разностороннего состава контента каналов.

Таблица 5.2 – Тематическая направленность сообществ графа G_1

Направленность содержания каналов, входящих в сообщества	Список итоговых сообществ с данной направленностью текстов	Количество сообществ, идентифицированных как имеющие данную направленность
Финансы (личные и государственные) и приобретение/ремонт недвижимости	0, 1, 11	3
Политические и новостные	2, 3, 5, 13, 14, 15	6
На тему моды и искусства	4, 6, 12, 16	4
На разные темы	7, 8, 9, 10	4

Аналогично, сообщества, выделенных в итоге на графе G_2, по итогам экспертной оценки были отнесены к следующим направлениям: общие спортивные; каналы про футбол; политические; на разные темы. Распределение сообществ по темам для G_2 приведено в таблице 5.3.

Таблица 5.3 – Тематическая направленность сообществ графа G_2

Направленность содержания каналов, входящих в сообщества	Список итоговых сообществ с данной направленностью текстов	Количество сообществ, идентифицированных как имеющие данную направленность
Общие спортивные	2, 11	2
Каналы про футбол	0, 7, 8, 14, 15, 17, 19	7
Политические	4, 9, 13, 16, 18	5
На разные темы	1, 3, 5, 6, 10, 12	6

Стоит отметить, что в некоторых группах у сообществ может быть выделена большая конкретика в направленности текстов, например, сообщество каналов не просто про футбол, а про конкретный футбольный клуб. Таким образом, результаты экспертной оценки тематики полученных сообществ указывают на качественное выделение их методом Галактик.

Тематики выделенных сообществ могут сильно отличаться от таковых у стартовых вершин, формирующих изначально множество V_0. Это обусловлено фактически имевшими место взаимодействиями между объектами. И порой представляет основной интерес при анализе графа.

Для текстов сообществ были подсчитаны психолингвистические характеристики по методике, подробно описанной в главе 6. Здесь приведем уже результаты, основанные на их использовании. Выбранные характеристики показывают, во-первых, лексическую наполненность контента каналов и ее логическую связность. Во-вторых, содержание в текстах агитационной компоненты для информационного воздействия. Подсчет этих характеристик ведется как для каждого из сообществ, так и для общего текста каждого из анализируемых графов. Соответствующие значения для ключевых характеристик приведены в таблице 5.4. Важно отметить, что

разброс значений между графами для каждой из рассмотренных характеристик невелик. Это дополнительно показывает, корректность данных и общую стилистику, характерную для каналов на русском языке в *Telegram*.

Таблица 5.4 – Характеристики текстов

Характеристика	G_1	G_2	G_3	G_4	G_5
Общий объем всех текстов (МБ)	31,1	27,6	40,3	107	27,1
Коэффициент лексического разнообразия 1 (ЛР1)	0,03	0,03	0,03	0,01	0,03
Коэффициент лексического разнообразия 2 (ЛР2)	0,04	0,04	0,04	0,01	0,04
Коэффициент глагольности (КГ)	0,16	0,16	0,16	0,16	0,15
Коэффициент действия 1 (КД1)	1,25	1,24	1,18	1,15	1,06
Коэффициент действия 2 (КД2)	1,53	1,51	1,45	1,44	1,32
Коэффициент опредмеченности действия (КОД)	0,39	0,40	0,38	0,38	0,36
Коэффициент логической связности 1 (ЛС1)	2,26	2,31	2,36	2,43	2,34
Коэффициент логической связности 2 (ЛС2)	0,19	0,19	0,19	0,19	0,18
Коэффициент связности лексики (СЛ)	3,74	3,55	3,59	3,6	3,46

Показатели лексического разнообразия связаны с направленностью канала и для тех из, где имеется большая частота однообразных записей, они ожидаемо ниже. Примером тут являются новостные каналы с большим объемом однотипных текстов. Характеристики наборов текстов, соответствующих выделенным сообществам, для рассматриваемых графов приведены далее в таблицах 5.5 – 5.9. Соответствующие результаты подсчетов и их анализа представлены в том числе в работах [68, 74].

Вторая часть показателей связана с частотами использования глагольных групп в анализируемых текстах. Входящие в эту группу коэффициенты коррелируют между собой. Как правило, они характеризуют направленность текстов на действия и их высокое значение свойственно мотивационным текстам. Для текстов новостных каналов не свойственны высокие значения этих показателей в силу констатации ими фактов. Но среди выделенных сообществ эти показатели были выше ожидаемых в случаях оппозиционно настроенных каналов.

Таблица 5.5 – Данные по текстам G_1

Характеристика	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Размер текстов (Мб)	0,38	0,34	5,05	1,72	0,80	0,49	0,08	7,25	4,96	0,25	2,44	0,87	0,04	1,90	1,40	2,08	1,08
Коэффициент лексического разнообразия 1 (ЛР1)	0,160	0,190	0,060	0,120	0,140	0,190	0,380	0,060	0,070	0,270	0,090	0,140	0,460	0,110	0,120	0,110	0,160
Коэффициент лексического разнообразия 2 (ЛР2)	0,190	0,230	0,090	0,160	0,200	0,250	0,460	0,090	0,100	0,340	0,120	0,190	0,550	0,140	0,160	0,140	0,220
Коэффициент глагольности (СГ)	0,146	0,130	0,160	0,160	0,170	0,170	0,140	0,160	0,160	0,150	0,160	0,159	0,165	0,150	0,163	0,151	0,162
Коэффициент действия 1 (КД1)	1,310	0,860	1,180	1,090	1,210	1,630	0,940	1,360	1,340	1,040	1,160	1,330	1,480	1,180	1,310	1,090	1,370
Коэффициент действия 2 (КД2)	1,600	1,180	1,460	1,390	1,640	1,880	1,230	1,620	1,610	1,290	1,420	1,640	1,670	1,450	1,590	1,360	1,640
Коэффициент опредмеченности действия (КОД)	0,360	0,240	0,370	0,360	0,350	0,470	0,330	0,410	0,420	0,350	0,370	0,370	0,470	0,350	0,420	0,360	0,450
Коэффициент логической связности 1 (ЛС1)	2,370	2,020	2,440	2,180	1,340	2,040	1,780	2,230	2,260	2,110	2,490	2,470	1,790	2,390	2,390	2,320	2,190
Коэффициент логической связности 2 (ЛС2)	0,220	0,190	0,180	0,180	0,190	0,190	0,190	0,190	0,190	0,200	0,200	0,190	0,180	0,180	0,200	0,180	0,200
Коэффициент связности лексики (СЛ)	4,220	3,850	3,780	3,570	3,890	4,160	3,070	3,840	3,690	3,320	3,630	4,122	3,490	3,810	3,720	3,460	3,480

Таблица 5.6 – Данные по текстам G_2

Характеристика	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
Размер текстов (Мб)	1,13	0,323	0,235	0,122	1,475	4,14	0,228	0,195	0,282	1,24	9,715	0,252	0,265	1,722	0,27	0,209	3,67	0,432	0,842	0,84
Коэффициент лексического разнообразия 1 (ЛР1)	0,092	0,284	0,164	0,313	0,115	0,075	0,271	0,228	0,216	0,122	0,050	0,208	0,259	0,104	0,192	0,236	0,070	0,170	0,158	0,142
Коэффициент лексического разнообразия 2 (ЛР2)	0,148	0,332	0,247	0,378	0,156	0,108	0,335	0,314	0,273	0,169	0,069	0,287	0,328	0,146	0,255	0,313	0,098	0,234	0,215	0,189
Коэффициент глагольности (КГ)	0,173	0,152	0,178	0,128	0,146	0,156	0,152	0,155	0,148	0,147	0,155	0,157	0,159	0,153	0,162	0,160	0,162	0,161	0,156	0,153
Коэффициент действия 1 (КД1)	1,928	1,240	1,833	0,886	1,014	1,287	1,238	1,755	1,409	0,927	1,234	1,600	1,367	1,143	1,674	1,530	1,167	1,623	1,097	2,012
Коэффициент действия 2 (КД2)	2,148	1,482	2,026	1,079	1,261	1,539	1,493	2,001	1,713	1,175	1,517	1,807	1,605	1,409	1,905	1,727	1,479	1,822	1,324	2,311
Коэффициент опредмеченности действия (КОД)	0,557	0,408	0,540	0,276	0,338	0,422	0,389	0,435	0,401	0,334	0,386	0,492	0,442	0,372	0,475	0,462	0,395	0,518	0,443	0,417
Коэффициент логической связности 1 (ЛС1)	1,937	2,557	2,022	2,065	2,279	2,455	2,368	1,996	1,832	2,652	2,371	2,262	2,174	2,350	1,952	1,834	2,542	2,006	2,061	1,394
Коэффициент логической связности 2 (ЛС2)	0,190	0,197	0,214	0,199	0,180	0,185	0,202	0,189	0,181	0,185	0,185	0,201	0,198	0,199	0,197	0,207	0,184	0,205	0,180	0,198
Коэффициент связности лексики (СЛ)	4,072	3,412	3,856	3,539	3,391	3,533	3,606	4,472	3,723	3,118	3,658	3,643	3,430	3,414	3,910	3,651	3,388	3,521	2,902	4,979

Таблица 5.7 – Данные по текстам G_3

Характеристика	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Размер текстов (Мб)	0,24	0,09	0,16	0,73	1,83	1,72	3,59	0,16	0,29	0,62	1,16	0,81	0,37	1,03	1,83	18,18	1,33	1,94	0,42	3,48	0,34
Коэффициент лексического разнообразия 1 (ЛР1)	0,234	0,306	0,288	0,160	0,098	0,114	0,071	0,263	0,237	0,165	0,132	0,158	0,196	0,154	0,090	0,040	0,130	0,108	0,204	0,080	0,214
Коэффициент лексического разнообразия 2 (ЛР2)	0,299	0,417	0,362	0,200	0,142	0,162	0,100	0,312	0,299	0,211	0,180	0,207	0,249	0,208	0,128	0,054	0,171	0,145	0,254	0,108	0,275
Коэффициент глагольности (КГ)	0,150	0,145	0,165	0,149	0,162	0,164	0,162	0,151	0,153	0,159	0,156	0,150	0,159	0,154	0,164	0,156	0,150	0,147	0,135	0,158	0,138
Коэффициент действия 1 (КД1)	1,019	0,791	1,241	0,890	1,646	1,446	1,163	1,077	1,194	1,114	1,213	1,083	1,367	1,061	1,303	1,157	1,060	0,974	0,689	1,337	0,860
Коэффициент действия 2 (КД2)	1,360	1,022	1,495	1,102	1,884	1,723	1,475	1,309	1,523	1,371	1,505	1,316	1,642	1,287	1,601	1,419	1,341	1,223	0,915	1,638	1,089
Коэффициент опредмеченности действия (КОД)	0,308	0,299	0,367	0,320	0,446	0,421	0,394	0,364	0,333	0,346	0,363	0,379	0,401	0,424	0,393	0,385	0,336	0,332	0,283	0,383	0,287
Коэффициент логической связности 1 (ЛС1)	2,367	1,700	2,209	1,611	2,943	2,175	2,549	2,305	2,854	2,092	2,997	2,191	2,285	2,132	2,545	2,312	2,425	2,189	2,941	2,471	2,147
Коэффициент логической связности 2 (ЛС2)	0,181	0,188	0,184	0,174	0,190	0,188	0,184	0,207	0,193	0,181	0,193	0,195	0,188	0,178	0,190	0,186	0,185	0,186	0,172	0,188	0,180
Коэффициент связности лексики (СЛ)	3,748	3,111	3,858	3,291	4,261	4,052	3,382	3,406	4,148	3,703	3,846	3,197	3,911	2,920	3,980	3,518	3,590	3,370	2,778	4,043	3,351

Таблица 5.8 – Данные по текстам G_4

Характеристика	0	1	2	3	4	5	6	7
Размер текстов (Кб)	0,33	38,2	13,2	3,4	1,59	37	2,85	10,5
Коэффициент лексического разнообразия 1 (ЛР1)	0,200	0,021	0,038	0,083	0,095	0,021	0,071	0,045
Коэффициент лексического разнообразия 2 (ЛР2)	0,237	0,030	0,053	0,121	0,139	0,030	0,103	0,065
Коэффициент глагольности (КГ)	0,106	0,160	0,153	0,153	0,164	0,160	0,151	0,161
Коэффициент действия 1 (КД1)	0,602	1,179	1,096	1,115	1,155	1,141	1,036	1,239
Коэффициент действия 2 (КД2)	0,776	1,473	1,372	1,342	1,446	1,438	1,329	1,501
Коэффициент опредмеченности действия (КОД)	0,202	0,386	0,356	0,357	0,381	0,380	0,318	0,395
Коэффициент логической связности 1 (ЛС1)	1,365	2,395	2,434	2,207	2,858	2,493	2,609	2,360
Коэффициент логической связности 2 (ЛС2)	0,157	0,187	0,190	0,185	0,186	0,186	0,184	0,188
Коэффициент связности лексики (СЛ)	3,267	3,592	3,542	3,613	3,683	3,560	3,829	3,704

Таблица 5.9 – Данные по текстам G_5

Характеристика	0	1	2	3	4	5	6	7
Размер текстов (Кб)	0,99	0,99	0,57	0,78	1,02	8,91	11,88	1,98
Коэффициент лексического разнообразия 1 (ЛР1)	0,103	0,123	0,159	0,142	0,139	0,043	0,044	0,089
Коэффициент лексического разнообразия 2 (ЛР2)	0,134	0,170	0,216	0,199	0,195	0,065	0,065	0,123
Коэффициент глагольности (КГ)	0,146	0,131	0,132	0,147	0,151	0,155	0,159	0,142
Коэффициент действия 1 (КД1)	0,933	0,713	0,828	0,895	1,076	1,063	1,159	0,866
Коэффициент действия 2 (КД2)	1,139	1,003	1,018	1,140	1,294	1,353	1,411	1,083
Коэффициент опредмеченности действия (КОД)	0,329	0,237	0,278	0,299	0,356	0,350	0,396	0,302
Коэффициент логической связности 1 (ЛС1)	2,544	2,877	1,879	2,022	1,712	2,617	2,221	2,414
Коэффициент логической связности 2 (ЛС2)	0,190	0,183	0,176	0,171	0,175	0,185	0,185	0,174
Коэффициент связности лексики (СЛ)	3,189	3,332	3,400	3,444	3,475	3,571	3,458	3,269

Это объяснение завышенных показателей, выделяющих соответствующие сообщества, подтверждает корректность их выделения рассматриваемым алгоритмом.

Показатели, связанные с разнообразием используемой лексики, часто указывают на сложность и профессиональную направленность предметной сферы текста. Как видно из таблицы 5.5 примером может служить сообщество S_0 на графе G_1, в котором сосредоточены официальные каналы с большим числом подписчиков. Схожие высокие показатели свойственны найденным сообществам, которые ведутся специалистами с профессиональными навыками и инструментами, финансированием. Что логично в силу уровня задействованных ресурсов, присущему поддержке и ведению официальных каналов новостного и политического характера. Выявленная особенность также подтверждает качество выделения каналов методом Галактик.

В дополнение к описанным ранее характеристикам для оценки качества выделения сообществ рассмотрим следующие, примененные в работе [74]. Подсчитаем в именных группах для анализируемого текста среднее число так называемых подгрупп. Аналогичное значение вычисляется и для глагольных групп, также число подгрупп.

Данные показатели являются стандартными для компьютерной лингвистики и характеризуют сложность описываемых в текстовых данных понятий. Значения этих маркеров выделяют каналы, посвященные профессиональным сферам. Как иллюстрация приведены эти значения для графа G_4 в таблице 5.10. По экспертной оценке, каналы из выделенных сообществ S_6 и S_0 как раз посвящены более содержательно сложным тематикам, тогда как канал S_7 относится к спорту и букмекерским конторам. Что также подтверждает качество выделения сообществ методом Галактик.

Теперь рассмотрим ранговый анализ словарей для объединенных текстов сообществ. Данный метод оценки корректности их выделения описан далее в главе 6. Применим его для разных сообществ единой, например, для новостной направленности. В таблицах 5.11 и 5.12 показаны результаты для G_1. Так как полученные

значения находятся либо около нуля, либо отрицательные, а в таблице 5.12 даже ближе к -1, то это подтверждает корректность выделения алгоритмом соответствующих сообществ.

Таблица 5.10 – Характеристики nsgNG и nsgVG для текстов сообществ графа G_4

	Среднее количество числа «подгрупп» в одной именной группе (nsgNG)	Среднее количество числа «подгрупп» в одной глагольной группе (nsgVG)
G_4	2,9	3,4
S_0	3,3	3,7
S_1	2,8	3,3
S_2	3,1	3,6
S_3	2,7	3,3
S_4	2,8	3,3
S_5	2,9	3,5
S_6	3,5	4
S_7	2,7	3,2

Аналогично в таблицах 5.13 и 5.14 представлен ранговый анализ словарей текстов сообществ из группы схожей тематики, но для графа G_2, это группа политических/экономических сообществ. Сравнение частотных словарей прилагательных и здесь дает значения коэффициентов попарной ранговой корреляции в районе 0, то есть словари различаются значительно. Словари глаголов дают корреляцию, близкую к -1, что указывает на различие в направленности текстов на активность у выделенных сообществ.

Таблица 5.11 – Сравнение словарей прилагательных текстов сообществ графа G_1

Номер сообщества	**2**	**3**	**5**	**13**	**14**	**15**
2	1					
3	0,17	1				
5	0,06	0,06	1			
13	−0,11	0,14	−0,12	1		
14	0,15	−0,07	0,15	−0,14	1	
15	0,02	0,13	−0,11	0,03	0,15	1

Таблица 5.12 – Сравнение словарей глаголов текстов сообществ графа G_1

Номер сообщества	**2**	**3**	**5**	**13**	**14**	**15**
2	1					
3	−0,91	1				
5	−0,89	−0,69	1			
13	−0,92	−0,63	−0,86	1		
14	−0,85	−0,79	−0,71	−0,75	1	
15	−0,89	−0,78	−0,86	−0,91	−0,86	1

Таблица 5.13 – Сравнение словарей прилагательных текстов сообществ графа G_2

Номер сообщества	**4**	**9**	**13**	**16**	**18**
4	1				
9	−0,13	1			
13	0,18	−0,11	1		
16	0,13	0,09	−0,06	1	
18	−0,05	0,03	−0,14	0,12	1

Таблица 5.14 – Сравнение словарей глаголов текстов сообществ графа G_2

Номер сообщества	**4**	**9**	**13**	**16**	**18**
4	1				
9	−0,64	1			
13	−0,81	−0,86	1		
16	−0,72	−0,76	−0,83	1	
18	−0,69	−0,91	−0,89	−0,87	1

Таким образом, анализ психолингвистических факторов, экспертная оценка и ранговый анализ словарей текстов выделенных методом Галактик сообществ дают положительную оценку корректности этого выделения.

5.4 Выводы по главе 5

1. Предложенный и реализованный метод Галактик, основан на конструировании алгоритма из базовых. Метод Галактик позволяет для графов больших размеров посредством выделения мета-сообществ сокращать количество элементов визуализации, что повышает эффективность визуального анализа распространения и

хранения информации в сетях взаимодействующих объектов больших объемов. Как итог работы алгоритма выделяются пересекающиеся сообщества на взвешенных графах.

2. Представленные экспериментальные результаты работы метода Галактик на данных, импортированных из сети *Telegram*-каналов, показывают его применимость для решения соответствующих актуальных задач.

3. Приведенные последующие за выполнением метода Галактик применение методов компьютерной лингвистики и экспертной оценки тематик полученных сообществ подтверждают качество полученного разбиения.

4. Основные результаты, представленные в главе 5, опубликованы в следующих работах: [68, 71, 72, 73, 74]. В работе [68] соискателю принадлежит алгоритм метода Галактик и методика его применения.

ГЛАВА 6 МЕТОДИКИ ОЦЕНКИ КАЧЕСТВА ВЫДЕЛЕНИЯ СООБЩЕСТВ

В данной главе представляется метод оценки корректности выделения сообществ на графе с помощью алгоритмов компьютерной лингвистики, предложенный и разработанный в работах [113, 116, 117, 118, 119, 120, 121].

Детально в рамках метода представлены ранговый анализ словарей текстов полученных сообществ и подтверждение качества выделенных сообществ за счет анализа статистических характеристик их текстов.

Оценка показателя субъектности рассмотрена в работах [65, 112], где вклад автора заключается в применении методов анализа графов взаимодействующих объектов, полученных при импорте данных из социальных сетей для формирования психологических показателей социального взаимодействия.

6.1 Анализ текстов неявных сообществ

В случае, если граф взаимодействующих объектов построен для социальной сети, сети мгновенного обмена сообщениями или иной сети, в которой объекты обмениваются текстовой информацией, вершины такого графа по итогам импорта данных предполагают текстовые атрибуты. Для каждого из неявных сообществ, выделенных описанными в предыдущих главах методиками, формируются наборы текстов на основе атрибутов вершин, входящих в это сообщество. Как правило, импортируются текстовые данные из сообщений, связанных с постами и/или комментариям к ним от пользователей/иных объектов исходной сети, соответствующих вершинам сообщества. Из полученной информации формируются единые текстовые данные для каждого сообщества. Считаем, что неявные сообщества представлены текстами преимущественно на русском языке. Анализ полученных массивов текстов методами компьютерной лингвистики позволяет оценить взаимодействие объектов, соответствующих вершинам, входящим в сообщества, и, таким образом,

оценить качество выделения неявных сообществ. Как и было описано в главе 1, оценка качества выделенных на графе сообществ крайне актуальна на сегодняшний день.

Прежде всего при анализе полученных для каждого из сообществ текстов необходимо определить их лингвистические характеристики. Сделать это можно с помощью методов автоматизированной обработки [113, 116, 117, 121, 122, 123]. Для реализации этой задачи разработано и реализовано программное обеспечение, позволяющее определять характеристики корпусов текстов и сравнение корпусов корреляционным анализом [113].

Данный программный продукт позволяет выделять словоупотребления с последующим морфологическим анализом [121, 122, 123] и их сопоставлением категориям. Вычислительными процедурами определяются такие грамматические категории и характеристики как: существительное, прилагательное, глагол несовершенного вида, предлог, глагол совершенного вида, количественное числительное, порядковое числительное, местоимение, местоименное прилагательное, сокращение, аббревиатура, фамилия, имя, отчество, причастие, союз, наречие, частица, междометие, топоним; именительный падеж, родительный падеж, дательный падеж, винительный падеж, творительный падеж, предложный падеж, единственное число, множественное число, мужской род, женский род, средний род, первое лицо, второй лицо, третье лицо, одушевленное, неодушевленное, настоящее время, прошедшее время, будущее время, повелительное наклонение, деепричастие и т.п.

