Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ
НА ТЕМУ:

методы визуализации больших графов в интерактивном режиме с функциональностью выбора вершины курсором

Студент _____ Жаринов Михаил Андреевич
Руководитель _____ Волкова Лилия Леонидовна

Нормоконтролёр 24. 12. 2025 МАЛЬЦЕВА Д.Ю.

Рекомендуемая оценка: отлично

2025 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой ИУ-7
(Индекс)
_____ И. В. Рудаков
(И.О.Фамилия)
«03» 10 2025 г.

З А Д А Н И Е
на выполнение научно-исследовательской работы

по теме

методы визуализации больших графов в интерактивном режиме с функциональностью выбора вершины курсором

Студент группы ИУ7-52Б

Жаринов Михаил Андреевич

Направленность НИР

учебная

Источник тематики

НИР кафедры

График выполнения НИР: 25% к 6 нед., 50% к 9 нед., 75% к 12 нед., 100% к 15 нед.

Техническое задание

Ввести базовые определения предметной области. Выделить основные задачи предметной области. Построить классификацию выделенных задач предметной области/ Проанализировать не менее 30 существующих программных решений. Сформулировать условия и критерии сравнения найденных программных решений. Построить сравнительную таблицу для найденных решений на основе сформулированных условий и критериев. Подготовить демонстрационный материал для защиты НИР. Подготовить окончательную версию отчета по НИР в соответствии с требованиям нормоконтроля и получить процент заимствования. Получить экспертное заключение на материалы НИР. Получить экспортное заключение для участия в конкурсе НИР.

Оформление научно-исследовательской работы:

Расчетно-пояснительная записка на **12-30** листах формата А4 с указанием по тексту ссылок на литературу. Список литературы не менее 20 источников.

Дата выдачи задания «03 »10 2025 г.

Руководитель НИР _____ 03.10.25 Волкова Лилия Леонидовна
(Подпись, дата) (Фамилия имя отчество)

Студент _____ 03.10.25 Жаринов Михаил Андреевич
(Подпись, дата) (Фамилия имя отчество)

РЕФЕРАТ

Расчётно-пояснительная записка 39 страниц, 7 рисунков, 2 таблицы, 22 источника.

ВИЗУАЛИЗАЦИЯ ГРАФОВ, БОЛЬШИЕ ДАННЫЕ, СИЛОВЫЕ АЛГОРИТМЫ, СНИЖЕНИЕ РАЗМЕРНОСТИ, ИНТЕРАКТИВНЫЙ РЕЖИМ, ВЫБОР ВЕРШИН, КЛАСТЕРИЗАЦИЯ, НАВИГАЦИЯ, ГРАФОВЫЕ ЭМБЕДДИНГИ.

Объект исследования — методы и алгоритмы визуализации и интерактивного взаимодействия с графовыми структурами большой размерности.

Цель работы — проанализировать существующие методы интерактивной визуализации больших графов. Поставленная цель достигается путём анализа предметной области, анализа и классификации 30 существующих программных решений алгоритмов укладки и способов интерактивного выбора вершин курсором.

В данной работе проведено исследование 20 методов визуализации от классических силовых алгоритмов до современных подходов на основе машинного обучения. Рассмотрены методы борьбы с информационной перегрузкой, такие как кластеризация и связывание ребер. Исследованы 10 методов навигации и выбора вершин курсором в условиях высокой плотности объектов.

В результате выполнения данной работы были классифицированы существующие методы укладки и агрегации графов по критериям: быстродействие, тип графа, устойчивость, выделение кластеров. Были классифицированы существующие методы интерактивности и навигации по критериям: точность выбора вершины, сохранение контекста, тип.