Определяются канонические (начальные) формы слова, для которых составляются далее частотные словари. Анализируются не только словоформы, но и именные группы и глагольные группы целиком. Для выделения именных и глагольных групп применяются процедуры синтаксического анализа.

Под именной группой понималась группа слов, где основным словом является существительное, а остальные слова синтаксически подчинены ему. Под глагольной группой понималось такое словосочетание, где основное слово – глагол. Связи

между глаголом и обнаруженными именными группами устанавливались на основании синтаксического анализа предложения. Для анализа глагольного управления использовался электронный словарь, включающий две тысячи наиболее часто употребляемых глаголов русского языка [113, 122, 124].

Анализируя каждое предложение, рассматривают употребления слов, на основе которых выделяют группы, состоящие из существительных и глаголов. В ситуации, когда неоднозначность морфологического анализа отдельных слов приводит к появлению нескольких именных или глагольных групп с идентичным составом, они считаются повторяющимися. Часть слова без аффиксов – псевдооснова, служит как возможная характеристика текста [123].

Далее описанные лингвистические характеристики текстов сообществ используются для применения метода рангового анализа словарей текстов и подсчета статистических характеристик текста.

6.2 Ранговый анализ словарей текстов

Пусть необходимо провести сравнение некоторого набора текстовых данных. Тогда для выявленных в каждом тексте этого набора лингвистических характеристик составляются частотные словари. После сортировки по частоте вхождения в словарь они получают соответствующие ранги внутри него. Возьмем произвольную пару из рассматриваемого набора словарей и сопоставим их записям случайные величины X и Y. Тогда можно посчитать коэффициент корреляции для построенных рангов. Может возникнуть ситуация, что некоторый элемент входит только в один из словарей, тогда во втором ему ставим нулевую частоту для подсчета ранга. Таким образом уравнивается количество элементов в двух словарях. На практике размер словарей берется с некоторым ограничением $n \leq 10000$ путем отбрасывания «хвоста» – элементов с самыми маленькими частотами. В большинстве случаев в «хвосте» содержатся элементы исходными значениями встречаемости близкими или равными единице, т.е. со значениями частот в районе нуля.

Для первого словаря берем выборку $X^n = \{X_i\}_{i=1}^n$, для второго словаря берем $Y^n = \{Y_i\}_{i=1}^n$. Обозначаем стандартно за $\overline{X^n}$ и $\overline{Y^n}$ средние значения для этих выборок. Определяем ковариацию и дисперсию:

$$cov(X^n, Y^n) = \frac{1}{n}\sum_{i=1}^n (X_i - \overline{X^n})(Y_i - \overline{Y^n}), \quad (6.1)$$

$$\sigma X^n = \sqrt{\frac{1}{n}\sum_{i=1}^n (X_i - \overline{X^n})^2} \quad (6.2)$$

Ранги после сортировки у входящих в словарь элементов выборок обозначим за rgX^n и rgY^n. Среднее значение ранга с учетом размера словаря получается равным $\frac{n+1}{2}$. Введенные обозначения дают возможность определить коэффициент попарной ранговой корреляции [125]:

$$r = r(rgX^n, rgY^n) = \frac{cov(rgX^n,\ rgY^n)}{\sigma rgX^n \cdot \sigma rgY^n} \quad (6.3)$$

Наконец, подставив в формулу (6.3) выражения из (6.1) и (6.2), с учетом среднего значения ранга имеем:

$$r = \frac{\sum_{i=1}^n (rgX_i - \frac{n+1}{2})(rgY_i - \frac{n+1}{2})}{\sqrt{\sum_{i=1}^n \left(rgX_i - \frac{n+1}{2}\right)^2}\sqrt{\sum_{i=1}^n \left(rgY_i - \frac{n+1}{2}\right)^2}} \quad (6.4)$$

Некоторые элементы внутри одного словаря могут иметь одинаковые значения ранга. При этом коэффициент попарной ранговой корреляции будет принимать отличающиеся значения, смотря какой порядок для таких элементов выбран. Поэтому берется усредненное значение. А именно, рассматриваются все возможные перестановки между собой элементов с одинаковым значением частоты и усредняется значение r по ним.

С использованием такого инструментария для текстовых данных двух выделенных сообществ составляются частотные словари. После их сортировки и уравнивания вычисляются значения усредненного коэффициента для выбранных элементов согласно формуле (6.4). В качестве лингвистических характеристик, по которым составляются словари, могут быть взяты буквосочетания длины от 2 до 6 символов, именные и глагольные группы и т.п.

Для примера рассмотрим графы G_4 и G_5 из Главы 5, полученные путем импорта данных из сети *Telegram*-каналов. На каждом из этих графов методом Галактик было выделено по 8 сообществ [119]. Для текстов, соответствующих выделенным на этих графах сообществам S_i составим словари частот некоторых лингвистических характеристик (таблица 6.1 и таблица 6.2).

Согласно формуле (6.4) подсчитаны соответствующие значения для всех пар словарей различных характеристик.

Таблица 6.1 – Размеры частотных словарей в единицах записей для G_4

Словари	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Буквосочетаний длиной 3 символа	5358	14161	12252	10092	8058	13785	8763	11958
Буквосочетаний длиной 4 символа	13224	68159	51774	37820	26166	65348	29962	50583
Буквосочетаний длиной 5 символа	17990	155459	106936	72282	44182	148137	53283	107010
Существительных	1373	12890	9100	6426	3785	12641	4507	9292
Глаголов	478	7581	5151	2947	1927	7255	2134	5109
Прилагательных	583	6182	4599	2838	1710	6128	2202	4336
Именных групп	11103	805899	368445	104458	45755	781353	94448	263547
Глагольных групп	134	14831	6145	1253	506	13413	1238	3682
Псевдооснов	4243	60280	38746	22625	12957	58487	16508	38054

Таблица 6.2 – Размеры частотных словарей в единицах записей для G_5

Словари	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Буквосочетаний длиной 3 символа	6429	6956	6174	6966	7749	11272	12475	8103
Буквосочетаний длиной 4 символа	18536	20912	17572	20366	25291	44828	54111	26704
Буквосочетаний длиной 5 символа	29077	33816	26633	31953	42117	90963	116542	45980
Существительных	2324	2990	2253	2771	3675	7926	10075	3959
Глаголов	1124	1270	882	1136	1574	4076	5831	1822
Прилагательных	1107	1385	1039	1208	1539	3904	4752	1858
Именных групп	31562	42606	20450	29896	29266	280770	325056	74379
Глагольных групп	416	501	219	407	531	3822	4706	850
Псевдооснов	7830	9706	7048	8850	11516	32135	42875	13887

Далее конкретные примеры значений коэффициента корреляции приведены в таблицах 6.3, 6.4 и 6.5, где представлены значения коэффициента корреляции для конкретных словарей всех сообществ одного из графов как указано в заголовках этих таблиц.

Таблица 6.3 – Коэффициенты корреляции словарей именных групп текстов сообществ графа G_4.

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	−0,685	1						
S_2	−0,683	0,476	1					
S_3	−0,556	0,086	0,029	1				
S_4	−0,699	0,209	0,062	−0,131	1			
S_5	−0,684	0,874	0,443	0,084	0,274	1		
S_6	−0,681	0,162	0,108	−0,08	−0,001	0,218	1	
S_7	−0,652	0,45	0,321	0,239	0,225	0,505	0,139	1

Таблица 6.4 – Коэффициенты корреляции словарей глагольных групп текстов сообществ графа G_4.

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	−0,898	1						
S_2	−0,852	0,424	1					
S_3	0,376	−0,56	0,172	1				
S_4	−0,19	−0,427	−0,558	−0,309	1			
S_5	−0,9	0,716	0,412	−0,595	−0,365	1		
S_6	0,297	−0,472	0,185	−0,798	−0,349	−0,441	1	
S_7	−0,724	0,456	0,148	0,089	0,38	0,503	0,11	1

Таблица 6.5 – Коэффициенты корреляции словарей именных групп текстов сообществ графа G_5.

	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_0	1							
S_1	−0,574	1						
S_2	−0,488	−0,406	1					
S_3	−0,598	−0,446	−0,373	1				
S_4	−0,609	−0,591	−0,107	−0,539	1			
S_5	−0,53	−0,378	−0,414	−0,495	−0,553	1		
S_6	−0,483	−0,399	−0,403	−0,466	−0,462	0,14	1	
S_7	−0,572	−0,351	−0,147	−0,35	−0,503	−0,28	−0,32	1

При сравнении по коэффициенту попарной ранговой корреляции частотных словарей буквосочетаний видно сильное совпадение частотных распределений для буквосочетаний длиной до 3, что подтверждает утверждение о том, что данные буквосочетания характеризуют язык (все наборы текстов на русском языке). Наблюдаемая согласованность частотных словарей буквосочетаний длиной 4 и 5, псевдооснов для наборов текстов разных сообществ указывает на близость текстов разных сообществ по содержательной направленности, поскольку эти характеристики определяют в первую очередь тематику текстов.

Сравнение частотных словарей словосочетаний (именных и глагольных групп) показывает наличие возможных различий между частотными словарями словосочетаний текстов разных сообществ.

Для текстов сообществ графа G_4 сравнение словарей именных и глагольных групп (таблицы 6.3 и 6.4) показывает возможность разделить наборы текстов большинства сообществ. Это указывает на различие текстов по их психологической направленности и направленности действия, следовательно, и на качество выделения сообществ на графе G_4. Наиболее яркие результаты разделения текстов сообществ по именным группам показаны в таблице 6.5. для графа G_5. Словари именных групп попарно «обратны» по частотам использования словосочетаний в текстах. Это указывает на возможность выделения наиболее часто используемых именных групп в наборах текстов разных сообществ, а также подтверждает качество полученного выделения сообществ на графе G_5.

В целом результаты сравнения, представленные в таблицах 6.3, 6.4 и 6.5 показывают возможности использования рангового анализа словарей словосочетаний как метод оценки качества выделения сообществ на графе.

6.3 Статистические характеристики текстов

Для анализа текстов рассмотрим некоторые их статические показатели как психолингвистические характеристики, которые могут описывать особенности акторов в неявных сообществах.

Вычисление таких показателей основано на грамматических характеристиках отдельных словоупотреблений. Выделим следующие статистические показатели. Во-первых, это 3 показателя, определяющих общие структурные характеристики текстов, а именно:

Средняя длина словоупотреблений в символах

Средняя длина предложения в словоупотреблениях.

Отношение числа знаков препинания к общему количеству словоупотреблений.

Во-вторых, это 12 показателей, указывающих на лексическое разнообразие текстов:

1) Коэффициент лексического разнообразия 1 (ЛР1) – отношение числа уникальных лексем к числу словоупотреблений.
2) Коэффициент лексического разнообразия 2 (ЛР2) – коэффициент разнообразия по псевдоосновам – отношение числа уникальных псевдооснов к числу словоупотреблений.
3) Отношение числа местоимений к числу словоупотреблений.
4) Отношение числа наречий к числу словоупотреблений.
5) Отношение числа прилагательные к числу словоупотреблений.
6) Коэффициент глагольности (КГ) – отношение количества глаголов и глагольных форм (причастий и деепричастий) к общему количеству всех словоупотреблений.
7) Коэффициент действия 1 (КД1) – отношение количества глаголов (деепричастия и причастия исключаются) к количеству прилагательных.
8) Коэффициент действия 2 (КД2) – отношение количества глаголов и глагольных форм (деепричастий и причастий) к количеству прилагательных.
9) Коэффициент опредмеченности действия (КОД) – отношение количества глаголов (деепричастия и причастия исключаются) к количеству существительных.

10) Коэффициент логической связности 1 (ЛС1) – отношение общего количества служебных слов (союзов и предлогов) к общему количеству предложений.
11) Коэффициент логической связности 2 (ЛС2) – коэффициент использования служебных слов – отношение общего количества служебных слов (союзов и предлогов) к общему количеству словоупотреблений.
12) Коэффициент связности лексики (СЛ) – отношение числа существительных и глаголов (деепричастия и причастия исключаются) к количеству прилагательных и наречий.

В-третьих, это 8 показателей, указывающих на использование синтаксических связей в словосочетаниях:

1) Средняя длина именных групп в словоупотреблениях.
2) Отношение числа именных групп к числу словоупотреблений.
3) Среднее отношение числа именных групп к длине предложения в словоупотреблениях.
4) Среднее количество числа «подгрупп» в одной именной группе.
5) Средняя длина глагольных групп в словоупотреблениях.
6) Отношение числа глагольных групп к числу словоупотреблений.
7) Среднее отношение числа глагольных групп к длине предложения в словоупотреблениях.
8) Среднее количество числа «подгрупп» в одной глагольной группе.

Таким образом, выделены 23 статистические лингвистические характеристики, которые можно рассматривать как предполагаемые факторы, характеризующие тематическую, психолингвистическую, социальную направленности текстов.

Подсчет статистических характеристик текстов сообществ и выявление различий используется для оценки качества выделенных на графе сообществ. На основе проведенных экспериментов из этих 23 характеристик выделены 9 наиболее

подходящих для сравнения психолингвистических особенностей текстов сообществ:

- Коэффициент лексического разнообразия 1 (ЛР1);
- Коэффициент лексического разнообразия 2 (ЛР2);
- Коэффициент глагольности (КГ);
- Коэффициент действия 1 (КД1);
- Коэффициент действия 2 (КД2);
- Коэффициент опредмеченности действия (КОД);
- Коэффициент логической связности 1 (ЛС1);
- Коэффициент логической связности 2 (ЛС2);
- Коэффициент связности лексики (СЛ).

Примеры применения статистических характеристик текстов с целью подтверждения качества выделения сообществ на графе представлены в главе 4, разделе 4.3 и в главе 5, разделе 5.3.

6.4 Исследование субъектности неявных сообществ

Для междисциплинарных исследований социальных сетей и сетей обмена сообщениями актуальным является изучение групповых процессов в динамике. Качество выделения сообществ на графах взаимодействующих объектов для таких сетей может быть дополнительно оценено с помощью такого понятия из психологи как субъектность. В данном случае речь пойдет об оценке субъектности выделенных сообществ. Одной из гипотез тут является предположение о последовательном возрастании субъектности при повышении значения для отношения числа ребер к числу вершин графа [112].

В данном разделе на основе примера описана методика по изучению выделяемых на графе взаимодействующих объектов, полученном при импорте данных на

разных временных интервалах из сети *Twitter* по модели (2.4). С этой целью в рамках данной модели для временного параметра t были взяты следующие значения $t_1 = 1$ час; $t_2 = 12$ часов; $t_3 = 24$ часа после момента публикации поста, от которого строится множество V_0. Рассмотрим развитие групп общения на примере трех постов, каждый из которых за сутки концентрировал вокруг себя несколько десятков акторов. Обозначим рассматриваемые далее посты как A, B и C.

Для каждого поста по трем временным отсечкам было построено по три графа в соответствии с моделью (2.4). Будем обозначать их далее, как G_t^p, где p – обозначает пост, а t – временной промежуток скачивания графа для этого поста. Например, для поста A получается следующая тройка графов:

G_1^A – граф для поста A через 1 час после поста;

G_{12}^A – граф для поста A через 12 час после поста;

G_{24}^A – граф для поста A через 24 час после поста.

Число вершин и ребер в графе G_t^p обозначим за $n_{G_t^p}$ и $m_{G_t^p}$ соответственно. Множество сообществ, выделенных на графе G_t^p обозначим как $S_{G_t^p} = \left\{S_{G_t^p}^i\right\}$. Количество выделенных сообществ тогда будет равно $\left|S_{G_{24}^p}\right|$. Тогда число вершин и ребер в сообществе $S_{G_t^p}^i$ обозначим за $n_{S_{G_t^p}^i}$ и $m_{S_{G_t^p}^i}$. Сводные данные по всем трем графам приведены в таблице 6.6.

Для этих графов выделение сообществ производилось с помощью алгоритмов, описанных в главе 3. Как легко видеть, для данного поста на первом временном отрезке можно ожидаемо выделить большую группу пользователей, связанных с автором поста, это большое сообщество с вершинами голубого цвета. Данное сообщество состоит из 31 вершины, включая самого автора исходного поста. Другие два выделенных сообщества имеют размеры 3 и 4 соответственно. На следующем временном отрезке происходит существенный рост числа вершин и ребер графа, что приводит к формированию новых крупных сообществ.

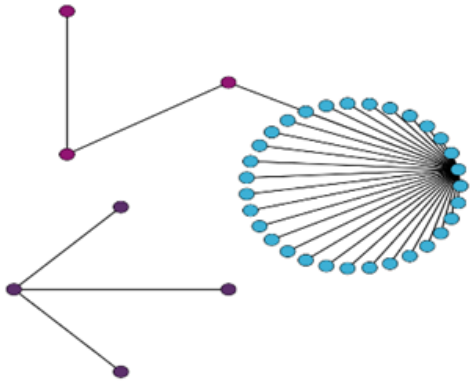

Рисунок 6.1 – Граф G_1^A с выделенными на нем сообществами

Граф G_{12}^A представлен на рисунке 6.2, на нем выделено 6 сообществ, имеющих следующие размеры: 43-20-4-3-3-2. На графе G_{24}^A выделяется уже 9 сообществ с размерами 41-15-8-5-4-4-3-2-2 (рисунок 6.3). Таким образом, для графа G_{24}^A, получается выделить сообщества, состоящие из 2, 3 или 4 вершин, обладающие в силу этого высокой плотностью.

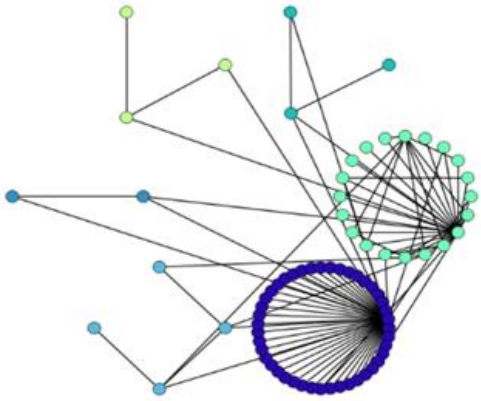

Рисунок 6.2 – Граф G_{12}^A с выделенными на нем сообществами

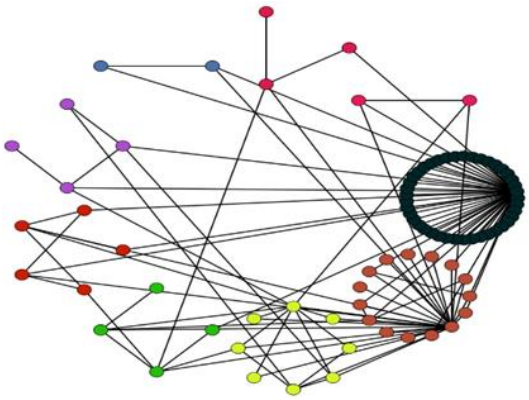

Рисунок 6.3 – Граф G_{24}^A с выделенными на нем сообществами

Если перейти к графам поста B, то для G_1^B, состоящего из 14 вершин и 11 ребер выделяются 4 сообщества, их размеры: 8-2-2-2 (рисунок 6.4). Граф G_{12}^B состоит из 28 вершин и 40 ребер, тут выделяются 5 сообществ (рисунок 6.5) следующих размеров: 12-6-4-4-2.

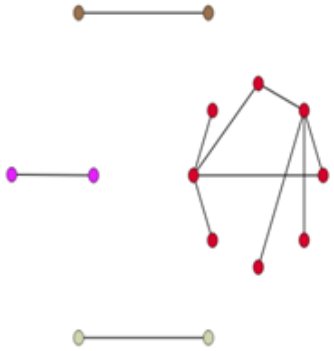

Рисунок 6.4 – Граф G_1^B с выделенными на нем сообществами

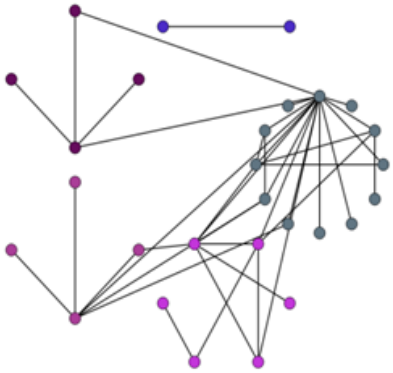

Рисунок 6.5 – Граф G_{12}^B с выделенными на нем сообществами

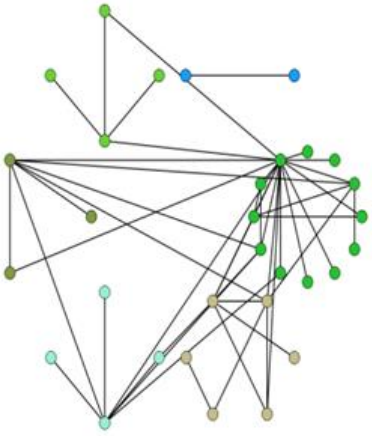

Рисунок 6.6 – Граф G_{24}^B с выделенными на нем сообществами

Для графа G_{24}^{B} из 31 вершины и 48 ребер получаем 6 сообществ следующих размеров: 12-6-4-4-3-2 (рисунок 6.6). Характерной чертой графов для поста B является то, что вершина, соответствующая исходному посту, при этом имеет меньше смежных вершин, чем другая вершина графа, у которой в графе G_{24}^{B} имеется 18 смежных вершин из 30 возможных.

Для графа G_{1}^{C} получаем 3 сообщества следующих размеров: 15-3-2 (рисунок 6.7). На графе G_{12}^{C} выделяются уже 7 сообществ: 92-13-9-9-2-2-2 (рисунок 6.8). Для взвешенного графа G_{24}^{C} получаем 6 сообществ размеров: 153-22-17-3-2-2 (рисунок 6.9). Переход от G_{12}^{C} к G_{24}^{C} характеризуется тем, что основное сообщество-ядро «оттягивает» на себя часть вершин, что приводит к частичному распаду второго по размерам сообщества.

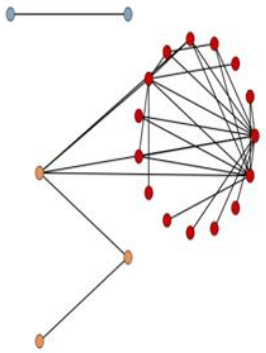

Рисунок 6.7 – Граф G_{1}^{C} с выделенными на нем сообществами

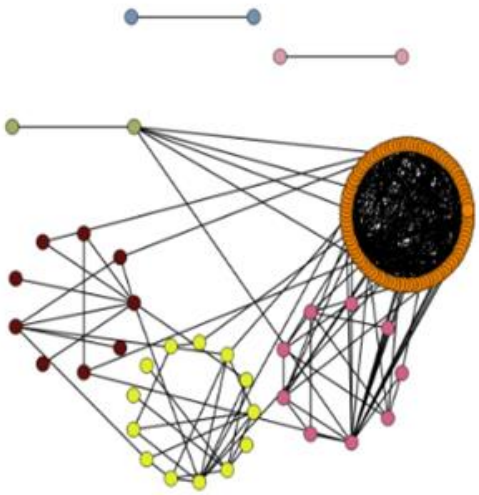

Рисунок 6.8 – На графе G_{12}^{C} выделено 7 сообществ

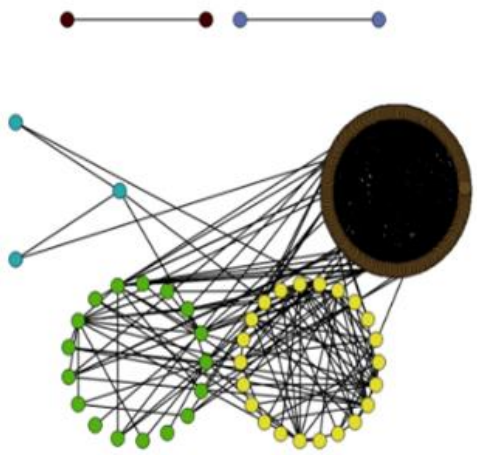

Рисунок 6.9 – Граф G_{24}^{C} – поглощение части вершин «ядром»

Таким образом, для графов, получаемых при скачивании почти сразу (1 час) или через непродолжительное время (12 часов) после публикации поста, характерен процесс набора пользователей, осуществляющих активное взаимодействие с постом, что влечет за собой процесс формирования сообществ. Для графов, получаемых при скачивании по истечению существенного времени после публикации поста (24 часа), как правило, уже достигается состояние стабилизации и формирование самого большого сообщества.

Ключевой психологической характеристикой сообщества как социальной группы является субъектность. Субъектность как самодетерминированная и самопроизвольная активность на уровне социальной группы проявляется в различных формах совместной активности: совместная деятельность, внутригрупповое взаимодействие, групповое поведение и групповое самопознание. Наиболее полно феномен субъектности на уровне групп раскрывается через анализ коллективного субъекта и такие его атрибуты как взаимосвязанность и взаимозависимость, совместная активность [112]. Одним из подходов к исследованию субъектности сетевых сообществ является дискурсивная парадигма, опирающаяся на выделение дискурсивных маркеров в текстах этих сообществ [126, 127] и оценке относительной частоты их встречаемости в соответствии с моделями их проявления [128].

В качестве методики оценки качества выделения неявных сообществ проведем их анализ на полученных графах с учетом соответствующих данных по сообщениям заданной психологической направленности [65, 112]. Для оценки субъектности сообществ трех исследуемых графов экспертами из Института психологии РАН проводился психолингвистический анализ: соответствующие им тексты размечались 3 экспертами-психолингвистами путём выделения дискурсивных маркеров [127, 128, 129]. Субъектность рассчитывалась как относительная частота дискурсивных маркеров, обнаруженных в текстах сообществ. В качестве единицы анализа использовались комментарии к посту. Для оценки субъектности использовались: показатели субъектности первого уровня – языковая и понятийная идентификация, готовность действовать, поддержка тематики сетевого сообщества, групповые нормы и ценности, отстранение «других», позитивная поддержка коммуникации, защита целостности сообщества, гражданская идентичность; два показателя субъектности второго уровня – обсуждение совместной деятельности, «Свои»-«чужие»; и показатель общей субъектности как среднее по всем показателям.

Для всех трех постов необходимо отметить следующую схожую черту полученных графов. При разбиении на сообщества среди них выделяется по одному основному сообществу, содержащему ключевое число пользователей. Если посмотреть подробнее на результаты разбиения на графах, полученных для $t = 24$ часа после поста, то для поста C имеет место больший перекос в сторону основного сообщества, что заметно из таблицы 6.6. Размер максимального сообщества $|S_{max}(G)|$, отнесенный к общему числу вершин для каждого из постов виден из этой же таблицы.

В ходе сопоставления сетевых характеристик и показателей субъектности [110] было выявлено, что общая субъектность значимо связана с отношением $\frac{m_{S^i_{G^p_t}}}{n_{S^i_{G^p_t}}}$. А именно, чем больше ребер между вершинами, входящими в выделенное сообщество выявлено, тем больше в контенте этих сообществ обнаруживается маркеров, связанных с определенными характеристиками субъектности.

Таблица 6.6 – Структура анализируемых групп общения для трех рассматриваемых графов.

	$\left\|S_{G_1^p}\right\|$	$n_{G_1^p}$	$m_{G_1^p}$	$\left\{S^i_{G_1^p}\right\}$	$\left\|S_{G_{12}^p}\right\|$	$n_{G_{12}^p}$	$m_{G_{12}^p}$	$\left\{S^i_{G_{12}^p}\right\}$	$\left\|S_{G_{24}^p}\right\|$	$n_{G_{24}^p}$	$m_{G_{24}^p}$	$\left\{S^i_{G_{24}^p}\right\}$	$\frac{\left\|S_{max}(G_{24}^p)\right\|}{n_{G_{24}^p}}$
G_t^A	3	38	33	31-4-3	6	75	100	43-20-4-3-3-2	9	84	124	41-15-8-5-4-4-3-2-2	41/84
G_t^B	4	14	11	8-2-2-2	5	28	40	12-6-4-4-2	6	31	48	12-6-4-4-3-2	12/31
G_t^C	3	20	33	15-3-2	7	129	414	92-13-9-9-2-2-2	6	199	890	153-22-17-3-2-2	153/199

Таблица 6.7 – Изменение значений $\frac{m_{G_t^p}}{n_{G_t^p}}$ от времени

	$\frac{m_{G_1^p}}{n_{G_1^p}}$	$\frac{m_{G_{12}^p}}{n_{G_{12}^p}}$	$\frac{m_{G_{24}^p}}{n_{G_{24}^p}}$	**Комментарий**
G_t^A	$\frac{33}{38}$	$\frac{100}{75}$	$\frac{124}{84}$	$\frac{m_{G_{24}^p}}{n_{G_{24}^p}} > \frac{m_{G_{12}^p}}{n_{G_{12}^p}} > \frac{m_{G_1^p}}{n_{G_1^p}}$
G_t^B	$\frac{11}{14}$	$\frac{40}{28}$	$\frac{48}{31}$	
G_t^C	$\frac{33}{20}$	$\frac{414}{129}$	$\frac{890}{199}$	

В случае изученных сообществ это вполне определенные характеристики: групповые нормы и ценности, защита целостности сообщества и гражданская идентичность. Это контент, в котором осуждаются и/или формулируются коммуникативные нормы и групповые ценности; осуществляется защита границ и целостности сообщества через выражение негативного отношения к собеседникам и возбуждение недоверия и враждебности к «чужим» собеседникам; обсуждаются проблемы социальной защищенности, гражданской идентичности и проявления активной жизненной позиции.

Наличие значимой корреляции между уровнем некоторых компонентов субъектности и $\frac{m_{G_t^p}}{n_{G_t^p}}$ позволяет сформулировать предположение о последовательном возрастании субъектности в связи с повышением значения этого коэффициента (см. таблицу 6.7). Проверка сформулированной гипотезы проводилась психологами Института психологии РАН и показала [112] значимость сдвигов по показателям дискурсивных маркеров субъектности в контенте сетевых сообществ в зависимости от коэффициента $\frac{m_{G_t^p}}{n_{G_t^p}}$. С ростом значения коэффициента последовательно возрастают такие показатели: Общая субъектность, «Свои»-«чужие», Языковая и понятийная идентификация и Отстранение «других». Коэффициент взаимодействия графа значимо коррелирует с частотой определенных дискурсивных маркеров субъектности: «Общая субъектность», «Групповые нормы и ценности», «Защита целостности сообщества» и «Гражданская идентичность».

Таким образом, рост показателей субъектности подтверждает качество выделения сообществ на графе.

6.5 Выводы по главе 6

1. Предложенная в данной главе методика для оценки эффективности выделения сообществ на графе основана на обработке текстовых атрибутов их вершин и

производится с использованием алгоритмов компьютерной лингвистики и психолингвистических факторов. Обработка объединенных массивов текстов выделенных сообществ позволяет оценить разделение на сообщества следующими методами компьютерной лингвистики:

- Попарным сравнением частотных словарей различных лингвистических характеристик, составленных для наборов текстов каждого сообщества;
- Сравнением наборов статистических (психолингвистических) характеристик текстов, описывающих различное поведение участников неявных сообществ.

2. Представленные экспериментальные результаты применения такого подхода подтверждают его применимость для оценки качества выделенных сообществ на графе.

3. Проведенный для построенных по модели (2.4) графов анализ оценки взаимодействия пользователей и продемонстрированный хронологический рост субъектности формирующихся сообществ могут быть использованы как одно из средств для подтверждения качества их выделения.

4. Основные результаты, представленные в главе 6, опубликованы в следующих работах: [65, 112, 113, 116, 117, 118, 119, 120, 121]. В работах [113, 116, 117, 118, 119, 120, 121] соискателю принадлежит метод оценки корректности для полученного разбиения графа на сообщества. Данный метод основан на алгоритмах компьютерной лингвистики. В работах [65, 112] соискателю принадлежит применение методов анализа графов взаимодействующих объектов, полученных при импорте данных из социальных сетей для формирования психологических показателей социального взаимодействия.

ГЛАВА 7 ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ АНАЛИЗА ГРАФОВ ВЗАИМОДЕЙСТВУЮЩИХ ОБЪЕКТОВ

В данной главе представляется созданное программное обеспечение для хранения, визуализации и анализа графов, реализующее описанные в предыдущих главах методы и алгоритмы. Разработанный программный комплекс и алгоритмы визуализации описаны в [62, 73, 130], хранение и алгоритмы работы с графами – в [73, 75, 131, 132, 133, 134, 135, 136]. Вклад автора заключается в разработке архитектуры программного комплекса и используемых структур данных, методов хранения и алгоритмов.

7.1 Программное обеспечение для анализа графов

Существует достаточно много программных продуктов, предназначенных для аналитической работы и визуализации сетевых структур. Такое программное обеспечение позволяет отображать результаты исследований в виде схем и диаграмм, что дает возможность аналитикам проводить визуальную работу с большим массивом данных. Профессиональные сферы, в которых может быть применен спектр подобных информационных систем широк [24, 137], и включают в себя как использование инструментария спецслужбами, так и научные исследования, коммерческие цели. Многие программные продукты (*Cytoscape* [138], *Tulip* [139], *VisuaLyze*r [140]) создавались как биомедицинские, но в дальнейшем переориентировались на анализ социальных сетей.

Выделяются крупные промышленные продукты, получившие признание у спецслужб разных стран мира: *i2 Analyst's Notebook* [141], *CrimeLink* [142] и *Xanalys Link Explorer* [143]. Эти программные комплексы предназначены в первую очередь для проведения криминальных расследований и содержат стандартные ме-

тоды визуализации графов. Как правило, эти продукты требуется дополнять встраиванием отдельных процедур, реализующих специальные средства визуализации и анализ структуры графов.

IBM i2 Enterprise Insight Analysis [141] – комплекс аналитического программного обеспечения для анализа информации, ориентированный на обеспечение следственной деятельности, включающий визуальную аналитическую среду *i2 Analyst's Notebook* [141]. Применяется для визуализации и исследования динамики процессов и событий. Широко применяется для задач финансовых разведок мира, в том числе в сфере противодействия отмывочной деятельности и финансированию терроризма. Система содержит набор классических средств автоматического размещения графов.

CrimeLink Explorer [142] – информационная система, предназначенная для извлечения и поиска связей между людьми из больших наборов данных. Позволяет строить сетевые схемы расследования и вести расследования преступлений. Предусматривает проведение автоматизированного анализа связей на основе нахождения кратчайшего пути и оценки важности связей.

Xanalys Link Explorer [143] – программный комплекс, предназначенный для составления сетевых схем расследований самых различных действий и преступлений. Создается гибкая модель безопасности на основе ролей; имеется управление уровнями доступа к информации и ролями отдельных пользователей. Содержит набор средств, позволяющих проводить визуальный анализ.

К средствам расследования можно отнести встраиваемые в программные продукты *Графовый анализатор Group-IB* [144] и программу *Sentinel Visualizer* [145], которая содержит элементы анализа структуры графа.

Графовый анализатор Group-IB [144] предоставляет процедуры графового анализа сетевой инфраструктуры. Является внутренним инструментом, который встраивается во все публичные продукты компании. Предназначен для автоматического построения и визуализации графов сетевого взаимодействия по ip-адресам с целью выявления фишинговых атак. Строится граф по вредоносному домену, который показывает связи с другими вредоносными доменами, атрибутирует это до

группы и показывает какие файлы использовались в кибератаке. Данные разработки не используют анализ структуры графа и предназначены в основном для простейшей визуальной обработки аналитиками в области информационной безопасности.

Sentinel Visualizer [145] – программный комплекс визуализации и анализа больших данных, позволяющий устанавливать многоуровневые связи между сущностями и моделировать различные типы отношений. Используемые характеристики социальных сетей позволяют выявлять наиболее интересные взаимодействия веб-сайтов. Реализована расширенная фильтрация, анализ кратчайшего пути. Богатый набор метрик позволяет решать задачи выявления центральности объекта в сети, исследования потока информации в сети.

Также надо отметить ряд схожих по своим возможностям программных продуктов, ориентированных на визуализацию графов социальных сетей: *Gephi* [146] и *VisuaLyzer* [140], *NetMiner* [147], *Cytoscape* [138], *Tulip* [139], *yEd* [148]. Две первые из указанных информационных систем содержат процедуры для анализа структуры графа. Так, *Gephi* – программный продукт с открытым исходным кодом и широкой функциональностью. В числе прочего инструментария присутствует подсчет некоторых центральностей для вершин. Существуют плагины, в которых реализованы процедуры выделения сообществ для невзвешенных графов методом «распространения меток», методом случайных блужданий, алгоритмом Гирвана-Ньюмана. *NetMiner* [147] – коммерческий продукт, позволяющий создавать карту сети в трех измерениях для семантического графа, составленного по результатам работы сборщика данных социальных сетей, лингвистического анализа корпусов текстов. Содержит подсчет центральности, коэффициента кластеризации.

Cytoscape [138] – бесплатная платформа для визуализации и анализа графов. Сочетает стандартный набор функций для работы с графами и дополнительные возможности посредством написания пользовательских приложений и взаимодействия с ними по *API.*

Tulip [139] – программа для анализа и визуализации графов, предназначенная для решения задач биоинформатики и проблем идентификации геномов в биологии. Содержит различные варианты автоматических размещений и инструменты анализа для подсчета характеристик центральности и иерархической кластеризации.

yEd [148] – инструмент для создания и визуализации графов и диаграмм. Предназначен для создания и редактирования организационных схем, UML-диаграмм, блок-схем. Реализовано разбиение графа на группы с использованием рёберной центральности между ними. Группы обнаруживаются постепенным удалением из графа ребра с самой высокой центральностью.

Некоторые разработки предназначены для исследований в области высоких технологий, включая анализ программного кода с целью оптимизации программ или их трансформации: *Графоанализатор* [149], *Visual Graph* [150]. Для анализа работы программ (инциденты с точки зрения безопасности) создано, например, *aiSee (Graph Visualization)* [151]. Предназначение этих продуктов практически ограничивают их функциональность по анализу структуры графа, что не требует, в частности, выявления неявных сообществ.

Имеются так же популярные наборы библиотек для создания интерактивных визуализаций графов и работы с данными: *Tom Sawyer Software* [152], *Graphviz* [153] и *Igraph* [154]. Такие библиотеки отличаются по своей функциональности, если библиотека *Tom Sawyer Software* [152] ориентирована главным образом на решение задач визуализации графов, то библиотека *Igraph* [154] содержит большой набор процедур для анализа структуры графа.

Tom Sawyer Software [152] является набором библиотек, позволяющим создавать приложения для визуализации графов больших размеров. Интегрированные интерфейсы проектирования и предварительного просмотра, а также обширные библиотеки прикладного программного интерфейса (API) позволяют разработчикам быстро создавать пользовательские приложения, которые интуитивно решают проблемы визуализации больших данных.

Graphviz [153] – это открытая библиотека и инструмент для визуализации графов и сетей. Она позволяет создавать графические представления структур данных, таких как деревья, графы и сети. Библиотека содержит инструментарий в том числе для автоматического кругового размещения графа.

Igraph [154] представляет собой библиотеку для анализа графов, доступную для различных языков программирования, в первую очередь – на Python и R. Кроме средств визуализации библиотека содержит реализации довольно большого количества алгоритмов для исследования графов. Данная библиотека представляет особый интерес в связи с большим количеством процедур, содержащих различные варианты алгоритмов анализа структуры графа и нахождения сообществ в структуре графа. Представлены алгоритмы, основанные на спектральных свойствах графа, алгоритмы, основанные на оценке показателя «модулярность» Ньюмана-Гирвана.

Резюмируя приведенный в этом разделе обзор существующего программного обеспечения, стоит отметить такой важный недостаток как отсутствие у них специализированных графовых хранилищ, ориентированных на работу с большими графами. Также существенным недостатком является, как правило, ограниченный функционал по анализу структуру графов, включая выделение неявных сообществ и подсчет ключевых характеристик.

7.2 Архитектура программного комплекса

Для работы с графами взаимодействующих объектов, в том числе их визуализации, применения алгоритмов выделения сообществ разработан программный комплекс *AVS* (*Analytics and Visualization System for graphs*), который реализован на языке *C++* (рисунок 7.1).

Графические интерфейсы пользователя и средства представления графов на сцене созданы на основе библиотеки *Qt*. [155], и поддерживают функционал обработки действий пользователя системы на основе возникающих событий, заложенных в библиотеке.

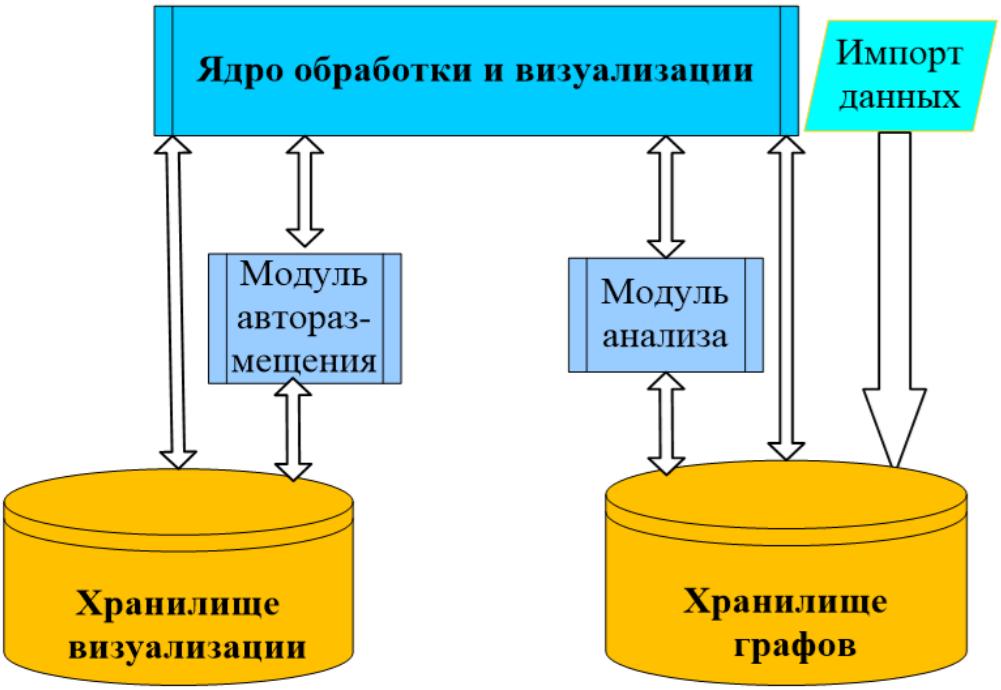

Рисунок 7.1 – Архитектура *Analytics and Visualization System for graphs*

Помимо авторазмещения и обработки событий *AVS* представляет пользователю возможности взаимодействия с графом, в том числе донастройки свойств его ребер и вершин, корректировки значений атрибутов. В рамках визуализации на основе атрибутов ребра могут быть скрыты от пользователя.

Программный комплекс предусматривает как создание небольших графов в «ручном режиме», так и загрузку графов больших размеров, полученных при импорте из коммуникационных социальных сетей. Получены два свидетельства о регистрации программ для ЭВМ (приложение 2).

Используются разработанные программы импорта данных из социальных сетей и сетей мгновенного обмена сообщениями для формирования графа взаимодействующих объектов (глава 2). Полученные графы загружаются в «Хранилище графов» (раздел 7.4). В хранилище могут быть загружены и другие графы в формате *AVS* (приложение 1).

Хранение графа разделено на две части: «*Хранилище графа*» и «*Хранилище визуализации*». В *Хранилище графа* используется специализированное хранилище, в котором реализовано эффективное сжатие данных, описанное в разделе 7.4, и базовые алгоритмы работы с графами. Ядро обработки и визуализации Приложения

позволяет работать с графом, загруженным в оперативную память. Функционал хранилища позволяет получать структуры и сохранять их в универсальный формат, используемый программным комплексом (описан в приложении 1).

Реализованы алгоритмы выделения неявных сообществ на графе и их визуализации на сцене.

Модуль авторазмещения содержит реализованную на языке C++ библиотеку алгоритмов автоматического размещения графа на плоскости. Базовым методом для визуализации является метод кругового размещения вершин, принадлежащих одному сообществу. Пример такого размещения представлен на рисунке 7.2.

Модуль анализа содержит реализованную на языке *C++* библиотеку алгоритмов выделения сообществ, описанных в главах 3, 4 и 5.

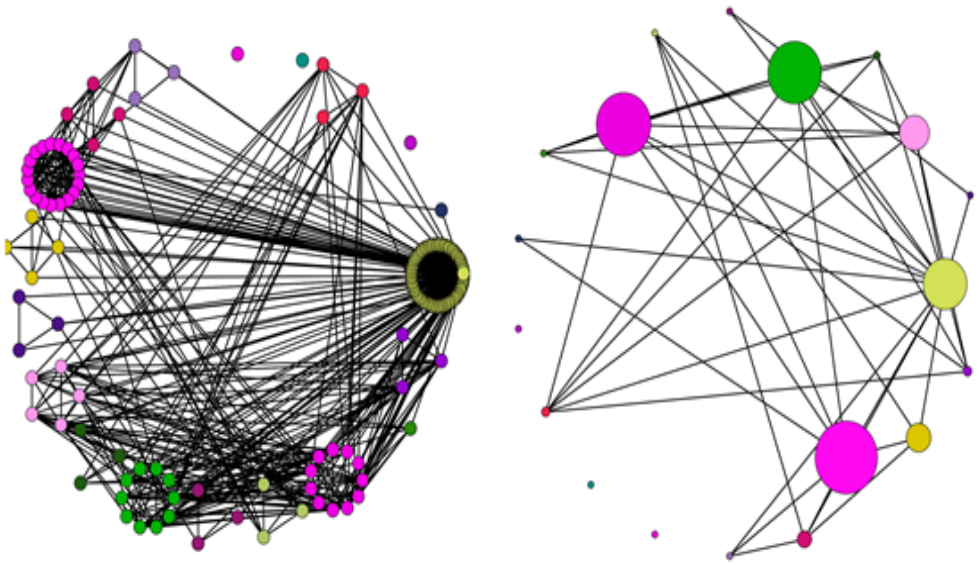

Рисунок 7.2 – Размещение сообществ и представление мета-вершин

В разработанном программном обеспечении реализуются две группы методов анализа графов взаимодействия объектов:

- Математические методы анализа структуры графа, включающие выделение неявных сообществ на графе.
- Визуальное представление графа для возможностей представления структуры и визуального анализа взаимодействий.