СОДЕРЖАНИЕ

РЕФЕРАТ **3**

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ **6**

ВВЕДЕНИЕ **7**

1 Анализ предметной области **9**

1.1 Понятие визуализации графов и основные задачи 9

1.2 Классификация подходов к пространственной укладке . . . 9

1.2.1 Силовые подходы 10

1.2.2 Спектральные подходы 10

1.2.3 Методы снижения размерности 10

1.2.4 Многоуровневые подходы 11

1.3 Агрегация и упрощение визуальной сложности 11

1.4 Проблематика интерактивного взаимодействия и селекции . 11

2 Обзор соответствующих программных решений **13**

2.1 Методы пространственной укладки и агрегации графов . . . 13

2.1.1 Алгоритм Фрухтермана — Рейнгольда 13

2.1.2 Алгоритм Камады — Каваи 14

2.1.3 Метод мажоризации напряжений 15

2.1.4 Метод t-SNE 16

2.1.5 Метод LargeVis 16

2.1.6 Метод Node2Vec 18

2.1.7 Метод VERSE 19

2.1.8 Упрощенная графовая свертка 21

2.1.9 Иерархическое связывание ребер 22

2.1.10 Топологический рыбий глаз 22

2.1.11 Алгоритм FM^3 23

2.1.12 Алгоритм OpenOrd 24

2.1.13 Метод LinLog 24

2.1.14 Размещение «Линия темы» 25

2.1.15 Матричное упорядочение SlashBurn 26

2.1.16 Алгоритм ForceAtlas2 26
2.1.17 Алгоритм sfdp 27
2.1.18 Метод HDE 28
2.1.19 Генетические алгоритмы укладки 28
2.1.20 Гиперболическое дерево 29
2.2 Методы интерактивного взаимодействия и выбора вершин . 30
2.2.1 Семантическое масштабирование 30
2.2.2 Искажение «Рыбий глаз» 30
2.2.3 Эксцентричные метки 30
2.2.4 Инкрементальный просмотр 31
2.2.5 Связывание и кисть 31
2.2.6 Волшебные линзы 31
2.2.7 Лассо-выделение 31
2.2.8 Приближение и переход 32
2.2.9 Метафора «Резиновый жгут» 32
2.2.10 Динамические запросы 32

3 Сравнительный анализ существующих решений 33
3.1 Сравнительный анализ методов укладки и агрегации графов 33
3.2 Сравнительный анализ методов навигации и интерактивности 35

ЗАКЛЮЧЕНИЕ 36

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 39

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящем отчёте о НИР применяют следующие сокращения и обозначения.

— PCA — метод главных компонент

— MDS — многомерное шкалирование

— t-SNE — стохастическое вложение соседей с t-распределением

— KNN — метод k-ближайших соседей

— GCN — графовые сверточные сети

— SGC — упрощенная графовая свертка

— FM^3 — быстрый мультипольный многоуровневый метод

— HDE — вложение высокой размерности

— BFS — поиск в ширину

— DFS — поиск в глубину

— LCA — наименьший общий предок

— GPU — графический процессор

— FPS — количество кадров в секунду

— KL-дивергенция — дивергенция Кульбака — Лейблера

ВВЕДЕНИЕ

Графы являются универсальным языком для описания сложных систем взаимосвязанных объектов в самых различных предметных областях: от социальных сетей и биоинформатики до транспортных систем и компьютерных сетей. Визуализация таких структур играет ключевую роль в анализе данных, позволяя исследователям выявлять скрытые паттерны, кластеры и аномалии, которые невозможно обнаружить при изучении табличных данных или статистических метрик.

С наступлением эры больших данных размерность анализируемых графов выросла до десятков тысяч и миллионов вершин. В этих условиях классические подходы к визуализации сталкиваются с двумя фундаментальными проблемами. Первая — алгоритмическая и вычислительная сложность: отрисовка и пересчет укладки в реальном времени требуют значительных ресурсов. Вторая — проблема когнитивного восприятия: плотные графы превращаются в нечитаемый «волосяной шар», где невозможно различить отдельные элементы.

В контексте работы с большими графами статические изображения теряют свою информативность. Критически важным становится интерактивный режим, обеспечивающий навигацию, фильтрацию и детализацию по требованию. Одной из центральных задач интерфейса в таких системах является предоставление функциональности выбора вершины курсором. При отображении миллионов объектов задача быстрого определения элемента под курсором и визуального отклика системы (подсветки, вывода информации) становится нетривиальной технической проблемой, требующей применения пространственных индексов, GPU-ускорения и специализированных алгоритмов.