В приложении *AVS* в качестве библиотек (Модуль анализа) используются алгоритмы разбиения графа на неявные сообщества. По результатам работы алгоритмов в качестве результата создаются файлы в папке проекта с соответствующими наборами идентификаторов вершин графа, отнесенных к выделенным сообществам. После выделения сообществ выполняется параллельный проход по всем вершинам графа с изменение их цвета и атрибутов, отвечающих за сообщества. Базовым средством визуализации сообществ является круговое авторазмещение элементов (рисунок 7.2).

Для удобства визуального анализа графов реализован функционал по представлению сообщества как вершины нового мета-графа, что удобно для рассмотрения взаимодействия групп вершин между собой и используется в алгоритмах выделения сообществ. На правой схеме рисунка 7.2 продемонстрирован мета-граф, получаемый из графа с левой схемы. Реализована возможность итеративного выделения сообществ в мета-графе и просмотра содержащихся в мета-вершине элементов (вершин, ребер, сообществ).

В интерфейсе *AVS* реализована возможность определения (и переопределения) пользователем весов на ребрах графа и их пересечете. Соответствующие данные хранятся в конфигурационном файле. Обработка этих правил позволяет создать атрибут, соответствующий весу при отсутствии такого в исходных данных.

Реализован подсчет разных показателей, характеризующих структуру графа и отдельных его элементов – вершин и ребер, включая набор стандартных центральностей.

7.3 Проблема хранения графов

В последние годы имеет место развитие специализированных баз данных, позволяющих осуществлять обработку и хранение графов [156, 157, 158]. Такие задачи имеют широкую область применения в различных отраслях знаний. Поддержка

сложных операций, актуальных для соответствующей предметной области, является ключевым их преимуществом по отношению к реляционным базам данных

Отметим, для хранения графов используются и неспециализированные системы баз данных, в первую очередь реляционные СУБД, такие как широко известные *MS SQL Server*, *Oracle* и популярные *PostgreSQL* [158], *MySQL* [160]. В рамках реляционной модели процедура хранения данных приводит к увеличению числа данных (структура базы данных), что осложняет возможности добиваться хорошего быстродействия.

Проявляется интерес в применении для работы с графами нереляционных хранилищ [156, 157, 161]. Типичные *NoSQL*-модели применяются для документных баз данных, для которых важно хранить документы (текстовые и графические данные) в принадлежности к конкретным коллекциям (именованным множествам) документов и получать быстрый доступ к ним через структуры, которые называются «наборы реплик». Популярной является *Oracle NoSQL Database* [156, 157], так же популярна и документо-ориентированная СУБД с открытым исходным кодом *MongoDB* [162, 163, 164].

Наибольшее количество *NoSQL*-баз данных [156, 157] хранят наборы не связанных между собой пар «ключ-значение». Такая особенность модели ограничивает ее использование для хранения и обработки графов и других взаимосвязанных данных: не поддерживается согласованность данных, невозможно выполнять сложные запросы, реализовывать некоторые операции, аналогичные операциям реляционной алгебры (операцию JOIN, например). В такой модели приходится вводить дополнительные агрегирующие идентификаторы, что приводит к потере производительности соответствующих приложений [156].

Системы хранения графов должны обеспечивать высокую скорость выполнения операций, что требует применения специализированных баз данных. Такие системы зачастую предусматривают уникальных средств запросов. Обзор и оценку известных баз данных, предназначенных для работы с такими данными можно найти в [158, 165, 166, 167]. Ключевые их различия связаны с методами поддержки реализации алгоритмов на графах и хранением данных.

Благодаря уровню гибкости и менее строгой структуре данных достигается обработка значительных объемов информации, недоступных системам, основанным на реляционных базах данных [157, 165, 167]. К наиболее значимым граф-ориентированным СУБД можно отнести *OrientDB* [157, 168, 169] и *NEO4J* [170, 171, 172]. Вторая из них – это граф-ориентированная СУБД с открытым исходным кодом, написанная на *Java* [166] и имеющая собственный формат хранения графов. При этом *OrientDB* является одновременно и документо-ориентированной, и графовой СУБД с открытым исходным кодом и широкими возможностями API.

Отметим еще *ArangoDB* [173] – СУБД с открытым исходным кодом, которая поддерживает как графы, так и документы. *Sparksee* (ранее – *DEX*) [174] – графовая СУБД, хранящая маркированный направленный мультиграф с атрибутами, которые не индексируются.

Высокая степень изменчивости и частое обновление характерны для графов взаимодействующих объектов, построенных при импорте данных из коммуникационных сетей, в том числе из сетей мгновенного обмена сообщениями и из социальных сетей. Поэтому для доступа к актуальным данным требуется часто перестраивать хранилище данных. При этом структура данных, которая будет оптимальной для выполнения поисковых операций, не обеспечивает скорость пополнения хранилища.

Стандартные механизмы сжатия, такие как представленные в [175], обычно применяются в базах данных общего назначения. Однако для графовых структур с конкретной организацией атрибутов ребер существуют более эффективные методы сжатия данных. Применение этих методов позволяет увеличить плотность хранения данных более чем вдвое.

Возможность более детально учитывать специфику происхождения данных позволяет достичь этого результата. Как правило, алгоритмы, учитывающие особенности сжимаемого материала, показывают значительно лучшие показатели по степени сжатия, чем широко применяемые алгоритмы общего назначения, что демонстрируют различные тесты [176, 177].

К основным недостаткам таких алгоритмов относятся невысокая скорость работы и повышенные требования к объему потребляемой памяти. Следует подчеркнуть, что показатели таких алгоритмов сильно зависят от размера сжимаемых данных — небольшие порции могут приводить к снижению скорости сжатия и его эффективности.

7.4 Хранилище графов

При проектировании хранилищ графовых структур актуальной является их реализация с соответствующей организацией компактного хранения данных. Реализация этой задачи возможна за счет учета специализированных свойств, характерных для определенного типа графов, под которые и предназначено хранилище. Далее в разделе 7.4. описано решение для графов взаимодействующих объектов, полученных из реальных коммуникационных сетей [131, 132, 135]. Детальнее описание операций, возможных режимов работы хранилища и процессов обработки данных даны в монографии автора [73].

7.4.1 Задача хранения графа

Рассматриваем граф $G(V,E)$, у вершин и ребер которого согласно описанному в главе 2 могут присутствовать определенные атрибуты (характеристики). Будем обозначать за $v_1(e)$ и $v_2(e)$ вершины, которые соединяет ребро e. Произвольному графу $G(V,E)$ ставится в соответствие таблица атрибутов $Table_G$. Тогда для произвольных v или e можно определить функцию восстановления значения атрибута под заданным из соответствующей таблицы номером α или β. Далее будем ее обозначать как $Attribute_\alpha(x)$, где в качестве x берется некоторая вершина v. И аналогично $Attribute_\beta(y)$, где за y принимается некоторое ребро e.

В таблице $Table_G$ содержится некоторый элемент, по которому можно однозначно определить v, обозначим его как $key(v)$. За $key(v)$ может быть взят просто числовой идентификатор.

Также в множестве значений для $Attribute_{\alpha}(x)$ и $Attribute_{\beta}(y)$ обязательно существует некоторое фиксированное служебное значение, которое принимается этими функциями в случае отсутствия у их аргумента характеристики с номером α (или β соответственно) в таблице $Table_{G}$.

Для множеств значений $Attribute_{\alpha}(x)$ и $Attribute_{\beta}(y)$ введем хэш-функцию h, принимающую натуральные и нулевые значения. Тогда для произвольной вершины v через $h(key(v))$ можно однозначно определить хэш-функцию и просто записать $h(v)$. И тогда на основании $h(v)$ можно все множество V разделить на непересекающиеся подмножества, в том числе для раздельного хранения группами.

Исходя из используемых при работе с графами взаимодействующих объектов действий с ними построим такую систему хранения, при которой наиболее просто выполняются задачи их объединения, пересечения, а также поиска кратчайших путей. С учетом больших размеров графов из коммуникационных сетей, в которых число вершин может исчисляться десятками миллионов, стандартные подходы не дают качественных результатов.

7.4.2 Архитектура файловой системы хранилища

Адаптированная под указанные ранее специальные задачи архитектура управления данными состоит в реализации виртуальных файлов логики внутри файловой системы. В свою очередь виртуальные файлы содержат записи разной длины и обеспечивают выполнение с ними следующих процедур: использование целочисленных дескрипторов для обращения к такому файлу, дополнение новой записью к его концу, двунаправленное считывание всех записей файла с любой начальной позиции, удаление виртуального файла.

Хранилище представлено набором файлов, идентифицируемых посредством числовой адресации натуральными числами до определенного зафиксированного значения. Каждый такой файл состоит из сегментов заданного неизменного раз-

мера, связанных в двунаправленный список. Для хранения смещений этих сегментов в каждом из файлов в оперативной памяти хранится карта, реализованная также как двунаправленный список. С целью повышения эффективности чтения таких виртуальных файлов логики и поиска по значению заданной характеристики дополнительно карта содержит битовую маску со значениями $h(Attribute_{\alpha}(x))$ для заданных α и каждой записи в сегменте. Такой механизм ускоряет чтение при фильтрации по α, позволяя пропускать сегменты, не содержащие интересующие записи.

Хранилище индекса и данных сформировано в формате файловой подсистемы при помощи потокобезопасной $C++$ библиотеки, способной работать с большим числом физических контейнеров данных, обеспечивая операцию быстрого добавления новой информации в конец контейнера. Многопоточность поддерживается для операций добавления и чтения записей; все прочие действия (создание файлов, добавление потоков в файл, операции сериализации, восстановление из резервной копии и т.д.) являются сервисными. Схема обработки данных в дополнительной файловой системе с 2 физическими и 8 виртуальными файлами логики по 4 потока в каждом показана на рисунке 7.3.

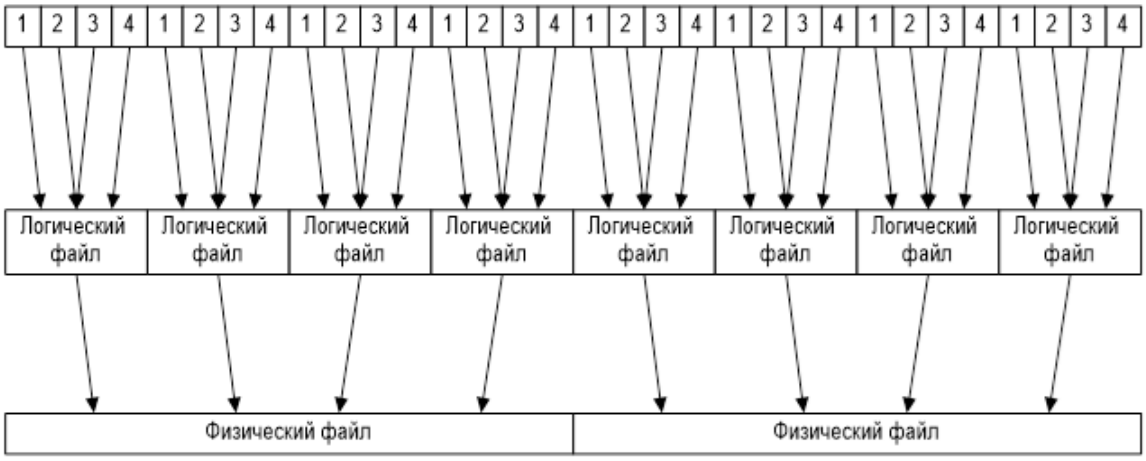

Рисунок 7.3 – Схема обработки данных

Применение самостоятельного механизма организации файлов обусловливается некоторыми плюсами по сравнению со стандартными системами. Возмож-

ность параллельного открытия не столь лимитированного числа файлов. Для виртуальных файлов нет необходимости выполнять дополнительные операции, требующие доступа к аппаратным ресурсам. Также описанная система обеспечивает прозрачную реализацию, использование буфера без необходимости дополнительно учитывать специфику системного программного обеспечения и исключает необходимость избыточных схем адресации и разграничения условий доступа. Благодаря этому отсутствуют дополнительные накладные расходы, связанные с управлением хранением.

7.4.3 Списки смежности

Использование списка смежности как метода хранения данных взято за основу. При этом вершине ставится в соответствие значение $h(v)$. Ее характеристики хранятся в описанном выше формате виртуального файла логики. Причем при их сохранении резервируется некоторое заданное заранее дополнительное место на случай изменения размера хранимого значения. В процессе работы возможно и пополнение новыми характеристиками.

В оперативной памяти каждой v отводится дескриптор $Descriptor(v)$ заданного размера, где указывается ссылка на первую запись в файле характеристик, отвечающую v. Также виртуальный файл $File(v)$, содержащий информацию о рёбрах, сопоставлен v, его номер хранится в $Descriptor(v)$ как и число соседних вершин, привязанных.

Пополнение ребром e оформляется в виде записи, в которую входят $h(v_1(e))$ и $h(v_2(e))$, $Attribute_\beta(e)$ для всех имеющихся β, индивидуальный идентификатор для e. Такая запись добавляется в файлы $File(v_1(e))$ и $File(v_2(e))$. При удалении ребра e формируется и добавляется в $File(v_1(e))$ и $File(v_2(e))$ служебная запись с указанием идентификатора, в случае изменения атрибутов у e аналогично записи добавляются в конец $File(v_1(e))$ и $File(v_2(e))$.

Следовательно, процесс получения списка смежности вершины сводится к последовательному считыванию $File(v)$ в реверсивном порядке. Если в процессе такого чтения в обратном порядке в начале встречается запись с флагом удаления, то далее для этого идентификатора ребра все записи с ним исключаются из конечного списка смежности. При этом для какого-то идентификатора ребра в сформированном списке может присутствовать более одной записи (при отсутствии флага удаления), что обусловлено прошедшими корректировками. В итоговом списке смежности учитывается лишь самая первая прочитанная запись (при обратном проходе).

Для переназначения вершины другому файлу считывается старый, копируется в конец нового список смежности. Далее в исходном файле необходимо выполнить операцию удаления всех записей данной вершины: для этого в файл добавляется специальная сервисная запись. Потом в $Descriptor(v)$ меняется номер файла.

Подобный метод переноса данных не сопряжен со значительными ресурсными затратами, однако подразумевает появление некоторого дублирования информации. Убрать его впоследствии можно при дефрагментации старого файла.

7.4.4 Индекс для характеристик

С целью ускорения поиска вершин по заданным значениям их характеристик в хранилище используется специальный индекс. Для имеющегося числа файлов $Files\_number$ добавление вершины v происходит следующим образом. Создается для характеристики α запись, включающая три компоненты: номер α, $h(Attribute_\alpha(v))$, $h(v)$. Добавляемая запись сохраняется в виртуальном файле, номер которого определяется остатком от деления $h(Attribute_\alpha(v))$ на $Files\_number$. Это позволяет впоследствии оперативно находить вершины с характеристикой, равной заданному значению α_0. При проведении такого поиска нужно взять файл с номером, равным остатку от деления $h(\alpha_0)$ на $Files\_number$ и, отобрав записи индекса по первой и второй его компонентам, получить полный список идентификаторов потенциально подходящих вершин. В силу коллизий хэш-

функции могут присутствовать и вершины с отличающимся значением характеристики. Чтобы исключить такие ложные совпадения, придётся обратиться к файлу для каждой потенциальной вершины, либо дополнительно хранить сами значения $Attribute_{\alpha}(v)$ вместо $h(Attribute_{\alpha}(v))$. Это избавляет от дополнительной фильтрации, однако существенно увеличивает размер индекса. Изменения в значениях характеристик, уже внесенных в индекс вершин подразумевают изменения и в индексе. Внесение же новых характеристик для проиндексированных вершин требует внесения дополнительных записей. Удаление характеристики для вершины в конец файла проводится запись со служебным флагом. Изменения характеристик реализуется через их удаление и добавление.

Иные актуальные операции, режимы работы хранилища и процессы кластеризации данных описаны подробно в монографии автора [73]. Коснемся еще вопроса сжатия данных в хранилище.

7.4.5 Компрессия данных

Для графа $G(V, E)$ хранение информации о E предполагает разделение множества V на непересекающиеся подмножества на основании значений $h(v)$. Внутри каждой такой группы хранение данных о ребрах реализуется через сквозной список. Для добавления в список используется буфер, переполнение которого влечет за собой компрессию с последующей дисковой записью. Его содержимое перед сохранением представляет из себя список наборов. Первые два элемента такого набора представляют из себя $h(v_i)$ и $h(v_j)$ – значения хэш-функции для вершин v_i и v_j, соединенных ребром. Далее идут параметры этого ребра:

$$Param_{i,j}^{s} \text{ где } s = 1, \dots, s_G \tag{7.1}$$

Параметры зависят от особенностей конкретной коммуникационной сети и набора из s_G факторов взаимодействия, учитываемых для графа $G(V, E)$, или совокупности полученных из этих факторов фиксированных свойств ребер этого графа, например вес ребра.

Построение процесса компрессии исходит из особенностей коммуникационной сети, что влечет за собой стандартизированные свойства графа, которые могут приводить к следующим характерным для сохраняемого буфера закономерностям наборов, входящих в список.

А именно, во-первых, исходим из того, что для фиксированного $s = tm$ разброс значений у параметров $Param_{i,j}^{tm}$ для всех записей списка несущественен (не считая какого-то ограниченного количества выбросов). Тогда можно определить медианное значение $Med\_Param_{i,j}^{tm}$ для $Param_{i,j}^{tm}$. Во-вторых, в случаях, если для некоторого $s = type$ значение $Param_{i,j}^{type}$ обозначает тип связи между исходными объектами, то он в большинстве случаев для одного и того же $h(v_i)$ повторяется внутри списка. В-третьих, значения $h(v_i)$ в основном совпадают внутри списка, ибо информация поступает частями и многие из ребер относятся к одинаковой вершине. В-четвертых, нет необходимости сохранять порядок наборов внутри списка. С учетом указанных особенностей перед компрессией, которая производится стандартными методами, проводится предобработка.

Алгоритм 7.1.

Шаг 1. Для параметров вида $Param_{i,j}^{tm}$ производится корректировка на $Med\_Param_{i,j}^{tm}$ с сохранением значения медианы как дополнительного поля для первой записи в списке:

$$Param_{i,j}^{tm} = Param_{i,j}^{tm} - Med\_Param_{i,j}^{tm} \quad (7.2)$$

Шаг 2. Наборы в списке сортируются по элементам лексикографически: $h(v_i)$, $h(v_j)$, $Param_{i,j}^{s}$. После этого для наборов, которые имеют совпадающие значения по первому элементу $h(v_i)$, производится группировка. К первому обновленному набору такой группы добавляется элемент с числом наборов группы. У остальных наборов группы убирается элемент $h(v_i)$ как совпадающий.

Шаг 3. В случае, если в силу особенностей графа возможно более одного набора с совпадающими значениями $h(v_j)$ внутри сформированной на прошлом шаге группы производится аналогичный процесс по этому элементу.

Шаг 4. Если внутри группы имеется последовательность наборов с совпадающим значением элемента вида $Param_{i,j}^{type}$, то он сохраняется в первом наборе, который дополняется элементом-флагом (битом).

Шаг 5. Внутри каждой группы (подгруппы), полученных на прошлых шагах рассматриваются элементы вида $Param_{i,j}^{tm}$. А именно, вычисляется следующая разность для второго и далее набора внутри группы. Из значения этого элемента вычитаем значение такого же элемента предыдущего набора внутри группы.

В данном алгоритме шаг 2 необратим, но исходя из четвертого предположения о составе наборов это не играет роли, ибо сами наборы полностью могут быть воспроизведены за счет обратимости остальных шагов алгоритма.

Для данных, полученных из реальных сетей коммуникаций могут выявляться некоторые дополнительные особенности, характерные для сетей из этого источника. Например, при сохраняющейся периодичности взаимодействия может быть выявлена дополнительная регулярность в последовательных элементах вида $Param_{i,j}^{tm}$, которую можно использовать схожим образом. А именно, определив стандартную разность $delta\_Param_{i,j}^{tm}$ и заменив элементы $Param_{i,j}^{tm}$ на значения $Param_{i,j}^{tm} - delta\_Param_{i,j}^{tm}$, сохранив для соответствующей группы значение $delta\_Param_{i,j}^{tm}$.

Реализация таких особенностей выполняется за счет добавления при компрессии дополнительных флагов битового размера. Это дает возможность хранить существенно меньше данных по объему.

Подобные модификации позволяют в ряде случаев хранить всю информацию о каком-то очень большом графе на одному узле компьютерной сети, что влечет за собой ускорение операций, ибо для стандартных графовых операций не требуется обмениваться данными между разными узлами этой сети.

7.5 Экспериментальные оценки характеристик хранилищ

Описанные в 7.4 методы и алгоритмы применены в *Analytics and Visualization System for graphs*. С целью оценки реализованного в системе хранилища были проведены вычислительные эксперименты. Сравнение производилось с иными программными продуктами схожей направленности и функционала.

Сравнительное тестирование проводилось для разработанного хранилища *AVS-Storage* и следующих баз данных: *OrientDB, NEO4J* и *ArangoDB*. Для полноты сравнения эксперименты проводились в том числе и для наиболее популярных систем баз данных, позволяющих хранить графы в табличном представлении, – *Microsoft SQL Server* и *MySQL* [159], а также и для *MongoDB* [164].

Сравнительное тестирование хранилищ проводилось в первую очередь на перечисленных в таблице 7.1 пяти графах, импортированных из коммуникационных сетей. Выделение неявных сообществ осуществлялось описанными в главах 3, 4 и 5 алгоритмами. Графы G_{tw_M} и G_{tw_sp} были получены при импорте данных из сети *Twitter* (глава 2, раздел 2.3). Графы G_{tg_im} и G_{tg_im} получены при импорте данных о *Telegram*-каналах (глава 2, раздел 2.4). Граф G_{vk} построен на основе данных, скачанных по моделям второй главы из сети *ВКонтакте.* Таким образом, первые пять графов, приведенные в таблице 7.1, достаточно полно представляют спектр данных, для которых предназначено хранилище *AVS-Storage*.

Для тестирования хранилища на больших объемах данных был выбран граф G_{fb}, полученный при импорте из социальной сети *Facebook*[1]. Этот граф был скачен с ресурса *Network Repository* [178], репозитория с огромными наборами данных. Тестовый набор данных представляет собой граф с 69400 вершинами и 1,6 млн ребрами, у которого средняя степень вершины равна 47, а максимальная степень вершины достигает значения 8900. Визуальное представление структуры данного графа показано на рисунке 7.4.

[1] Принадлежит компании Meta, которая признана экстремистской и запрещена в Российской Федерации

Таблица 7.1 – Размеры тестовых графов

Граф	Количество вершин	Количество ребер
G_{tw_M}	632	1 002
G_{tw_sp}	524	265
G_{tg_im}	625	6 137
G_{tg_rsv}	773	6 611
G_{vk}	8 781	13 789
G_{fb}	69 400	1 600 000

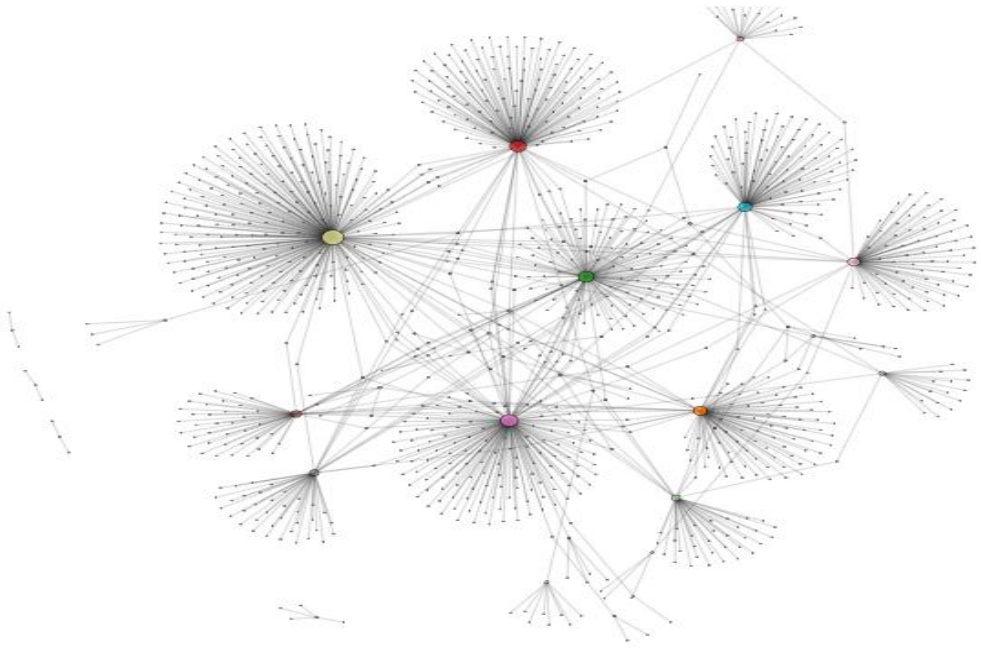

Рисунок 7.4 – Визуализация структуры тестового графа G_{fb}

Экспериментальное тестирование хранилища с помощью первых пяти графов производилось под операционными системами Fedora 12 64 bit, kernel 2.6.31.5-127.fc12.x86_64 и Windows 2008 Server Service Pack 2 64 bit. Данные ОС устанавливались на вычислительной машине следующей конфигурации: процессор Intel Core 2 Quad Q6600, 2400 МГц, 8МБ кэш-память L2; оперативная память 8 ГБ DDR2 800 МГц; дисковая подсистема Seagate Barracuda 7200.12, 1000 ГБ, буфер 32 МБ. Тестирование хранилища на большом графе G_{fb} проводилось на сервере с характеристиками: 2 процессора *Intel Xeon X5650* 2.67 ГГц, оперативная память 96 ГБ, дисковая полка из 20 дисков, суммарным объемом 38 ТБ. Для большей прозрачности

в таблицах 7.2, 7.4-7.6 результаты тестирования приведены с указанием количества вершин или ребер графов.

Таблица 7.2 – Выполнение записи тестовых графов в СУБД

Число ребер/ время, ms	AVS	Neo4j	OrientDB	ArangoDB	MongoDB	MySql
265	4,36	5210	5070	13100	594	27
1002	7,11	5670	13400	23100	997	43
6137	39,53	11300	53620	63300	4080	236
6611	35,53	13200	63950	68540	4670	345
13789	95,9	323100	1112000	247000	18400	485

Временные показатели (в миллисекундах) загрузки тестовых графов в различные средства хранения приведены в таблице 7.2, что проиллюстрировано графиками на рисунках 7.5, 7.6 и 7.7. При этом для удобства визуализации результатов на рисунке 7.6 на оси ординат графика время представлено в логарифмической шкале.

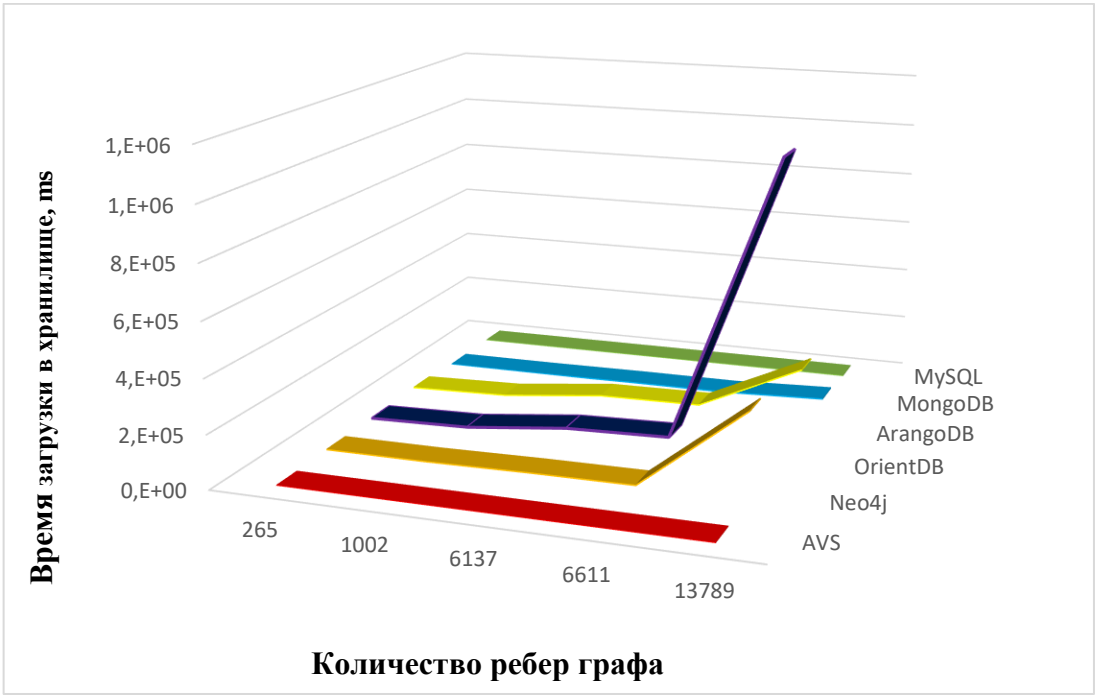

Рисунок 7.5 – Время выполнения загрузки графа в хранилище

Из приведенных результатов (таблицы 7.2 и рисунки 7.5, 7.6) видно, что временные характеристики загрузки данных в хранилища графов не только намного выше для *OrientDB*, *NEO4J* и *ArangoDB*, но и недопустимо растут при небольшом увеличении размеров графа. При этом наилучшими показателями отличаются *AVS-Storage* и *MySQL*. Сравнение временных показателей для этих хранилищ приведено на рисунке 7.7, из которого видна эффективность разработанного *AVS-Storage* по временным характеристикам загрузки графа в хранилище.

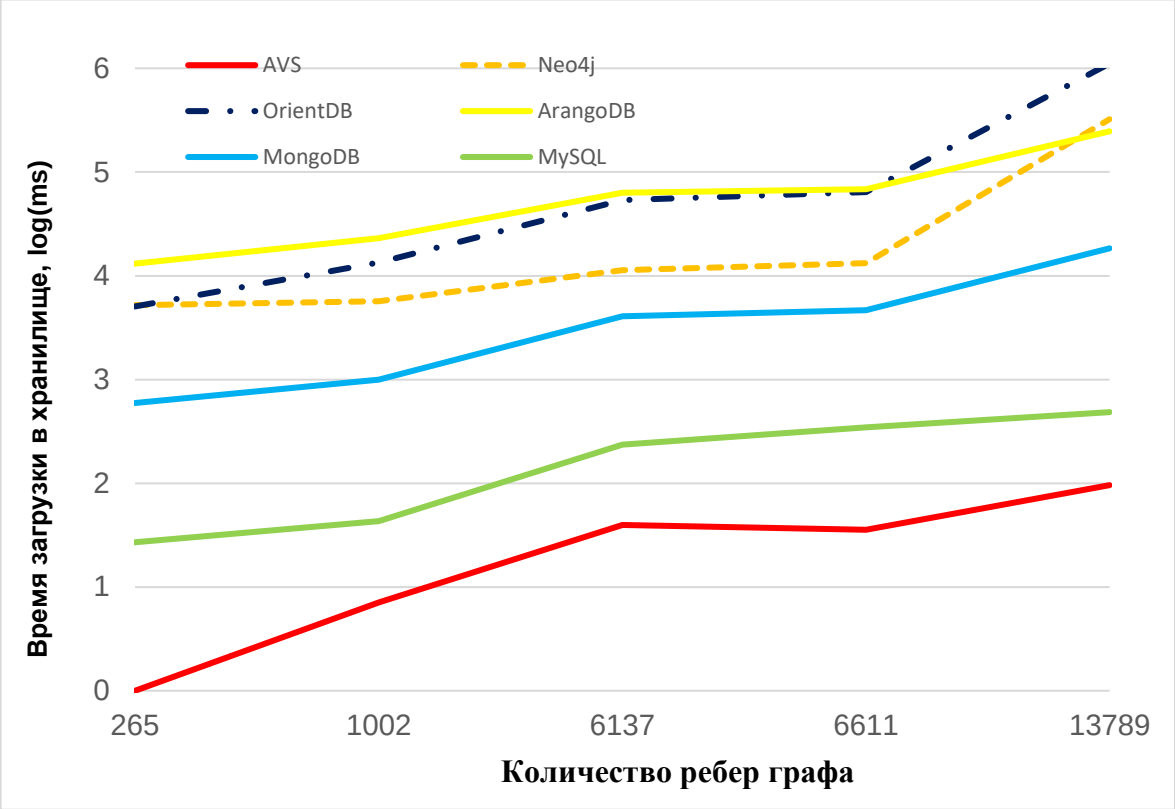

Рисунок 7.6 – Время (логарифм) выполнения загрузки графа в хранилище

Требуемые ресурсы для записи и хранения данных большого размера на примере G_{fb} приведены в таблице 7.3. Сохранение тестового графа соответствует 1,6 млн записей (число ребер) в реляционную базу данных (*MS SQL Server*). Отметим наилучшие временные и объемные характеристики для разработанного хранилища *AVS-Storage*, как специализированной системы хранения графов. При этом хранилище *AVS-Storage*, по быстродействию и оптимальности дисковой памяти намного превышает другие графовые хранилища (*OrientDB* и *NEO4J*).

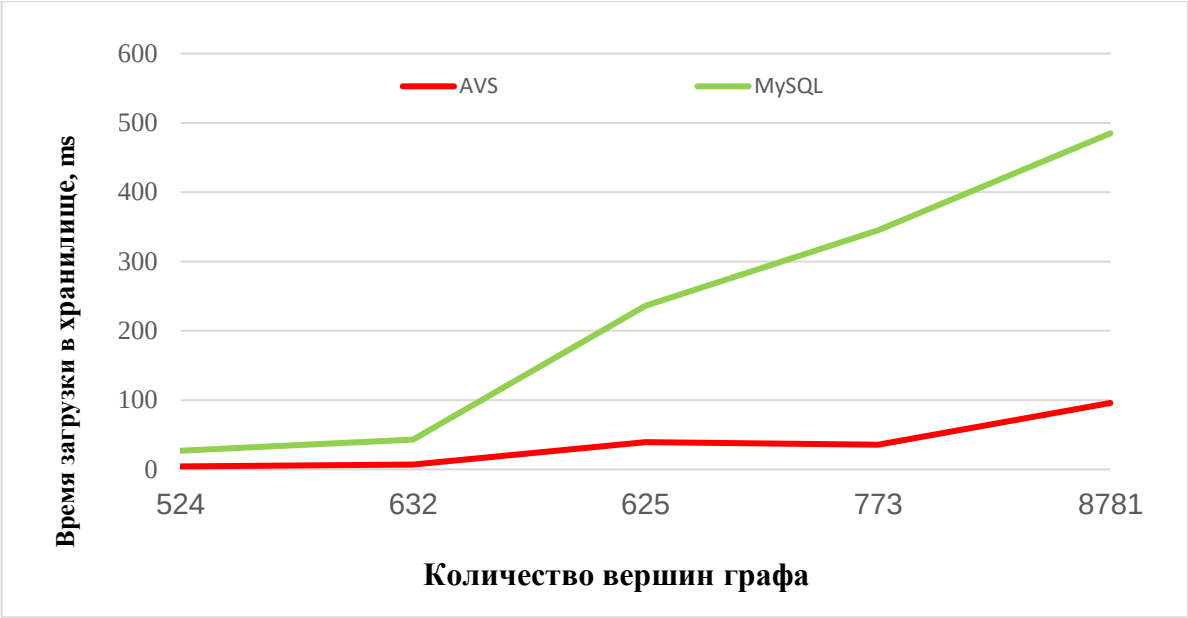

Рисунок 7.7 – Время выполнения загрузки графа в хранилище AVS и в СУБД MySQL

Таблица 7.3 – Выполнение записи графа G_{fb} в СУБД

СУБД	Время записи, сек.	Память на диске, МБ
MS SQL Server	158,1	224,9
OrientDB	112,4	195,1
Neo4j	134,8	452,7
AVS-Storage	74,2	133,9

Время выполнения запросов к хранилищам с графами для получения атрибутов вершин приведено в таблице 7.4 и на рисунке 7.8. Время выполнения запросов к хранилищам для получения атрибутов ребер приведено в таблице 7.5 и на рисунке 7.9. Приведенные результаты демонстрируют высокое быстродействие этом хранилище *AVS-Storage*.

Таблица 7.4 – Выполнение запросов по получению атрибута вершины

Число вершин/ время, ms	AVS	Neo4j	OrientDB	ArangoDB	MongoDB	MySql
524	0,036	1,58	1,78	1,37	0,4	0,27
625	0,042	2,02	2,06	1,96	0,77	0,35
632	0,039	1,82	2,18	1,37	0,44	0,35
773	0,061	1,78	2,4	1,27	0,47	0,37
8781	0,58	5,54	30,8	1,34	3,78	4,32

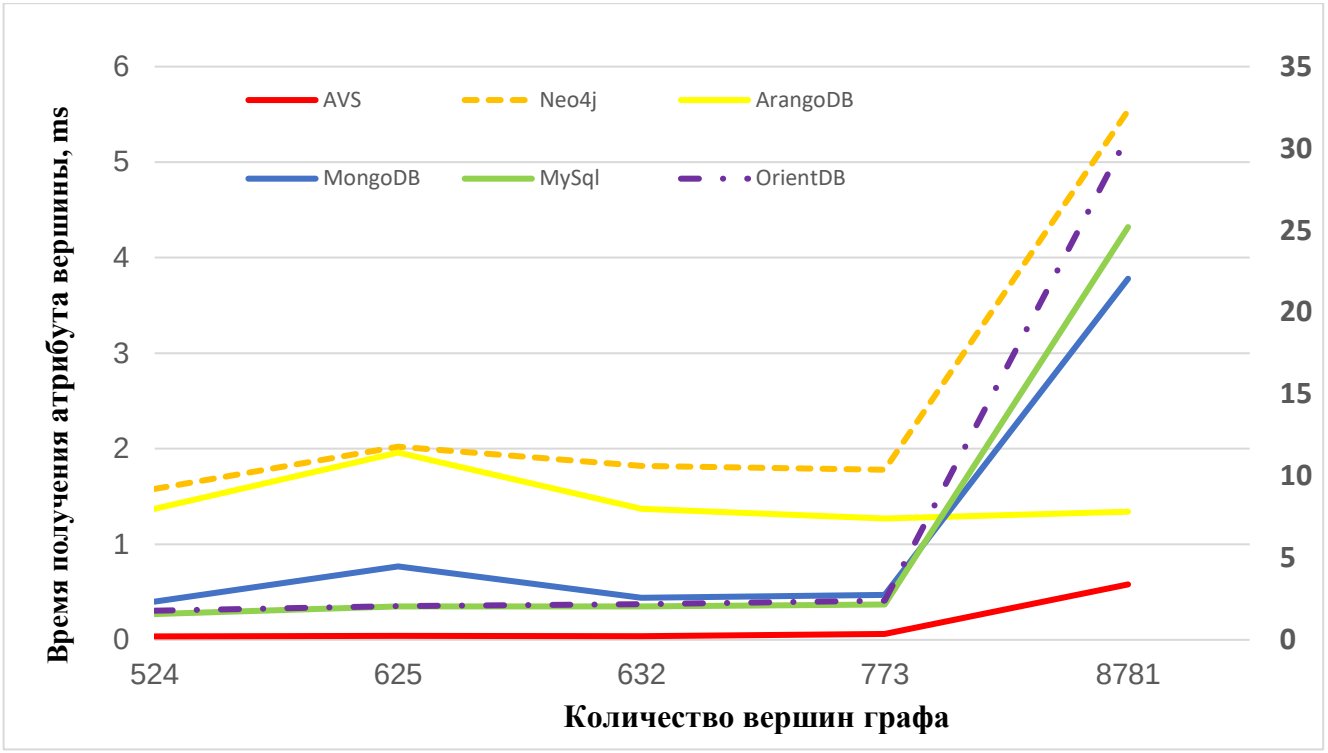

Рисунок 7.8 – Время выполнения запросов на получение атрибута вершины (дополнительная ось для OrientDB)

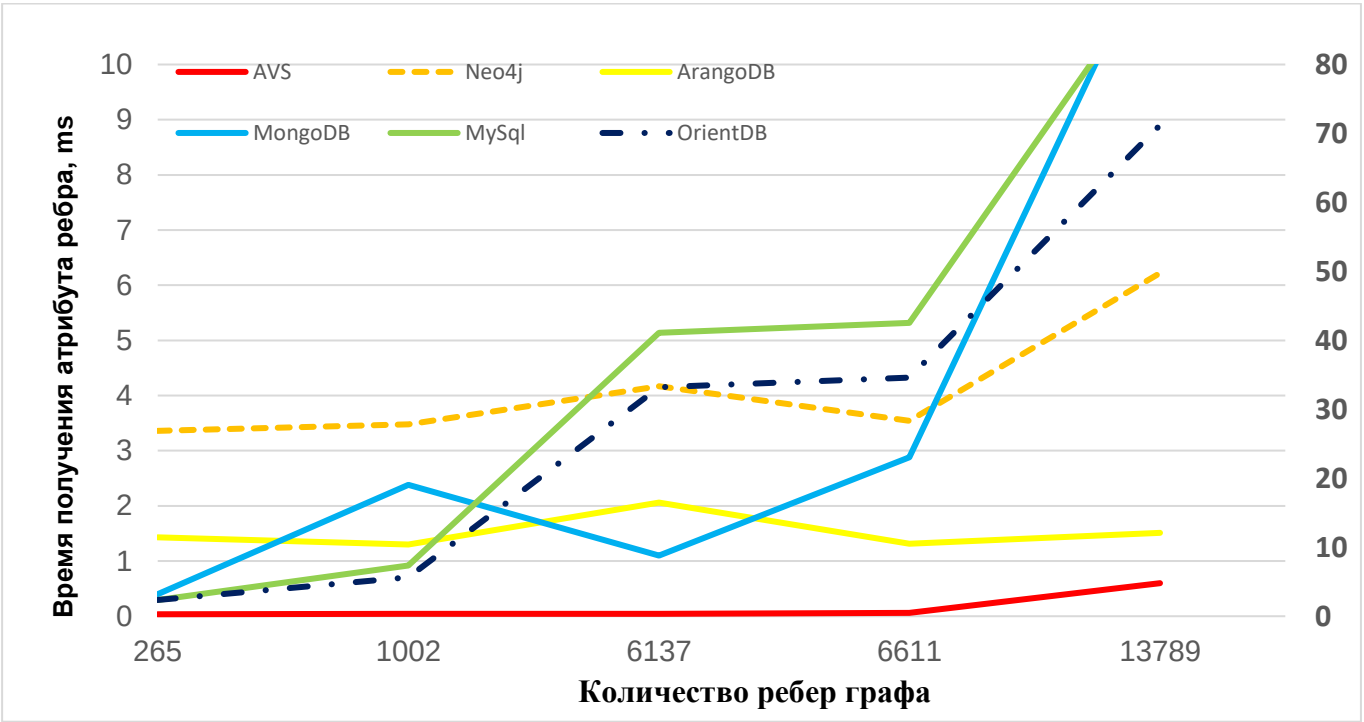

Рисунок 7.9 – Время выполнения запросов на получение атрибута ребра (дополнительная ось для OrientDB)

Таблица 7.5 – Выполнение запросов по получению атрибута ребра

Число ребер/ время, ms	AVS	Neo4j	OrientDB	ArangoDB	MongoDB	MySql
265	0,036	3,36	2,36	1,43	0,4	0,294
1002	0,043	3,48	5,64	1,3	2,38	0,92
6137	0,046	4,17	33,2	2,06	1,1	5,14
6611	0,0597	3,54	34,6	1,316	2,88	5,32
13789	0,599	6,22	71,2	1,514	12,48	11,86

Возможности использования хранилища для реализации алгоритмов анализа графов определялись по эффективности операций поиска соседних вершин. Для типовых графов взаимодействия объектов в социальных сетях, использованных в качестве тестовых, замерялось время выполнения запросов по поиску соседей первого порядка. Результаты представлены в таблице 7.6 и на рисунках 7.10 и 7.11.