Цель данной научно-исследовательской работы — проанализировать существующие методы интерактивной визуализации больших графов. Для достижения поставленной цели необходимо выполнить следующие задачи:

— провести анализ предметной области, ввести базовые определения области визуализации графов и обозначить ключевые проблемы отображения больших массивов данных;

— проанализировать 30 существующих программных решений алгоритмов укладки и способов интерактивного выбора вершин курсором;

— сформулировать условия и критерии сравнения найденных программных решений;

— построить сравнительную таблицу для найденных решений на основе сформулированных условий и критериев.

1 Анализ предметной области

1.1 Понятие визуализации графов и основные задачи

Граф $G = (V, E)$, где V — множество вершин (узлов), а E — множество ребер (связей), является фундаментальной абстракцией для моделирования систем со сложной структурой взаимодействий. Задача визуализации графа заключается в построении отображения $M : V \to \mathbb{R}^k$ (обычно $k = 2$ или $k = 3$), которое сопоставляет каждому объекту координату в визуальном пространстве, и отображения ребер в виде геометрических примитивов (линий, кривых), соединяющих соответствующие вершины [1].

В контексте «больших данных» (Big Data) под графами большой размерности обычно понимают структуры, содержащие от 10^4 до 10^8 элементов [2]. При работе с такими объемами классическая задача «рисования графов» трансформируется в задачу визуальной аналитики, где ключевыми становятся не столько критерии читаемости (количество пересечений ребер графа, площадь конечного изображения, количество изгибов ребер, наименьший угол между ребрами), сколько сохранение топологических свойств и выявление глобальной структуры данных (кластеров, аномалий) [3].

Основной вызов при визуализации больших графов заключается в противоречии между ограниченным разрешением устройств вывода (экранов) и когнитивными способностями человека с одной стороны, и объемом отображаемой информации с другой. Это приводит к феномену «визуального шума» или эффекту «волосяного шара», когда плотное переплетение ребер делает невозможным извлечение полезной информации без применения алгоритмов укладки и агрегации.

1.2 Классификация подходов к пространственной укладке

Пространственная укладка — это процесс назначения координат вершинам графа. Для больших графов алгоритмы укладки принято классифицировать не по конкретным реализациям, а по математическим принципам,

лежащим в их основе [4].

1.2.1 Силовые подходы

Данный класс алгоритмов моделирует граф как физическую систему. Вершины рассматриваются как материальные тела, на которые действуют силы отталкивания (например, электростатические), а ребра — как пружины, создающие силы притяжения. Целью алгоритма является поиск равновесного состояния системы, соответствующего минимуму потенциальной энергии. Достоинством подхода является способность выявлять симметрии и локальные кластерные структуры без предварительного обучения. Однако наивная реализация расчета сил требует вычислений сложности $O(|V|^2)$, что делает их неприменимыми для больших графов без использования техник пространственной аппроксимации.

1.2.2 Спектральные подходы

Спектральные методы базируются на линейной алгебре, в частности, на разложении матриц, ассоциированных с графом (матрицы смежности, лапласиана). Координаты вершин получаются из собственных векторов, соответствующих определенным собственным числам этих матриц. Эти методы отличаются высокой скоростью работы и детерминированностью результата. Они эффективны для выявления глобальной структуры графа, однако часто дают результаты с высокой плотностью перекрытия вершин, что затрудняет интерактивную селекцию отдельных элементов.

1.2.3 Методы снижения размерности

Если вершины графа обладают многомерными векторными атрибутами, задача визуализации сводится к проекции из многомерного пространства $\mathbb{R}^d$ в пространство визуализации $\mathbb{R}^2$. Методы делятся на линейные (сохраняющие глобальные расстояния) и нелинейные (сохраняющие локальное соседство и топологию многообразия). Для больших графов критически важно использование алгоритмов, способных моделировать вероятностное распределение соседей, избегая проблемы «скученности».

1.2.4 Многоуровневые подходы

Для преодоления проблемы локальных минимумов в силовых алгоритмах и ускорения сходимости используется стратегия «огрубления». Исходный граф G_0 последовательно сжимается в цепочку графов $G_1, \ldots, G_k$ меньшего размера. Укладка рассчитывается для самого малого графа, а затем интерполируется обратно на исходный граф с локальной оптимизацией.

1.3 Агрегация и упрощение визуальной сложности

Даже при идеальной укладке отображение миллионов объектов делает визуализацию нечитаемой. Для решения этой проблемы используются методы абстракции данных.