Таблица 7.6 – Выполнение запросов по поиску соседей

Число ребер/ время, ms	AVS	Neo4j	OrientDB	ArangoDB	MongoDB	MySql
265	0,000095	0,792	4,46	1,58	0,006	0,29
1002	0,0001	0,976	12,92	1,86	0,006	0,91
6137	0,00015	1,57	60,2	4,56	0,0065	6,04
6611	0,000147	0,656	70,2	4,72	0,0059	5,8
13789	0,000087	3,72	19,4	8,12	0,0061	11,8

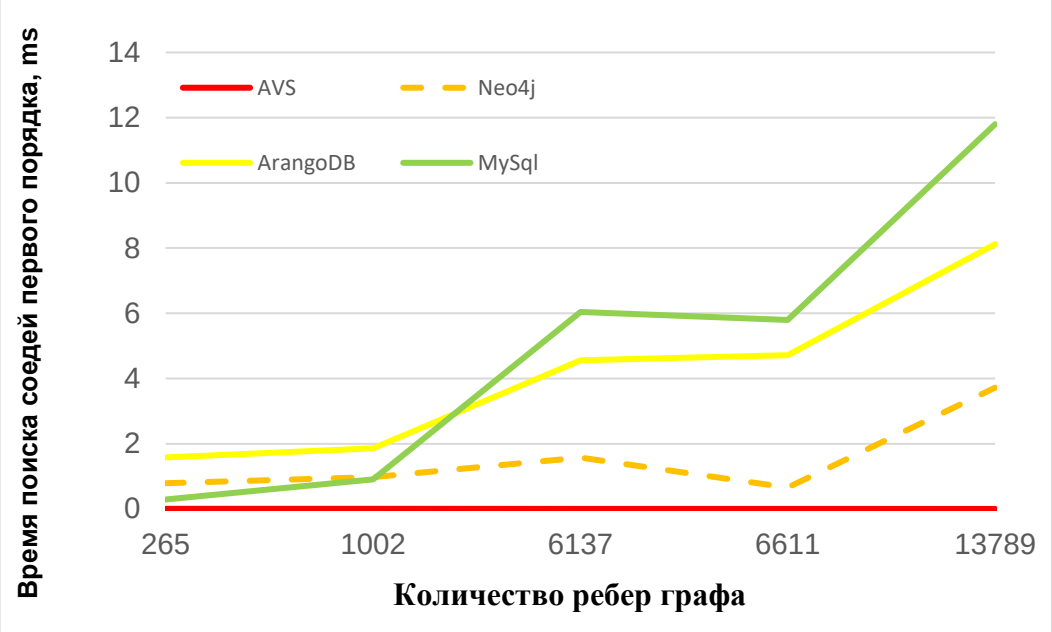

Рисунок 7.10 – Время выполнения запросов на поиск соседей вершин первого порядка

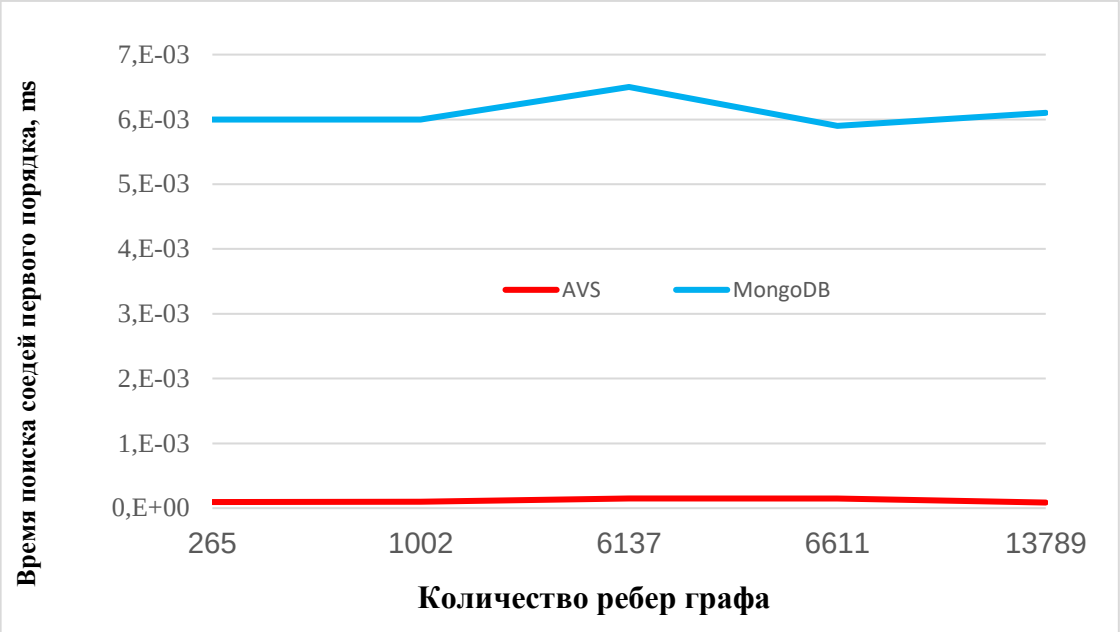

Рисунок 7.11 – Время выполнения запросов на поиск соседей вершин первого порядка

Характеристики требуемого времени и используемых объемов памяти при выполнении запросов по поиску соседей вершин графа G_{fb} приведены в таблице 7.7. Для наглядности результаты для трех хранилищ графов визуализированы на рисунках 7.12 и 7.13, где представлены соответственно время выполнения запросов на поиск соседей у вершин степени n и количество оперативной памяти, необходимой для выполнения таких запросов.

Таблица 7.7 – Выполнение запросов по поиску соседей графа G_{fb}

СУБД	Поиск соседей 1-го порядка		Поиск соседей 2-го порядка		Поиск соседей 3-го порядка		Поиск соседей 4-го порядка	
	RAM, МБ	Время, сек.	RAM, МБ	Время, сек.	RAM, МБ	Время, сек.	RAM, МБ	Время, сек.
MS SQL Server	182	6,07	191	14,8	266	26,4	281	65,9
Orient DB	143	1,2	149	2,2	164	3,4	182	13,3
Neo4j	167	0,4	179	0,7	195	1,1	237	2,3
AVS-Storage	405	0,001	405	0,013	405	0,022	405	0,7

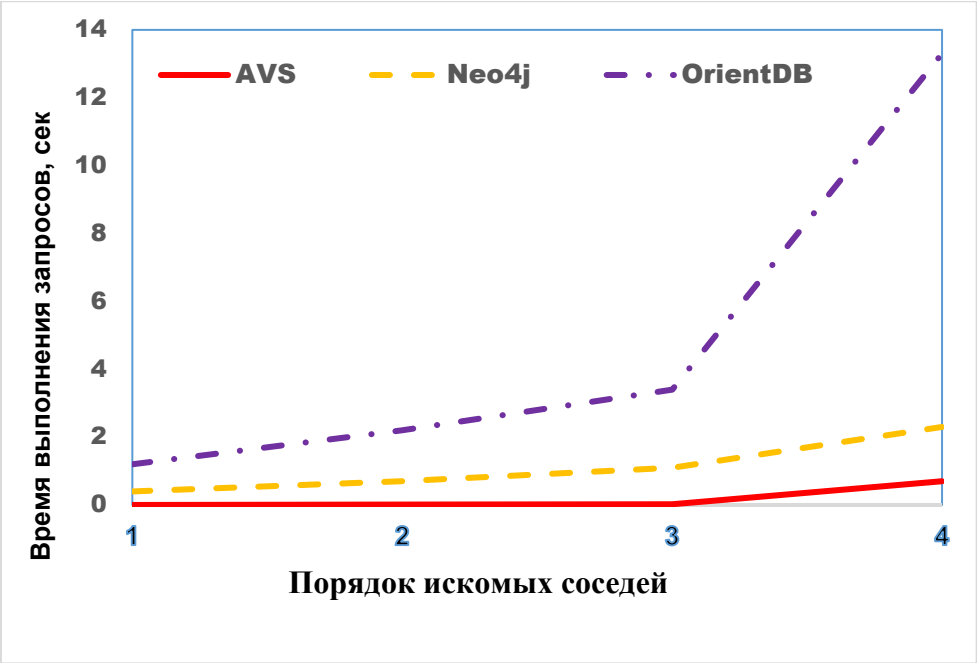

Рисунок 7.12 – Время выполнения запросов на поиск соседей вершин степени n

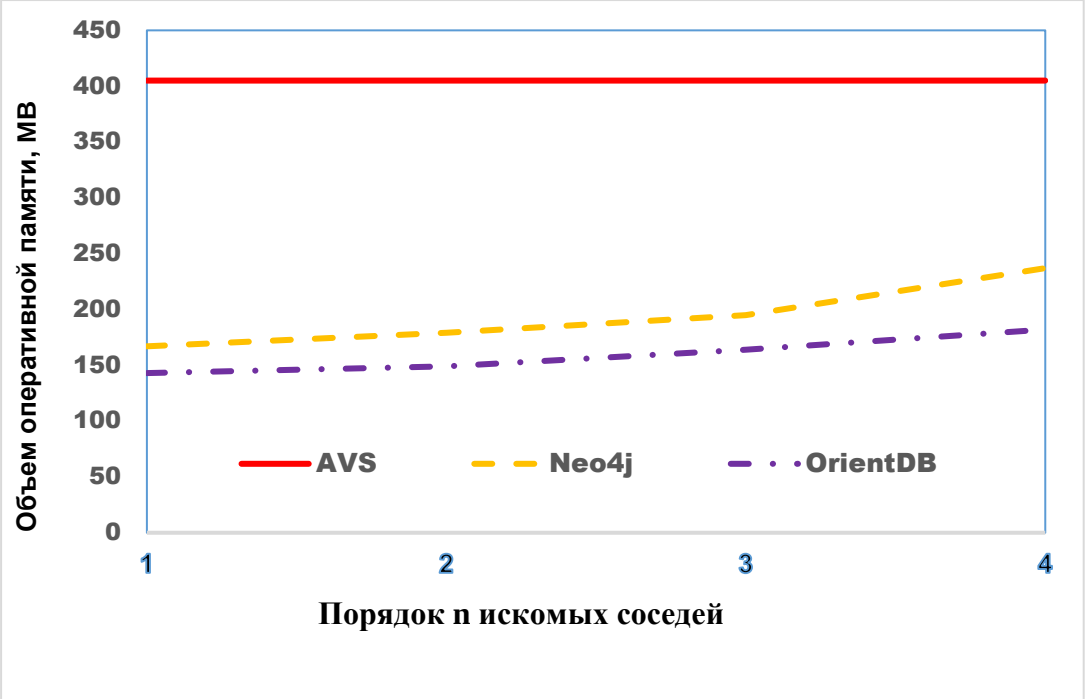

Рисунок 7.13 – Количество оперативной памяти, необходимой для выполнения запросов на поиск соседей вершин порядка n

Из графика, приведенного на рисунке 7.10, видно, что для большинства хранилищ (*NEO4J, ArangoDB* и *MySQL*) даже для поиска ближайших соседей (первого порядка) время выполнения операций растет с увеличением размера графа. Этой

тенденции не наблюдается для хранилища *AVS-Storage*, где время выполнения операций поиска ближайших соседей существенно меньше (таблица 7.6 и рисунок 7.10).

Рисунок 7.13 демонстрирует, что количество оперативной памяти, необходимой для выполнения запросов на поиск соседей вершин степени n для хранилища *AVS-Storage* постоянно и превосходит другие специализированные СУБД. Это можно объяснить тем, что *AVS-Storage* выделяет максимальную оперативную память для работы с графом. Плюсом является то, что операция поиска в этой связи затрачивает минимально оперативной памяти, ибо достигается за счет доступа к области памяти с нужным элементом графа. Это приводит к тому, что для хранилища *AVS-Storage* время выполнения запросов на поиск соседей является минимальным среди специализированных СУБД (рисунок 7.12).

Из результатов, приведенных в таблицах 7.3 и 7.7 для графа G_{fb} большого размера видно, что *MS SQL* является самым неэффективным инструментом для реализации алгоритмов анализа графов, что отражает ситуацию с классическими реляционными СУБД. При этом созданные специализированные решения для графов более эффективны по характеристикам хранения и доступа к данным. Хранилище *AVS-Storage* показывает наилучшую эффективность по быстродействию и дисковой памяти по сравнению с другими хранилищами графов.

7.6 Выводы по главе 7

1. Разработанная и представленная в данной главе модель для эффективного хранения графов взаимодействующих объектов основана на алгоритмах сжатия и оптимизации операций с графами по памяти и по скоростным характеристикам.

2. Представленные экспериментальные данные показывают эффективность модели по скоростным характеристикам загрузки и доступа к данным созданного хранилища графов.

3. Разработанная архитектура и созданное программное обеспечение позволяют проводить анализ графов и выделенных на них сообществ с удобной визуализацией. В созданном программном комплексе реализованы разработанные в диссертации методы и алгоритмы, получившие таким образом практическую реализацию в прикладном программном обеспечении.

4. Основные результаты, представленные в главе 7, опубликованы в следующих работах: [62, 73, 75, 130, 131, 132, 133, 134, 135, 136]. Вклад соискателя заключается в разработке архитектуры программного комплекса и используемых структур данных, методов хранения и алгоритмов.

ЗАКЛЮЧЕНИЕ

ОСНОВНЫЕ РЕЗУЛЬТАТЫ, ПОЛУЧЕННЫЕ В ДИССЕРТАЦИОННОЙ РАБОТЕ

1. Представлено решение научной проблемы, имеющей важное хозяйственное значение, заключающееся в создании моделей, разработки численных методов и программного обеспечения для анализа структуры графов взаимодействующих объектов, полученных при импорте данных из социальных сетей и сетей мгновенного обмена сообщениями с целью описания информационного взаимодействия объектов.

2. Разработаны и реализованы в программном обеспечении вариации модели формирования взвешенного графа информационного взаимодействия для разных социальных сетей и сетей мгновенного обмена сообщениями. Приведены вариации данной модели для импорта данных из сети Twitter, сети Telegram-каналов и социальной сети ВКонтакте.

3. Построены итерационные численные методы и алгоритмы для выделения неявных сообществ и ключевых вершин графов с использованием эвристик, а именно.

3.1. Предложен и реализован «Комбинированный алгоритм» для выделения пересекающихся и вложенных сообществ на графе, позволяющий убирать из рассмотрения малозначимые элементы сети и предусматривающий параметрические модификации для формирования разнородных разбиений в зависимости от задач оператора.

3.2. Предложен и реализован «Метод ядра» для выделения непересекающихся сообществ на взвешенных графах, предусматривающий выделение ключевой компоненты на основании вычисляемых в явном виде характеристик графа. Апробация метода продемонстрирована на реальных данных из сети Twitter.

3.3 Предложен и реализован «Метод Галактик» для выделения пересекающихся сообществ на взвешенных графах, основанный на последовательном применении других алгоритмов, обработке графа, переходам к мета-графу из мета-сооб-

ществ и последующем выделении пересекающихся сообществ. Показано применение алгоритма на реальных данных из сети Telegram-каналов с последующим экспертным обоснованием качества полученного разбиения.

4. Предложена методика оценки эффективности выделения сообществ на графе с помощью алгоритмов компьютерной лингвистики для обработки текстовых метаданных (атрибутов вершин выделенных сообществ) и анализа психолингвистических факторов.

5. Разработана модель для эффективного хранения графов взаимодействующих объектов, основанная на алгоритмах сжатия и оптимизации операций с графами по памяти и по скоростным характеристикам. Создано программное обеспечение для анализа графов взаимодействующих объектов.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ ДИССЕРТАЦИИ ОПУБЛИКОВАНЫ В РАБОТАХ

Основные результаты диссертационного исследования опубликованы в 37 работах. Из них 22 статьи [63, 64, 65, 66, 67, 68, 69, 72, 74, 90, 91, 92, 95, 113, 116, 117, 118, 119, 131, 132, 133, 134] в ведущих рецензируемых научных журналах, которые входят в утвержденный ВАК Минобрнауки России «Перечень российских рецензируемых научных журналов, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученых степеней доктора и кандидата наук» по специальности 1.2.2 (05.13.18) [64, 66, 68, 69, 72, 92, 95, 113, 116, 117, 118, 119, 131, 132], в том числе 12 в журналах категории К1 и К2 по распределению на 2022 и 2023 годы, и приравненных к ним изданиям в зарубежных рецензируемых изданиях и в базе RSCI (на платформе Web of Science) [63, 65, 67, 74, 90, 91, 133, 134], две монографии [73, 112], 13 публикаций [70, 71, 75, 76, 88, 89, 93, 94, 96, 120, 130, 135, 136] в трудах международных научных конференций.-Два свидетельства о регистрации программ для ЭВМ [179, 180]. Результаты диссертации использованы в 2 учебных пособиях [62, 121].

СПИСОК ЛИТЕРАТУРЫ

1. Евин, И. А. Введение в теорию сложных сетей / И. А. Евин // Компьютерные исследования и моделирование. — 2010. — 2(2). — С. 121–141. DOI: 10.20537/2076-7633-2010-2-2-121-141
2. Newman, M. E. J. Networks: An Introduction / M. E. J. Newman. — Oxford University Press, 2010. — 784 p.
3. Fortunato, S. Community Detection in Graphs / S. Fortunato // Physics Reports. — 2010. —486(3). — P. 75-174.
4. Aggarwal, C. Social Network Data Analytics / C. Aggarwal. —NY: Springer New York, 2011. — 502 p. DOI: 10.1007/978-1-4419-8462-3.
5. Рабинович, Б. И. Кластерный анализ детализаций телефонных переговоров / Б. И. Рабинович // Системы и средства информатики. Ин-т пробл. информатики РАН. — М.: Наука, 2007. — *Вып. 17*. — С. 52-78.
6. Себякин, А. Г. Анализ информации о соединениях между абонентами, использование его результатов в раскрытии и расследовании преступлений / А. Г. Себякин // Полицейская и следственная деятельность. — 2018. — 4. — С. 29-38. DOI: 10.25136/2409-7810.2018.4.27992.
7. Семенищев, И. А. Синтез массивов биллинговой информации на основе статистико-событийной модели взаимодействия абонентов сетей сотовой связи / И. А. Семенищев, А. Н. Синадский, Н. И. Синадский, П. В. Сушков // Вестник УРФО. Безопасность в информационной сфере. — 2018. — 1(27). — С. 47-56.
8. Еремеев, И. Ю. Анализ мер центральности вершин сетей на основе метода главных компонент / И. Ю. Еремеев, М. В. Татарка, Ф. Л. Шуваев, А. С. Цыганов // Информатика и автоматизация (Труды СПИИРАН) . — 2020. — 19(6). — P. 1307-1331. DOI: 10.15622/ia.2020.19.6.7.
9. Кириченко, Л. Обнаружение киберугроз с помощью анализа социальных сетей / Л. Кириченко, Т. Радивилова, А. Барановский // International Journal Information Technologies& Knowledge. — 2017. — 11(1). — С. 23-48.

10. Rahiminejad, S. Topological and functional comparison of community detection algorithms in biological networks / S. Rahiminejad, M. R. Maurya, S. Subramaniam // BMC Bioinformatics. — 2019. — 20.212. — 25 p.
11. Wu, F. Biomolecular Networks for Complex Diseases / F. Wu, L. Chen, J. Wang, M. Li, H. Wang // Complexity. — 2018. — Article ID 4210160. — 3 p.
12. Šubelj, L. Ubiquitousness of link-density and link-pattern communities in real-world networks / L. Bajec, M. Šubelj // The European Physical Journal. B. — 2012. — 85(1). — 32 p.
13. Šubelj, L. Group detection in complex networks: an algorithm and comparison of the state of the art / L. Bajec, M. Šubelj // Physica A: Statistical Mechanics and Its Applications. — 2014. — 397. —P. 144-156.
14. Landsman, D. Zoning of St. Petersburg Through the Prism of Social Activity Networks / D. Landsman, P. Kats, A. Nenko, S. Sobolevsky // Procedia Computer Science. — 2020. — 178. — P. 125–133.
15. Ser-Giacomi, E. Explicit and implicit network connectivity: Analytical formulation and application to transport processes / E. Ser-Giacomi, T. Legrand, I. Hernández-Carrasco, V. Rossi // Physical Review E 103.042309. — 2021. — 15 p.
16. Banerjee, S. Designing and connectivity checking of implicit social networks from the user-item rating data / S. Banerjee // Multimedia Tools and Applications. — 2021. — 80(17). — P. 26615–26635.
17. Castillo-de Mesa, J. Connectedness, Engagement, and Learning through Social Work Communities on LinkedIn / J. Castillo-de Mesa, L. Gómez-Jacinto // Psychosocial Intervention. — 2020. — 29(2). —P. 103-112.
18. Skobtsov, Y. A. Building And Analysing A Skills Graph Using Data From Job Portals / Y. A. Skobtsov, D. M. Obolensky, V. I. Shevchenko, O. V. Chengar / In I. Kovalev, & A. Voroshilova (Eds.), Economic and Social Trends for Sustainability of Modern Society (ICEST-III 2022). European Proceedings of Social and Behavioural Sciences. European Publisher. — 2022. — 127. — P. 147-162. DOI: 10.15405/epsbs.2022.08.17/.

19. Райгородский, А. М. Модели Интернета: учебное пособие / А. М Райгородский. – Долгопрудный: Издательский Дом «Интеллект», 2019. — 64 с.
20. Ермолин, Н. А. Теоретико-игровые методы нахождения сообществ в академическом Вебе / Н. А. Ермолин, В. В. Мазалов, А. А. Печников // Труды СПИИРАН. — 2017. — 6(55). — С. 237-254. DOI 10.15622/sp.55.10.
21. Мазалов, В. В. О сообществах в коммуникационных графах / В. В. Мазалов, Н. Н. Никитина, А. А. Печников / Вероятностные методы в дискретной математике Расширенные тезисы докладов X Международной Петрозаводской конференции. — Петрозаводск: Изд-во: Федеральное государственное бюджетное учреждение науки Федеральный исследовательский центр "Карельский научный центр Российской академии наук", 2019. — С. 99–100.
22. Черемисинов, Д. И. Поиск часто встречающихся подграфов / Д. И. Черемисинов, Л. Д. Черемисинова / BIG DATA Advanced Analytics: collection of materials of the fourth international scientific and practical conference, Minsk, Belarus, May 3 – 4, 2018. — Minsk: BSUIR, 2018. — С. 171 – 176.
23. Rehman, S. U. A Graph Mining Approach for Ranking and Discovering the Interesting Frequent Subgraph Patterns / S. U. Rehman, Liu K. Kexing, Ali T. Tariq, A. Nawaz, S. J. Fong// International Journal of Computational Intelligence Systems. — 2021. — 14(152). — 17 p. DOI: 10.1007/s44196-021-00001-4.
24. Базенков, Н. И. Обзор информационных систем анализа социальных сетей / Н. И. Базенков, Д. А. Губанов / Управление большими системами: сб. трудов, 2013. — С. 357-394.
25. Girvan, M. Finding and evaluating community structure in networks / M. Girvan, M. E. Newman // Physical Review. — 2004. — E 69. 026113. — 16 p.
26. Батура, Т. В. Методы анализа данных из социальных сетей / Т. В. Батура, Н. С. Копылова, Ф. А. Мурзин, А. В. Проскуряков // Вестник НГУ. Серия: Информационные технологии. — 2013. — 11(3). — С. 5-21.
27. Губанов, Д. А. Социальные сети: модели информационного влияния, управления и противоборства / Д. А. Губанов, Д. А. Новиков, А. Г. Чхартишвили. — М.: Физматлит: МЦНМО, 2010. — 228 с.