1) Кластеризация и группировка — замена плотных групп вершин на мета-узлы (суперузлы). Это позволяет пользователю работать с графом на разных уровнях абстракции, раскрывая мета-вершины для их визуализации по мере повышения уровня детализации или, наоборот, сворачивание вершин и отображение мета-вершин взамен при уменьшении масштаба отображения графа или его видимой (рассматриваемой) части.

2) Связывание ребер — метод агрегации связей, при котором ребра группируются в пучки (по аналогии с кабельными трассами). Это снижает визуальный шум и позволяет выявлять высокоуровневые потоки информации между группами вершин.

3) Топологическая фильтрация — отображение только значимых элементов графа, например, остовного дерева или вершин с высокой метрикой центральности, с динамической подгрузкой деталей по запросу.

1.4 Проблематика интерактивного взаимодействия и селекции

Интерактивность является обязательным требованием для анализа больших графов. Ключевой функцией является возможность выбора вер-

шины курсором для получения детальной информации или выделения подграфа.

В условиях визуализации большого количества объектов ($>10^5$) задача выбора сопряжена со следующими проблемами.

— Проблема точности — при отображении всего графа размер отдельных вершин может быть меньше одного пикселя, либо вершины могут перекрывать друг друга. Стандартный клик мышью становится неэффективным.

— Вычислительная задержка — линейный перебор всех объектов для определения того, какой из них находится под курсором (hit-testing), недопустим при частоте обновления экрана 60 FPS. Требуется использование пространственных индексов (квадродеревьев, k-d деревьев) или GPU-ускорения.

— Контекстная обратная связь — при наведении курсора необходимо визуально выделять не только саму вершину, но и ее соседей или инцидентные ребра, что требует быстрого обхода графовых структур в памяти.

Таким образом, эффективная визуализация больших графов требует комплексного подхода, сочетающего быстрые алгоритмы укладки, методы визуальной агрегации и оптимизированные механизмы интерактивной селекции.

2 Обзор соответствующих программных решений

2.1 Методы пространственной укладки и агрегации графов

В данном разделе рассматриваются ключевые алгоритмы размещения вершин графа на плоскости.

2.1.1 Алгоритм Фрухтермана — Рейнгольда

Это классический силовой алгоритм, который моделирует граф как физическую систему. Вершины рассматриваются как атомные частицы или стальные кольца, которые отталкивают друг друга, а ребра — как пружины, притягивающие связанные вершины. Цель алгоритма — минимизировать энергию системы, обеспечивая равномерное распределение вершин и минимизацию пересечений ребер [5].

Основные этапы работы следующие.

1) Инициализация — вершины размещаются случайным образом в ограниченной области.

2) Расчет сил отталкивания — между каждой парой вершин действует сила $f_{rep}(d) = k^2/d$, где d — текущее расстояние, а k — оптимальное расстояние.

3) Расчет сил притяжения — только для связанных вершин рассчитывается сила $f_{attr}(d) = d^2/k$.

4) Смещение вершин — каждая вершина сдвигается в направлении вектора результирующей силы, но не дальше, чем позволяет текущая «температура» (параметр, ограничивающий максимальное смещение).

5) Охлаждение — температура постепенно снижается с каждой итерацией до полной остановки системы.

2.1.2 Алгоритм Камады — Каваи

Данный алгоритм относится к энергетическим алгоритмам. В его основе лежит идея, что геометрическое расстояние между вершинами на экране должно быть пропорционально их теоретико-графовому расстоянию (длине кратчайшего пути). Алгоритм ищет конфигурацию с минимальной общей энергией «напряжения» пружин [6].

Основные этапы работы следующие.

1) Вычисление матрицы кратчайших путей d_{ij} для всех пар вершин, обычно с помощью алгоритма Флойда — Уоршелла или BFS.

2) Определение идеальной длины пружины $l_{ij} = L \cdot d_{ij}$ и ее жесткости $k_{ij} = K/d_{ij}^2$.

3) Минимизация функции энергии $E = \sum_{i<j} k_{ij}(|p_i - p_j| - l_{ij})^2$, где p_i и p_j — координаты вершин.

4) Оптимизация — на каждом шаге выбирается вершина с наибольшей энергией смещения и перемещается в позицию локального минимума с помощью метода Ньютона — Рафсона.