28. Borgatti, S. P. Analyzing social networks / S. P. Borgatti S. P., M. G. Everett, J. C. Johnson. — SAGE Publications Limited, 2013. — 304 p.
29. Coscia, M. Demon: a local-first discovery method for overlapping communities / M. Coscia, G. Rossetti, F. Giannotti, D. Pedreschi / In: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. –ACM, 2012. — P. 615-623.
30. Gaisbauer, F. Ideological differences in engagement in public debate on twitter / F. Gaisbauer, A. Pournaki, S. Banisch // Plos One. — 2021. — 16(3). — 12 p.
31. Kanavos A. Evaluating Methods for Efficient Community Detection in Social Networks / A. Kanavos, Y. Voutos, F. Grivokostopoulou, P. Mylonas // Information. — 2022. — 13(209). — 19 p.
32. Yang, J. Defining and evaluating network communities based on ground-truth / J. Yang, J. Leskovec // Knowledge and Information Systems. — 2015. — 42(1). — P. 181–213.
33. Гусарова, Н. Ф. Анализ социальных сетей. Основные понятия и метрики / Н. Ф. Гусарова. — СПб: Университет ИТМО, 2016. — 67 с.
34. Евин, И. А. Социальные сети / И. А. Евин, Т. Ф. Хабибуллин // Компьютерные исследования и моделирование. — 2012. — 4(2). — P. 423-430. DOI: 10.20537/2076-7633-2012-4-2-423-430.
35. Newman, M. E. J. The structure and function of complex networks / M. E. J. Newman // SIAM Review. — 2003. — 45(10). — P. 167-256.
36. Проноза, А. А. Методика выявления каналов распространения информации в социальных сетях / А. А. Проноза, Л. А. Виткова, А. А. Чечулин, И. В. Котенко, Д. В. Сахаров // Вестник Санкт-Петербургского университета. Прикладная математика. Информатика. Процессы управления. — 2018. — 14(4). — С. 362–377. DOI: 10.21638/11702/spbu10.2018.409.
37. Градосельская, Г. В. Картирование политически активных групп в Фейсбуке: динамика 2013-2018 гг. / Г. В. Градосельская, Т. Е. Щеглова, И. А. Карпов // Вопросы кибербезопасности. — 2019. — 32(4). — С. 94-104. DOI: 10.21681/2311-3456-2019-4-94-104.

38. Borgatti, S. P. Centrality and network Flow / S. P. Borgatti // Social Networks. — 2005. — 27(1). — С. 55-71. DOI: 10.1016/j.socnet.2004.11.008.
39. Щербакова, Н. Г. Меры центральности в сетях / Н. Г. Щербакова // Проблемы информатики. — 2015. — 2 (27). — С. 18-30.
40. Печенкин, В. В. Прикладные аспекты использования алгоритмов ранжирования для ориентированных взвешенных графов (на примере графов социальных сетей) / В. В. Печенкин, М. С. Королёв, Л. В. Димитров // Труды СПИИРАН. — 2018. — 6(61). — С. 94-118. DOI: 10.15622/sp.61.4.
41. Rajeh, S. Comparative evaluation of community-aware centrality measures / S. Rajeh, M. Savonnet, E. Leclercq // Qual Quant. — 2022. — 31p. DOI: 10.1007/s11135-022-01416-7.
42. Girvan, M. Community structure in social and biological networks / M. Girvan, M. E. J. Newman // *Proc. Natl. Acad. Sci. USA*. — 2002. — 99(12). — P. 7821-7827.
43. Clauset, A. Finding community structure in very large networks / A. Clauset, M. E. J. Newman // Physical Review. — 2004. — E 70. 066111. — 6 p.
44. Radicchi, F. Defining and identifying communities in networks / F. Radicchi, C. Castellano, V. Loreto, F. Cecconi, D. Parisi // Proc. Natl. Acad. Sci. USA. — 2004. — 101(9). — P. 2658-2663.
45. Palla, G. Uncovering the overlapping community structure of complex networks in nature and society / G. Palla, I. Derenyi, I. Farkas, T. Vicsek // Nature. — 2005. — 435. — P. 814-818.
46. Newman, M. E. J. Fast algorithm for detecting community structure in networks / M. E. J. Newman // Physical Review. — 2004. — E 69. 066133. — 5 p.
47. Newman, M. E. J. Modularity and community structure in networks / M. E. J. Newman // Proc. Natl. Acad. Sci. USA. — 2006. — 103(23). — P. 8577-8582.
48. Newman, M. E. J. Finding and evaluating community structure in networks / M. E. J. Newman, M. Girvan // Physical Review. — 2004. — E 69. 026113. — 16 p.
49. Fortunato, S. 20 years of network community detection / S. Fortunato, M. E. J. Newman // Nat. Phys. — 2022. — 18. — P. 848–850.

50. Blondel, V. D. Fast unfolding of communities in large networks / V. D. Blondel, J. - L. Guillaume, R. Lambiotte, E. Lefebvre // Journal of Statistical Mechanics: Theory and Experiment. — 2008. — 10. P10008. — 12 p.
51. Rosvall, M. An information-theoretic framework for resolving community structure in complex networks / M. Rosvall, C. T. Bergstrom // Proc. Natl. Acad. Sci. USA. — 2007. — 104(18). — P. 7327-7331.
52. Rosvall, M. Maps of information flow reveal community structure in complex networks / M. Rosvall, C. T. Bergstrom // Proc. Natl. Acad. Sci. USA. — 2008. — 105(4). — P. 1118-1123.
53. Rosvall, M. The map equation/ M. Rosvall, C. T. Bergstrom, D. Axelsson // The European Physical Journal Special Topics. — 2009. — 178(1). — P. 13-23.
54. Esquivel, A. Compression of flow can reveal overlapping modular organization in networks/ A. Esquivel, M. Rosvall // Physical Review. — 2011. — X 1. 021025. — 10 p.
55. Domenico, M. Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems / M. Domenico, A. Lancichinetti, A. Arenas, M. Rosvall // Physical Review. — 2015. — X. 5. 011027. — 14 p.
56. Мазалов, В. В. Метод максимального правдоподобия для выделения сообществ в коммуникационных сетях / В. В. Мазалов, Н. Н. Никитина // Вестник Санкт-Петербургского университета. Прикладная математика. Информатика. Процессы управления. — 2018. — 14(3). — С. 200–214.
57. Fortunato, S. Resolution limit in community detection / S. Fortunato, M. Barthélemy // Proc. Natl. Acad. Sci. USA. — 2007. — 104. — P. 36-41.
58. Lancichinetti, A. Benchmark graphs for testing community detection algorithms / A. Lancichinetti, S. Fortunato, F. Radicchi // Physical Review. — 2008. — E 78. 046110. — 6 p.
59. Danon, L. Comparing community structure identification / L. Danon, A. Díaz-Guilera, J. Duch, A. Arenas // J. Stat. Mech. — 2005. — P09008. — 10 p.
60. Amelio, A. Is Normalized Mutual Information a Fair Measure for Comparing Community Detection Methods? / A. Amelio, C. Pizzuti / Proceedings of the 2015

IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015. — Association for Computing, 2015. — P. 1584 – 1585.

61. Jerdee M. Normalized mutual information is a biased measure for classification and community detection/ M. Jerdee, A. Kirkley, M. E. J. Newman // arXiv preprint arXiv:2307.01282v2, 2024. (Дата обращения: 03.09.2024).
62. Коломейченко, М. И. Методы визуального анализа графов / М. И. Коломейченко, И. В. Поляков, А. А. Чеповский, А. М. Чеповский. – М.: Национальный открытый университет «ИНТУИТ», 2016. – 167 с.
63. Лещёв, Д. А. Алгоритмы выделения групп общения / Д. А.Лещёв, Д. В. Сучков, С. П. Хайкова, А. А. Чеповский // Вопросы кибербезопасности. – 2019. – Т. 32. – № 4. – С. 61-71. [RSCI].
64. Попов, В. А. Модели импорта данных из Твиттера/ В. А. Попов, А. А. Чеповский // Вестник НГУ. Серия: Информационные технологии. – 2021. – Т. 19. – № 2. – С. 76–91. DOI 10.25205/1818-7900-2021-19-2-76-91. [05.13.18 список ВАК (К1) 2021 № 486].
65. Воронин, А. Н. Взаимосвязь сетевых характеристик и субъектности сетевых сообществ в социальной сети Твиттер / А. Н. Воронин, Ю. В. Ковалева, А. А. Чеповский // Вопросы кибербезопасности. – 2020. – Т. 37. – № 3. – С. 40-57. [RSCI].
66. Попов В. А., Чеповский А. А. Модели импорта данных из мессенджера Telegram // Вестник Новосибирского государственного университета. Серия: Информационные технологии. — 2022. — 20(2). — С. 60–71. DOI: 10.25205/1818-7900-2022-20-2-60-71. [список ВАК (К1) 2022 № 518].
67. Попов, В. А. О моделях построения графа взаимодействующих объектов в сети Telegram-каналов / В. А. Попов, А. А. Чеповский // Вопросы кибербезопасности. – 2024. – № 3 (61). – С. 105-112. DOI: 10.21681/2311-3456-2024-3-105-112. [RSCI].
68. Попов, В. А. Выделение неявных сообществ на графе взаимодействия Telegram-каналов с помощью «метода Галактик»/ В. А. Попов, А. А. Чеповский // Труды

ИСА РАН. – 2022. – Т.72. – №4. С. 39–50. DOI: 10.14357/20790279220405. [список ВАК (К1) 2022 № 2348].

69. Лобанова, С. Ю. Комбинированный алгоритм выделения сообществ в графах взаимодействующих объектов / С. Ю. Лобанова, А. А. Чеповский // Бизнес-информатика. – 2017. – Т. 42. – № 4. – С. 64-73 (Chepovskiy A., Lobanova S. Combined method to detect communities in graphs of interacting objects / Пер. с рус. // Business Informatics. – 2017. – Vol. 42. – No. 4. – P. 64-73.) [05.13.00 список ВАК (К1) 2017 и 2018 № 90].
70. Chepovskiy, A. A. Core Method for Community Detection/ A. A. Chepovskiy, D. Leshchev, S. P. Khaykova / in: Complex Networks & Their Applications IX. Volume 1: Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020. – Springer, 2021. – P. 38-50. [Scopus].
71. Чеповский, А. А. Построение и анализ графов взаимодействующих объектов / А. А. Чеповский / В кн.: Международная конференция «Математика в созвездии наук». К юбилею ректора МГУ академика Виктора Антоновича Садовничего: Тезисы докладов / Орг. комитет: В. А. Садовничий, А. И. Шафаревич, И. А. Соколов [и др.]. — Москва: Издательство Московского университета, 2024. — С. 355 - 357.
72. Чеповский, А. А. О неявных сообществах на графе взаимодействующих объектов/ А. А. Чеповский// Успехи кибернетики. – 2023. – Т.4. – № 1. – С. 56-64. DOI: 10.51790/2712-9942-2023-4-1-08. [список ВАК (К3) 2023 № 2719].
73. Чеповский, А. А. Анализ графов взаимодействующих объектов / А. А. Чеповский, – М.: Национальный открытый университет «ИНТУИТ», 2022. – 270 с.
74. Чеповский А. А. Об особенностях построения и анализа графов взаимодействующих объектов в сети Telegram-каналов/ А. А. Чеповский // Вопросы кибербезопасности. – 2023. – № 1 (53). – С. 75-81. DOI:10.21681/2311-3456-2023-1-75-81 [RSCI].
75. Коломейченко, М. И. Хранение и скачивание сетей больших размеров / М. И. Коломейченко, И. В. Поляков, А. А. Чеповский / В кн.: Труды Международной

научной конференции Resilience2014 Международного Центра по ядерной безопасности Института физико-технической информатики. – М., Протвино: Институт физико-технической информатики, 2015. – С. 139-143.

76. Михайлов, А. С. О моделях оценки информационного воздействия в социальных сетях. / А. С. Михайлов, А. А. Чеповский, А. М. Чеповский / В кн.: SCVRT2013-14 Труды Международной научной конференции Международного центра по ядерной безопасности Института физико-технической информатики. – Протвино: Изд-во ИФТИ, 2014. – С. 247-249.
77. JSON for Modern C++ [Электронный ресурс]. — Режим доступа: https://nlohmann.github.io/json/. (Дата обращения: 08.03.2020).
78. Official YAML format website/ [Электронный ресурс]. — Режим доступа https://yaml.org/ (Дата обращения 23.03.2020)
79. A YAML parser and emitter in C++. [Электронный ресурс]. — Режим доступа: https://github.com/jbeder/yaml-cpp. (Дата обращения 03.04.2020).
80. API ВКонтакте [Электронный ресурс]. — Режим доступа: https://vk.com/dev/manuals. (Дата обращения 07.03.2020).
81. Libcurl – the multiprotocol file transfer library [Электронный ресурс]. — Режим доступа: http://curl.haxx.se/ libcurl. (Дата обращения 07.03.2020).
82. Twitter API. [Электронный ресурс]. — Режим доступа: https://developer.twitter.com/en/docs/basics/getting-started. (Дата обращения: 01.02.2020).
83. Mitchell, R. Web Scraping with Python / R. Mitchell. — Sebastopol: O’Reilly Media, 2015. — 306 p.
84. Библиотека Python Selenium. [Электронный ресурс]. — Режим доступа: https://selenium-python.readthedocs.io. (Дата обращения: 01.02.2020).
85. Telegram API. [Электронный ресурс]. — Режим доступа: https://core.telegram.org/api. (Дата обращения: 01.02.2020).
86. Caldarelli, G. Scale-Free Networks / G. Caldarelli. — Oxford: Oxford University Press, 2007. — 336 p.

87. Clauset, A. Power-law distributions in empirical data / A. Clauset, C.R. Shalizi, M. E. J. Newman // SIAM Review. — 2009. — 51. — P. 661–703.
88. Chepovskiy A. A. Methods to reveal communities without the property of ”picking up junk” / A. A. Chepovskiy // In The 6 th International Conference on Complex Networks &Their Applications. Nov. 29 - Dec. 01, 2017. – Lyon (France). – P. 336-340. [Scopus].
89. Лобанова, С. Ю. О применении алгоритмов разбиения сети взаимодействующих объектов на сообщества / С. Ю. Лобанова, А. А. Чеповский / В кн.: Управление информационной безопасностью в современном обществе. Сборник научных трудов. М.: Изд. дом Высшей школы экономики, 2017. – С. 83-84.
90. Kolomeychenko, M. I. Detection of Communities in a Graph of Interactive Objects / M. I. Kolomeychenko A. A. Chepovskiy, A. M. Chepovskiy, I. V. Polyakov, // Journal of Mathematical Sciences. – 2019. – Vol. 237. – No. 3. – P. 426-431 (Коломейченко М. И., Поляков И. В., Чеповский А. А., Чеповский А. М. Выделение сообществ в графе взаимодействующих объектов // Фундаментальная и прикладная математика. – 2016. – Т. 21. – № 3. – С. 131-139). [Scopus, Q3].
91. Kolomeychenko, M. I. An Algorithm for Detecting Communities in Social Networks / M. I. Kolomeychenko, A. A. Chepovskiy, A. M. Chepovskiy /// Journal of Mathematical Sciences. – 2015. – Vol. 211. – No. 3. – P. 310-318 (Коломейченко М. И., Чеповский А. А., Чеповский А. М. Алгоритм выделения сообществ в социальных сетях // Фундаментальная и прикладная математика. – 2014. – Т. 19. – № 1. – С. 21-32) [Scopus, Q3].
92. Орлов, А. О. О свойствах модулярности и актуальных корректировках алгоритма Блонделя / А. О. Орлов, А. А. Чеповский // Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2017. – Т. 15. – № 3. – С. 64-73. [05.13.00 список ВАК (К1) 2017 № 1834].
93. Орлов, А. О. Особенности алгоритма Блонделя при выявлении сообществ в графе социальной сети / А. О. Орлов, А. А. Чеповский / В кн.: Труды Международной научной конференции Московского физико-технического института

(государственного университета) и Института физико-технической информатики (SCVRT1516). – М., Протвино: Институт физико-технической информатики, 2016. – С. 124-129.

94. Орлов, А. О. Особенности алгоритмов выделения сообществ в графах социальных сетей / А. О. Орлов, А. А. Чеповский / В кн.: Управление информационной безопасностью в современном обществе. Сборник научных трудов. М.: Изд. дом Высшей школы экономики, 2017. – С. 108-109.
95. Соколова, Т. В. Анализ профилей сообществ социальных сетей / Т. В. Соколова, А. А. Чеповский // Системы высокой доступности. – 2018. – Т. 14. – № 3. – С. 82-86. [05.13.00 список ВАК (К2) 2018 № 1852].
96. Золотых, А. А. О задаче анализа графа социальной сети / А. А Золотых, М. И. Коломейченко, А. А Чеповский / В кн.: Труды Международной научной конференции по физико-технической информатике (СРТ2014). – М., Протвино: Институт физико-технической информатики, 2015. – С. 131-134.
97. Brandes, U. Maximizing Modularity is hard / U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, D. Wagner // arXiv: physics. — 2006. — 0608255. — 10 p.
98. Donetti, L. Improved spectral algorithm for the detection of network communities / L. Donetti, M. A. Munoz // arXiv: physics. — 2005. — 0504059. — 4 p.
99. Duch, J. Community detection in complex networks using extremal optimization / J. Duch, A. Arenas //Phys. Rev. — 2005. — E 72(2). 027104. — 6 p.
100. Dugue, N. Directed Louvain: maximizing modularity in directed networks. Research Report / N. Dugue, A. Perez. — Universite d`Orleans, 2015. — hal-01231784. – 14 p.
101. Shen, H. W. Detect overlapping and hierarchical community structure in networks / H. W. Shen, X. Q. Cheng, K. Cai, M. B. Hu // Physica A: Statistical Mechanics and its Applications. – 2009. – Vol. 388. – No. 8. – P. 1706–1712.
102. Liu, H. Social Computing, Behavioral Modeling, and Prediction / H. Liu, J. Salerno, M. Young. – Springer Science, 2008. – 264 p.

103. Lovasz, L. Random Walks on Graphs: A Survey / L. Lovasz // Combinatorics. — Volume 2. — Keszthely (Hungary). — 1993. — P. 1–46.

104. Lambiotte, R. Ranking and clustering of nodes in networks with smart teleportation / R. Lambiotte, M. Rosvall // Physical Review. – 2012. – E 85. 056107. – 10 p.

105. Шеннон, К. Э. Математическая теория связи // Работы по теории информации и кибернетике / К. Э. Шеннон. — М.: ИИЛ, 1963. — С. 243—332.

106. Lancichinetti, A. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities / A. Lancichinetti, S. Fortunato // Physical Review. – 2009. – E 80. 016118. – 9 p.

107. Lancichinetti, A. Community detection algorithms: a comparative analysis / A. Lancichinetti, S. Fortunato // Physical Review. – 2009. – E 80. 056117. – 12 p.

108. Palla, G. k-Clique percolation and clustering / G. Palla, D. Abel, I. J. Farkas, P. Pollner, I. Dernyi, T. Vicsek // Handbook of Large-scale Random Networks. – Springer, 2009. – Ch. 9. – P. 1–40.

109. Gregory, S. Fuzzy overlapping communities in networks / S. Gregory // Journal of Statistical Mechanics: Theory and Experiment. – 2011. – Vol. 2011. – No. 02. – P. 1–18.

110. Gregory, S. An algorithm to find overlapping community structure in networks / S. Gregory / Proceedings of Knowledge Discovery in Databases: PKDD 2007: 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, 17–21 September 2007. Berlin, Heidelberg: Springer. – 2007. – Vol. 4702. – P. 91–102.

111. Collins, L. M. Omega: A general formulation of the Rand index of cluster recovery suitable for non-disjoint solutions / L. M. Collins, C. W. Dent // Multivariate Behavioral Research. – 1988. – Vol. – 23. – No. 2. – P. 231–242.

112. Субъектность и жизнеспособность сетевых сообществ в дискурсивном пространстве Интернета/ Алдашева А.А., Воронин А.Н., Гребенщикова Т.А., Китова Д.А., Ковалева Ю.В., Кубрак Т.А., Латынов В.В., Нестик Т.А., Павлова Н.Д., Рунец О.В., Смирнов И.В., Станкевич М.А., Чеповский А.А. – М.: Изд-во «Институт психологии РАН», 2021 – 373 с.

113. Аванесян, Н. Л. Характеристики текстов сообществ социальных сетей/ Н. Л. Аванесян, Ф. Н. Соловьев, А. А. Чеповский // Вестник НГУ. Серия: Информационные технологии. – 2021. – Т.19. – №1. – С. 5–14. DOI: 10.25205/1818-7900-2021-19-1-5-14 [05.13.18 список ВАК (К1) 2021 № 486].

114. Kelley, S. The existence and discovery of overlapping communities in large-scale networks. Ph.D. thesis, Rensselaer Polytechnic Institute / S. Kelley – Troy, NY, 2009.

115. Yang, J. Overlapping Community Detection at Scale: A Nonnegative Matrix Factorization Approach / J. Yang, J. Leskovec /in 'Proceedings of the Sixth ACM International Conference on Web Search and Data Mining', ACM, New York, NY, USA, 2013. – P. 587-596.

116. Поляков, И. В. Проблема классификации текстов и дифференцирующие признаки / И. В. Поляков, Т. В. Соколова, А. А. Чеповский, А. М. Чеповский // Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2015. – Т. 13. – № 2. – С. 55-63. [список ВАК (К1) 2015 (до 30.06) № 365].

117. Михайлов, А. С. Выявление тематической направленности текстов на естественных языках / А. С. Михайлов, Т. В. Соколова, А. А. Чеповский, А. М. Чеповский // Искусственный интеллект и принятие решений. – 2016. – № 1. – С. 9-17. [05.13.00 список ВАК (К1) 2016 № 718].

118. Фокина, А. И. Использование платформы ТХМ корпусного анализа для анализа текстов сообществ социальных сетей / А. И. Фокина, А. А. Чеповский, А. М. Чеповский// Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2023. – Т.21. – №2. С. 29–38. DOI: 10.25205/1818-7900-2022-21-2-29-38. [список ВАК (К2) 2023 № 583].