Пример применения алгоритма Камады — Каваи представлен на рисунке 1.

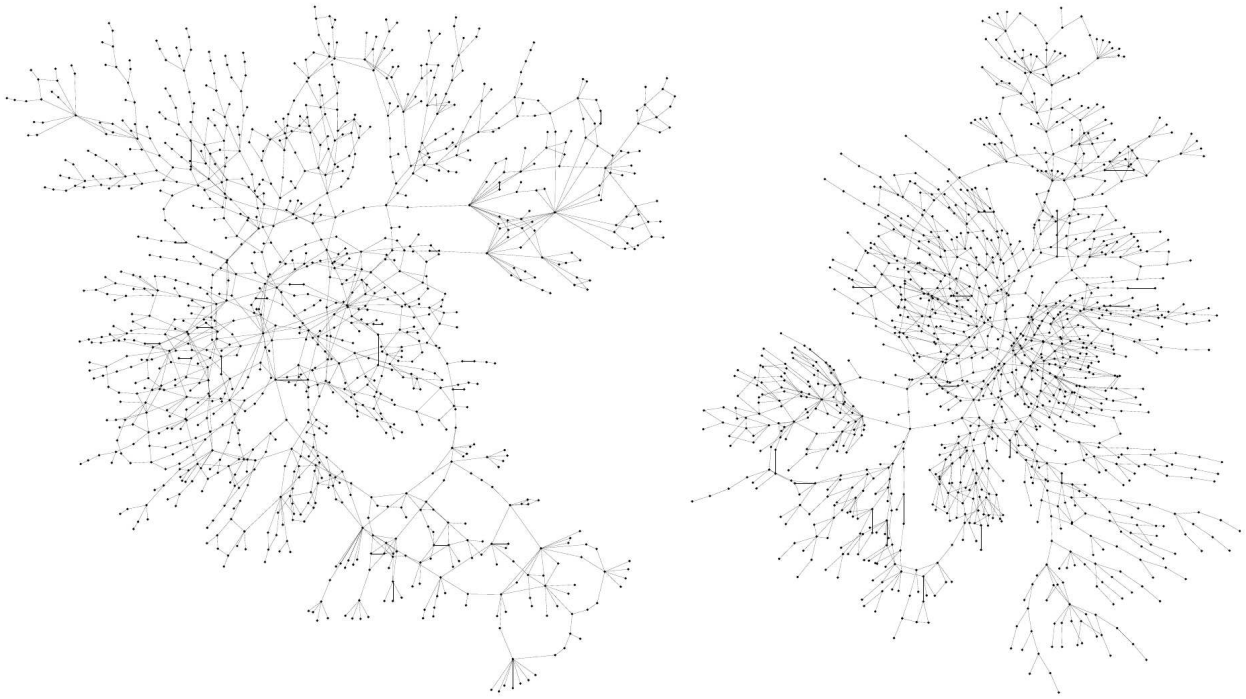

Рисунок 1 – Пример применения алгоритма Камады — Каваи (справа) и мажоризации напряжений (слева) к одному исходному графу [6]

2.1.3 Метод мажоризации напряжений

Этот метод является современной оптимизацией алгоритма Камады — Каваи. Вместо локального перемещения по одной вершине, он использует глобальную оптимизацию функции стресса. Мажоризация позволяет заменить сложную целевую функцию на более простую вспомогательную функцию (мажоранту), минимум которой найти легче [6].

Основные этапы работы следующие.

1) Вычисление функции стресса по формуле (1).

$$stress(X) = \sum_{i<j} w_{ij}(||X_i - X_j|| - d_{ij})^2. \quad (1)$$

2) Построение мажорирующей функции, которая является квадратичной формой и касается исходной функции в текущей точке.

3) Решение системы линейных уравнений вида $L^w X = L^Z Z$, где L^w — взвешенный Лапласиан (дифференциальный оператор, эквивалентый последовательному взятию операций градиента и дивергенции), для нахождения минимума мажоранты.

4) Итеративное повторение процесса до сходимости. Метод гарантирует монотонное уменьшение энергии и является более стабильным, чем метод Ньютона — Рафсона.

Пример применения алгоритма мажоризации напряжений представлен на рисунке 1.