119. Аванесян, Н. Л. Анализ текстов сообществ социальных сетей / Н. Л. Аванесян, В. В. Зенькова, А. А. Чеповский, А. М. Чеповский // Успехи кибернетики. – 2023. – Т.4. – № 2. – С. 33-39. DOI: 10.51790/2712-9942-2023-4-2-05. [список ВАК (К3) 2022 № 2664].

120. Михайлов, А. С. Методика выявления нарушений в текстах Интернета / А. С. Михайлов, Т. В. Соколова, А. А. Чеповский, А. М. Чеповский / В кн.: Труды

Международной научной конференции по физико-технической информатике (CPT2014). М., Протвино: Институт физико-технической информатики, 2015. – С. 115-119.

121. Поляков, И. В. Задача распознавания для текстов на естественных языках / И. В. Поляков, Ф. Н. Соловьев, А. А. Чеповский, А. М. Чеповский. – М.: Национальный открытый университет «ИНТУИТ», 2017. – 119 с.
122. Соловьев, Ф. Н. Автоматическая обработка текстов на основе платформы ТХМ с учетом анализа структурных единиц текста / Ф. Н. Соловьев // Вестник НГУ. Серия: Информационные технологии. – 2020. – Т. 18. – №1. – С. 74–82.
123. Egorova, E. A structural pattern based method for automated morphological analysis of word forms in a natural language / E. Egorova, A. Chepovskiy, A. Lavrentiev // Journal of Mathematical Sciences. – 2016. – Vol. 214. – No. 6. – P. 802-813.
124. Чеповский, А.М. Информационные модели в задачах обработки текстов на естественных языках. Второе издание, переработанное / А.М. Чеповский. – М.: Национальный открытый университет «ИНТУИТ», 2015. – 228 с.
125. Бендат, Дж. Прикладной анализ случайных данных / Дж. Бендат, А. Пирсол. – М.: Мир, 1989. –540 с.
126. Воронин, А. Н. Субъектность сетевого сообщества: сравнение психометрических моделей проявления дискурсивных маркеров в контенте / А. Н. Воронин, Т. А. Гребенщикова, Т. А. Кубрак, Н.Д. Павлова //Вестник Московского государственного областного университета. Серия: Психологические науки. – 2019. – № 3. – С. 6-24.
127. Voronin A. N., Grebenschikova T. A., Kubrak T. A., Nestik T. A., Pavlova N. D. THE STUDY OF NETWORK COMMUNITY CAPACITY TO BE A SUBJECT: DIGITAL DISCURSIVE FOOTPRINTS // Behavioral Sciences. – 2019. – Т. 9. – № 12. – P. 119.
128. Павлова, Н. Д. Разработка подхода к типологии сетевых сообществ на основе дискурсивных признаков коллективной субъектности / Н.Д. Павлова, А. Н. Воронин, Т. А. Гребенщикова, Т. А. Кубрак // Вестник Российского университета

дружбы народов. Серия: Психология и педагогика. – 2019. – Т. 16. – № 3. – С. 341-358.

129. Воронин, А. Н. Изменение субъектности сетевого сообщества в процессе троллинга / А. Н. Воронин, Ю. В. Ковалева // Социальная и экономическая психология. Институт психологии Российской академии наук. – 2019. – Т. 4. – № 3 (15). – С. 25-61.

130. Коломейченко, М. И. Автоматическое размещение графа на основе метода физических аналогий / М. И. Коломейченко, И. В. Поляков, А. А. Чеповский / В кн.: Труды Международной научной конференции Московского физико-технического института (государственного университета) и Института физико-технической информатики (SCVRT1516). М., Протвино: Институт физико-технической информатики, 2016. – С. 93-97.

131. Поляков, И. В. Хранение и обработка графа социальных сетей / И. В. Поляков, А. А. Чеповский, А. М. Чеповский // Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2013. – Т. 11. – № 4. – С. 77-83. 14 [список ВАК (К1) 2013 № 365].

132. Поляков, И. В. Особенности хранения графов социальной сети / И. В. Поляков, В. И. Полякова, А. А. Чеповский // Системы высокой доступности. – 2018. – Т. 14. – № 3. – С. 63-67. [05.13.00 список ВАК (К2) 2018 № 1852].

133. Polyakov, I. V. Data Compression in Big Graph Warehouse / Пер. с рус. / I. V. Polyakov, A. A. Chepovskiy, A. M. Chepovskiy // Journal of Mathematical Sciences. – 2020. – Vol. 245. – P. 197-201. DOI:10.1007/s10958-020-04686-4 (Поляков И. В., Чеповский А. А., Чеповский А. М. Сжатие данных в хранилище больших графов // Фундаментальная и прикладная математика. – 2016. – Т. 21. – № 4. – С. 125-132) [Scopus, Q3].

134. Polyakov, I. V. Algorithms for Searching Paths in Huge Graphs / I. V. Polyakov, A. A. Chepovskiy, A. M. Chepovskiy // Journal of Mathematical Sciences. – 2015. – Vol. 211. – No. 3. – P. 413-417. (Поляков И. В., Чеповский А. А., Чеповский А. М. Алгоритмы поиска путей на графах большого размера // Фундаментальная и прикладная математика. – 2014. – Т. 19. – № 1. – С. 165-172). [Scopus, Q3].

135. Поляков, И. В. Буферизация и сжатие данных при хранении мультиграфа / И. В. Поляков, А. А. Чеповский / В кн.: Труды Международной научной конференции Московского физико-технического института (государственного университета) и Института физико-технической информатики (SCVRT1516). – М., Протвино: Институт физико-технической информатики, 2016. – С. 76-78.
136. Коломейченко, М. И. О хранении графа социальной сети / М. И. Коломейченко, И. В. Поляков, А. А. Чеповский, А. М. Чеповский / В кн.: Труды Международной научной конференции по физико-технической информатике (СРТ2015). – М., Протвино: Институт физико-технической информатики, 2016. – С. 175-178.
137. Доронин, А. И. Бизнес-разведка / А. И. Доронин. – М.: Ось-89, 2010. – 704 с.
138. Cytoscape. [Электронный ресурс]. – Режим доступа: http://www.cytoscape.org/. (Дата обращения: 07.12.2021).
139. Tulip. [Электронный ресурс]. – Режим доступа: http://tulip.labri.fr. (Дата обращения: 07.12.2021).
140. VisuaLyzer [Электронный ресурс]. – Режим доступа: http://socioworks.com/productsall/visualyzer (Дата обращения: 07.12.2021).
141. i2 Analyst's Notebook. [Электронный ресурс]. – Режим доступа: https://i2group.com/. (Дата обращения: 07.12.2021).
142. CrimeLink. [Электронный ресурс]. – Режим доступа: https://www.crimelink.co.uk/. (Дата обращения: 06.12.2021).
143. XAnalys Link Explorer. [Электронный ресурс]. – Режим доступа: http://www.xanalys.com/solutions/linkexplorer.html. (Дата обращения: 07.12.2021).
144. Group-IB Графовый анализ [Электронный ресурс]. – Режим доступа: https://www.group-ib.ru/media/graph/ (Дата обращения: 06.12.2021).
145. Sentinel Visualyzer. [Электронный ресурс]. – Режим доступа: http://www.fmsasg.com/ (Дата обращения: 07.12.2021).
146. Gephi. [Электронный ресурс]. – Режим доступа: https://gephi.org/. (Дата обращения: 07.12.2021).

147. NetMiner 4. [Электронный ресурс]. – Режим доступа: http://www.netminer.com/main/main-read.do. (Дата обращения: 07.12.2021).
148. yEd. [Электронный ресурс]. – Режим доступа: http://www.yworks.com/products/yed. (Дата обращения: 07.12.2021).
149. Графоанализатор. [Электронный ресурс]. – Режим доступа: http://grafoanalizator.unick-soft.ru/. (Дата обращения: 06.12.2021).
150. Касьянов, В. Н. Visual Graph – система визуализации сложноструктурированной информации большого объема на основе графовых моделей / В. Н. Касьянов, Т. А. Золотухин / 25-я Международная конференция по компьютерной графике GraphiCon2015, 2015. – С. 154–163.
151. aiSee [Электронный ресурс]. – Режим доступа: https://www.absint.com/aisee/. (Дата обращения: 06.12.2021).
152. Tom Sawyer. [Электронный ресурс]. – Режим доступа: http://www.tomsawyer.com. (Дата обращения: 07.12.2021).
153. GraphViz. [Электронный ресурс]. – Режим доступа: http://www.graphviz.org/. (Дата обращения: 07.12.2021).
154. Igraph software. [Электронный ресурс]. – Режим доступа: http://igraph.org/. (Дата обращения: 07.12.2021).
155. Qt documentation. [Электронный ресурс]. – Режим доступа: https://doc.qt.io. (Дата обращения: 06.04.2020).
156. Робинсон, Ян Графовые базы данных: новые возможности для работы со связанными данными / Ян Робинсон, Джим Вебер, Эмиль Эифрем. -2 изд. – М.: ДМК Пресс, 2016. — 256 с.
157. Фаулер, Мартин NoSQL: новая методология разработки нереляционных баз данных / Мартин Фаулер, Дж Садаладж Прамодкумар. — М.: ООО «И.Д.Вильямс», 2017. — 192 с.
158. Angles, R. A Comparison of Current Graph Database Models / R. Angles / Proceedings of the 2012 IEEE 28th International Conference on Data Engineering Workshops, ICDEW '12, IEEE Computer Society. Washington, DC, USA, 2012. – P. 171

159. PostgreSQL [Электронный ресурс]. – Режим доступа: https:// https://www.postgresql.org/. (Дата обращения 16.03.2022).
160. MySQL documentation. [Электронный ресурс]. – Режим доступа: https://dev.mysql.com/doc/. (Дата обращения 16.03.2022).
161. Марчук, А. Г. PolarDB — система создания специализированных NoSQL баз данных и СУБД / А. Г. Марчук // Моделирование и анализ информационных систем. – 2014. – Том 21. – № 6. – 169–175.
162. Бэнкер, Кайл MongoDB в действии / Пер. с англ / Кайл Бэнкер. — М.: ДМК Пресс, 2014. — 394 с.
163. Győrödi, Cornelia A comparative study: MongoDB vs. MySQL / Cornelia Győrödi, et al. / 13th International Conference on Engineering of Modern Electric Systems (EMES). IEEE, 2015. — 8 p.
164. MongoDB documentation, [Электронный ресурс]. – Режим доступа: https://www.mongodb.com/docs/. (Дата обращения 17.03.2022).
165. Batra, S. Comparative Analysis of Relational And Graph Databases / S. Batra, C. Tyagi //International Journal of Soft Computing and Engineering (IJSCE). — 2012. — Volume 2. — Issue 2. — P. 509 — 512.
166. DB-Engines Ranking of Graph DBMS [Электронный ресурс]. – Режим доступа: https://db-engines.com/en/ranking/graph+dbm/. (Дата обращения 15.05.2022).
167. Shrinivas, S. G. Applications of graph theory in computer science an overview / S. G. Shrinivas, et. al. // International Journal of Engineering Science and Technology. — 2010. — Vol. 9. — P. 4610 — 4621.
168. OrientDB documentation. [Электронный ресурс]. – Режим доступа: https://orientdb.org/docs/3.0.x/. (Дата обращения 12.03.2022).
169. OrientDB Orient Technologies – OrientDB Distributed Graph Database. [Электронный ресурс]. – Режим доступа: http://www.orientechnologies.com/orientdb/. (Дата обращения: 15.02.2020).
170. Webber, Jim A programmatic introduction to Neo4j / Jim Webber / In Proceedings of the 3rd annual conference on Systems, programming, and applications: software

for humanity (SPLASH '12). Association for Computing Machinery, New York, NY, USA, 2012. – P. 217–218. DOI: 10.1145/2384716.2384777

171. Neo4j - The World's Leading Graph Database. [Электронный ресурс]. – Режим доступа: http://www.neo4j.org/ (Дата обращения: 15.02.2020).
172. Neo4J Official documentation. [Электронный ресурс]. – Режим доступа: https://neo4j.com/product/neo4j-graph-database/. Дата обращения 12.02.2022
173. ArangoDB documentation. [Электронный ресурс]. – Режим доступа: https://www.arangodb.com/docs/stable/. (Дата обращения 15.03.2022).
174. Sparksee (graph database). [Электронный ресурс]. – Режим доступа: https://www.sparsity-technologies.com/. (Дата обращения: 15.02.2020).
175. Kang. U. GBASE: a scalable and general graph management system / U. Kang, Tong Hanghang, Sun Jimeng, Ching Yung Lin, Christos Faloutsos / Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, August 21— 24, 2011, San Diego, California, USA . — 9 p.
176. Mahoney, M. Adaptive Weighing of Context Models for Lossless Data Compression. Technical Report CS-2005–16 / M. Mahoney. — Florida Institute of Technology CS Department: Melbourne, FL, USA, 2005. — 8 p.
177. Peter, P. Evaluating the true potential ofdiusion-based inpainting in a compression context / P. Peter, S. Homann, F. Nedwed, L. Hoeltgen, J. Weickert // Signal Processing: Image Communication. – 2016. – №46. – P. 40— 53.
178. Network Repository. [Электронный ресурс]. – Режим доступа: http://networkrepository.com/socfb-A-anon.php/. (Дата обращения: 15.02.2020).
179. Программа для импорта и построения графов на основе данных из сети мгновенного обмена сообщениями. Свидетельство о государственной регистрации программы ЭВМ № 2024680205 Российская Федерация. Дата регистрации 27.08.2024 / В. А. Попов, А. А. Чеповский.
180. Определение характеристик корпусов текстов и сравнения корпусов корреляционным анализом. Свидетельство о государственной регистрации программы ЭВМ № 2024680546 Российская Федерация. Дата регистрации 30.08.2024 / Н. Л. Аванесян, А. А. Чеповский, А. М. Чеповский.

ПРИЛОЖЕНИЕ 1. ФОРМАТ AVS-ФАЙЛА

Программный комплекс *AVS* (*Analytics and Visualization System for graphs)* использует для импорта и экспорта данных собственный формат AVS-файлов, который, в свою очередь, основан на формате GraphML.

Первая строка файла:

```
<?xml version="1.0" encoding="utf-8"?>
```

Далее идет вложенная система парных тегов. Первый корневой элемент:

```
<avs>
```

Внутри него идет две пары тегов. Первая содержит перечисление используемых в файле типов вершин и ребер:

```
<description> и </description>
```

Вторая содержит непосредственное перечисление списка вершин и ребер. Как атрибут этого тега указывается тип графа. Поддерживается вариант как ориентированных, так и не ориентированных графов:

```
<graph edgedefault="undirected">
    и
</graph>

<graph edgedefault="directed">
    и
</graph>
```

Файл завершается записью:

```
</avs>
```

П1.1 Описание основных элементов

```
<avs></avs>
```

Описание:

<avs> корневой элемент

Потомки:

<description></description>

<graph></graph>

Атрибуты:

нет

<description></description>

Описание:

<description> посредством использования вложенного элемента <type> описывает все возможные типы вершин и связей. Каждому типу вершин и связей соответствует один вложенный элемент <type>

Потомки:

<type></type>

Атрибуты:

нет

<type></type>

Описание:

<type> дает описание для типа вершины или связи через вложенные элементы <key/>. В атрибутах самого тега указывается, что это за тип.

Потомки:

<key/>

Атрибуты:

id

Описание:

идентификатор типа вершины или связи. Должен быть уникальным среди элементов типа <type>

Тип:

строковый, латинские символы и цифры.

for

Описание:

указание, к чему относится тип — к вершине или к ребру

Тип:

строковый, возможные значения: "edge" или "node".

<key/>

Описание:

<key/> задает атрибут и его свойства для родительского типа <type>

Потомки:

нет

Атрибуты:

id

Описание:

идентификатор атрибута. Должен быть уникальным среди элементов типа <key/>

Тип:

строковый, латинские символы и цифры.

attr.name

Описание:

название атрибута

Тип

строковый

attr.type

Описание:

тип данных атрибута

Тип:

строковый, возможные значения:

1. int
2. string

3. datetime
4. date
5. time
6. double

<graph></graph>

Описание:

предоставляет описание сети, сохраненной в формате AVS

Потомки:

<node></node>

<edge></edge>

Атрибуты:

edgedefault

Описание: определяет ориентированные/неориентированные графы

Тип

строковый, возможные значения:

"undirected"

"directed"

<node></node>

Описание:

описывает вершину сети.

Потомки:

<data></data>

Атрибуты:

id

Описание:

идентификатор вершины. Должен быть уникальным среди элементов типа <node>

Тип

строковый, латинские символы и цифры

type

Описание:

тип вершины из списка типов (идентификаторов) вершин, определённых в теге <description>

Тип

внешний ключ из числа значений атрибута id в тегах <type> внутри <description>

Первичный ключ:

<type>.id

Ограничения:

значение атрибута for в теге <type> у вершины, с данным ключом должно быть равным "node"

<edge></edge>

Описание:

описывает связь между вершинами сети.

Потомки:

<data></data>

Атрибуты:

Id

Описание:

идентификатор ребра. Должен быть уникальным среди элементов типа <edge>

Тип

строковый, латинские символы и цифры

type

Описание:

тип связи из списка типов (идентификаторов) связей, определённых в элементе <description>

Тип:

внешний ключ из числа значений атрибута id в тегах <type> внутри <description>

Первичный ключ:

<type>.id

Ограничения:

значение атрибута for в теге <type> у связи с данным ключом должно быть равным "edge"

source

Описание:

идентификатор начальной вершины текущей связи.

Тип:

внешний ключ

Первичный ключ:

<node>.id

target

***Описание*:**

идентификатор конечной вершины текущей связи.

***Тип*:**

внешний ключ

Первичный ключ:

<node>.id

<data></data>

Описание:

задает значение атрибута для конкретной вершины <node>или ребра <edge>

Потомки:

нет

Атрибуты:

Key

Описание:

идентификатор атрибута из списка определённых в элементах <type>

Тип:

внешний ключ

Первичный ключ:

<key/>.id

Ограничения:

идентификатор id родительского элемента <type> для описания типа вершины <key/>, соответствующий данному ключу, должен совпадать со значением атрибута type родительского элемента <node> или <edge>.

Механизм CDATA

В случае если значение атрибута содержит не поддерживаемые форматом XML символы, следует экранировать их используя стандартный механизм CDATA:

Пример:

<data key="a15"><![CDATA[2024-03-03 19:26:35]]></data>

Рекомендуется экранировать так любые строковые атрибуты.

Ссылки на внешние данные

Слишком большие атрибуты (тексты или бинарные данные) следует хранить в отдельных файлах. В таком случае при задании значения данного атрибута следует добавить атрибут external и экранировать содержимое, в котором необходимо указать относительный путь к файлу

external="text"

или

external="binary"

Примеры:

<data key ="a15" external="text"><![CDATA[/texts/example.txt]]></data>

```
<data key ="a16" external="binary"><![CDATA[/img/example.png]]></data>
```

П1.2 Пример содержимого файла

Ниже приведен пример AVS-файла, в котором импортируется и экспортируется между модулями программного комплекса граф:

```
<?xml version="1.0" encoding="utf-8"?>
<avs>
  <description>
    <type id=" channel" for="node">
      <key attr.name=" ChannelID" attr.type="string" id=" NodeAttr1"/>
      <key attr.name=" texts" attr.type=" string " id=" NodeAttr2"/>
    </type>
    <type id="link" for="edge">
      <key attr.name="weight" attr.type="int" id="EdgeAttr1"/>
    </type>
  </description>
 <graph edgedefault="undirected">
  <node id="channel_id0" type="ChannelID">
    <data key="NodeAttr1"><![CDATA[@channel_M]></data>
    <data key="NodeAttr2"><![CDATA[M_text.yaml]></data>
  </node>
  <node id="channel_id1" type="ChannelID">
    <data key="NodeAttr1"><![CDATA[@channel_I]></data>
    <data key="NodeAttr2"><![CDATA[I_text.yaml]></data>
  </node>
  <node id="channel_id2" type="ChannelID">
    <data key="NodeAttr1"><![CDATA[@channel_A]></data>
    <data key="NodeAttr2"><![CDATA[A_text.yaml]></data>
```

```
    </node>
    <node id="channel_id3" type="ChannelID">
     <data key="NodeAttr1"><![CDATA[@channel_V]></data>
     <data key="NodeAttr2"><![CDATA[V_text.yaml]></data>
    </node>
    <node id="channel_id3" type="ChannelID">
     <data key="NodeAttr1"><![CDATA[@channel_S]></data>
     <data key="NodeAttr2"><![CDATA[S_text.yaml]></data>
    </node>
    <edge id="e0" source="channel_id1" target="channel_id0" type=" link">
     <data key="EdgeAttr1">10</data>
    </edge>
    <edge id="e1" source="channel_id1" target="channel_id2" type=" link">
     <data key="EdgeAttr1">8</data>
    </edge>
    <edge id="e2" source="channel_id2" target="channel_id0" type=" link">
     <data key="EdgeAttr1">7</data>
    </edge>
    <edge id="e3" source=" channel_id2" target=" channel_id3" type=" link">
     <data key="EdgeAttr1">3</data>
    </edge>
    <edge id="e3" source=" channel_id2" target=" channel_id4" type=" link">
     <data key="EdgeAttr1">2</data>
    </edge>
   </graph>
</avs>
```

Конец файла.

При визуализации данного примера средствами AVS получается граф из 5 вершин и 5 ребер с соответствующими весами на ребрах и заданными текстовыми атрибутами (рисунок П1.1).

Рисунок П1.1. Визуализация примера

ПРИЛОЖЕНИЕ 2. СВИДЕТЕЛЬСТВА О РЕГИСТРАЦИИ ПРОГРАММ ДЛЯ ЭВМ

Имеются два свидетельства о регистрации программ для ЭВМ:

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024680205

Программа для импорта и построения графов на основе данных из сети мгновенного обмена сообщениями

Правообладатели: ***Попов Владимир Александрович (RU), Чеповский Александр Андреевич (RU)***

Авторы: ***Попов Владимир Александрович (RU), Чеповский Александр Андреевич (RU)***

Заявка № **2024669156**

Дата поступления **14 августа 2024 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***27 августа 2024 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Ю.С. Зубов

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024680546

Определение характеристик корпусов текстов и сравнения корпусов корреляционным анализом.

Правообладатели: ***Аванесян Нина Левоновна (RU), Чеповский Александр Андреевич (RU), Чеповский Андрей Михайлович (RU)***

Авторы: ***Аванесян Нина Левоновна (RU), Чеповский Александр Андреевич (RU), Чеповский Андрей Михайлович (RU)***

Заявка № **2024669508**

Дата поступления **19 августа 2024 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***30 августа 2024 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Ю.С. Зубов