2.1.4 Метод t-SNE

Метод нелинейного снижения размерности, который часто используется для визуализации графовых эмбеддингов или матриц смежности. Алгоритм t-SNE преобразует расстояния в многомерном пространстве в условные вероятности, стремясь сохранить локальную структуру данных [7].

Основные этапы работы следующие.

1) Вычисление попарных сходств в исходном пространстве высокой размерности с использованием распределения Гаусса.

2) Определение сходств в низкоразмерном (2D) пространстве с использованием t-распределения Стьюдента с одной степенью свободы (чтобы избежать проблемы «скученности»). Вероятность q_{ij} определяется по формуле (2).

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||y_k - y_l||^2)^{-1}}. \quad (2)$$

3) Минимизация дивергенции Кульбака — Лейблера между распределениями исходного и целевого пространств с помощью градиентного спуска.

2.1.5 Метод LargeVis

Метод, разработанный для визуализации очень больших графов и многомерных данных (масштаба миллионов вершин). Он устраняет недостатки t-SNE (высокую вычислительную сложность) путем аппроксимации графа ближайших соседей и использования вероятностной модели [7].

Основные этапы работы следующие.

1) Построение приближенного графа K-ближайших соседей (KNN-граф). Используются деревья случайной проекции (метод разбиения пространства случайными гиперплоскостями для быстрого поиска соседей в многомерных данных, основанный на k-d деревьях [7]) для быстрого разбиения пространства.

2) Уточнение графа с помощью алгоритма распространения соседей, основанного на принципе «сосед моего соседа — вероятно, мой сосед».

3) Визуализация графа с использованием вероятностной модели, которая максимизирует вероятность наблюдения связей. Оптимизация выполняется с помощью асинхронного стохастического градиентного спуска, что обеспечивает линейную сложность $O(N)$.

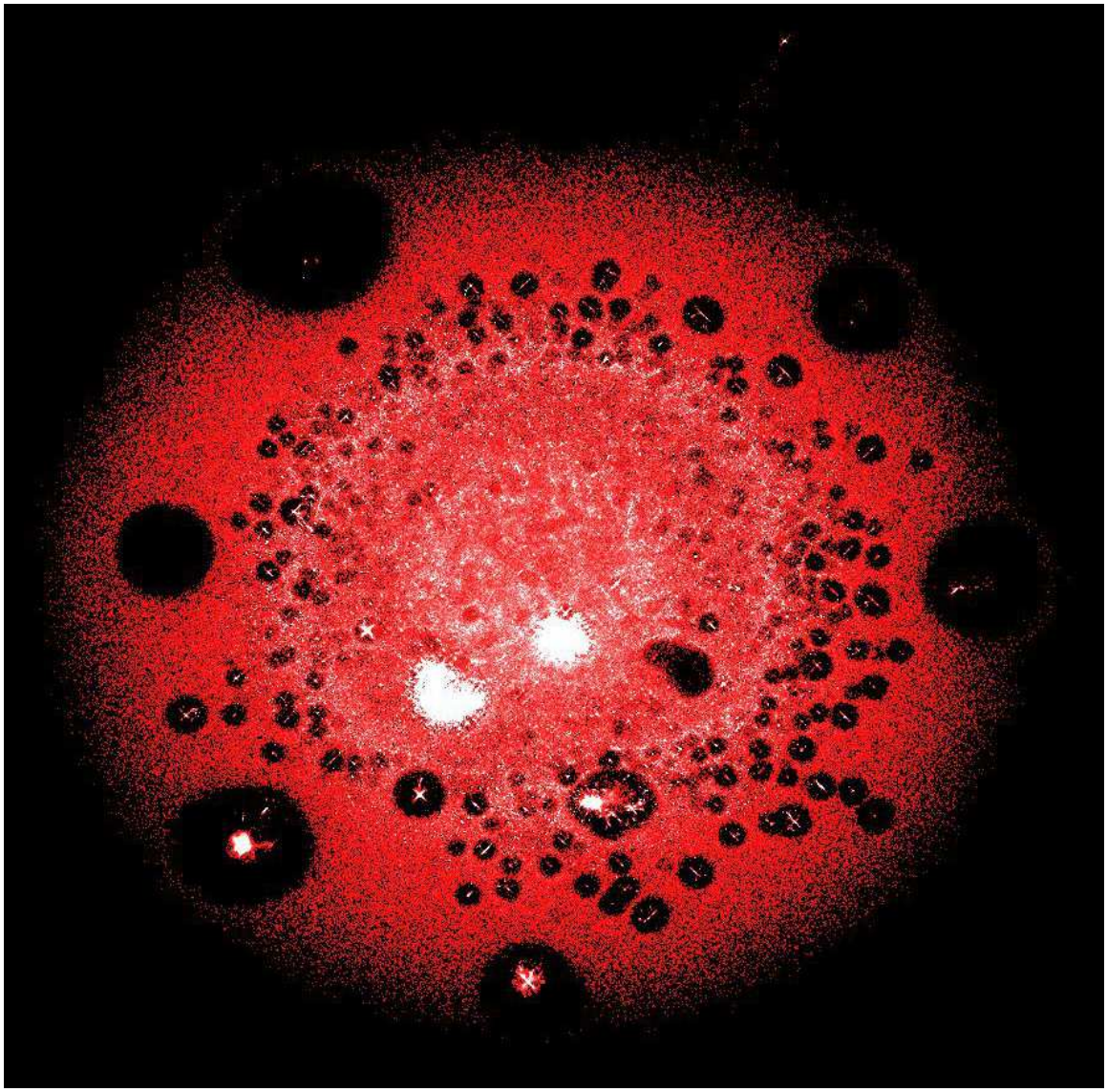

Рисунок 2 – Пример применения алгоритма LargeVis [8]

2.1.6 Метод Node2Vec

Метод обучения признаков, который преобразует вершины графа в векторы низкой размерности. Хотя сам по себе Node2Vec не дает координат на плоскости, полученные векторы затем визуализируются методами снижения размерности (например, t-SNE или PCA). Основная идея — использование смещенных случайных блужданий [9].

Основные этапы работы следующие.

1) Генерация случайных блужданий фиксированной длины из каждой вершины. Стратегия обхода регулируется параметрами p (вероятность возврата) и q (вероятность ухода вглубь), что позволяет балансировать между BFS и DFS.

2) Обучение модели Skip-gram (аналогично Word2Vec), где «предложениями» являются последовательности вершин в блужданиях. Максимизируется вероятность появления соседей в контексте данной вершины, которая вычисляется по формуле (3).

$$P = \max_{f} \sum_{u \in V} \log Pr(N_S(u)|f(u)). \tag{3}$$

3) Получение векторов признаков $f(u)$ и их последующая проекция в 2D.

Пример применения алгоритма Node2Vec представлен на рисунке 3.

Рисунок 3 – Пример применения алгоритма Node2Vec [8]

2.1.7 Метод VERSE

Метод VERSE (Versatile Graph Embeddings from Similarity Measures) представляет собой универсальный фреймворк для получения эмбеддингов (компактных векторных представлений вершин в низкоразмерном пространстве, сохраняющих структурную информацию исходного графа). В отличие от алгоритмов, использующих фиксированные стратегии случайных блужданий (как Node2Vec), данный метод не привязан к жесткой схеме сэмплирования, а обучается явно сохранять распределение любой выбранной пользователем меры сходства вершин. На вход алгоритм принимает структуру графа и выбранную функцию сходства, а результатом работы является матрица признаков, строки которой служат координатами вершин

в латентном пространстве для задач визуализации, классификации или предсказания связей [10].

Ключевой особенностью подхода является использование мер сходства — функций, которые определяют степень близости или родства между любой парой вершин в графе. Рассматриваются три основные меры: Personalized PageRank, которая учитывает глобальную структуру и пути в графе; SimRank, основанная на структурной эквивалентности; и простая смежность, учитывающая только непосредственных соседей. Выбор меры позволяет адаптировать метод под конкретную прикладную задачу.

Основные этапы работы следующие.

1) Выбор меры сходства sim_G, которая для каждой вершины задает эталонное распределение вероятностей перехода к другим вершинам графа.

2) Обучение однослойной нейронной сети, задача которой — сформировать векторы вершин таким образом, чтобы их скалярное произведение в пространстве эмбеддингов аппроксимировало выбранную меру сходства.

3) Оптимизация теоретической целевой функции, представляющей собой дивергенцию Кульбака — Лейблера. Эта функция минимизирует различие между распределением сходства в исходном графе (sim_G) и восстановленным распределением в пространстве эмбеддингов. Иными словами, алгоритм «штрафует» модель, если «расстояния» между векторами не соответствуют «расстояниям» в графе.

4) Применение техники Noise Contrastive Estimation для масштабируемого обучения. Поскольку явное вычисление полной матрицы сходства и минимизация KL-дивергенции вычислительно слишком дороги, задача сводится к обучению логистической регрессии. Модель учится отличать примеры вершин, выбранные из реального распределения сходства, от случайных «шумовых» вершин. Параметры обновляются с помощью стохастического градиентного спуска, что позволяет обрабатывать графы с миллионами узлов.

Пример применения алгоритма VERSE представлен на рисунке 4.

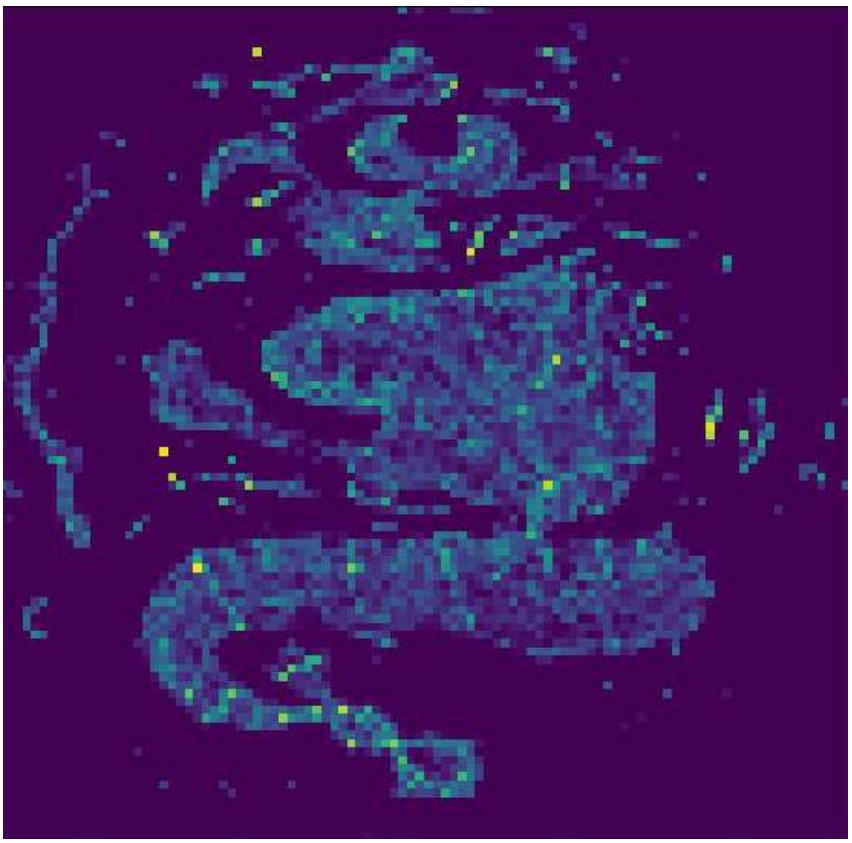

Рисунок 4 – Пример применения алгоритма VERSE [8]

2.1.8 Упрощенная графовая свертка

Упрощенная модель графовых сверточных сетей, которая устраняет нелинейности между слоями, сводя глубокую сеть к линейной модели. Это значительно ускоряет обучение и делает модель интерпретируемой, действуя как низкочастотный фильтр в спектральной области [11].

Основные этапы работы следующие.

1) Добавление петель к графу (ренормализация): $\tilde{A} = A + I$.

2) Расчет нормализованной матрицы смежности $\tilde{S} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$.

3) Распространение признаков (feature propagation) путем K-кратного умножения матрицы признаков X на $\tilde{S}$ по формуле (4).

$$\bar{X} = \tilde{S}^K X. \tag{4}$$

4) Использование полученных сглаженных признаков $\bar{X}$ для классификации или визуализации (например, через PCA).

2.1.9 Иерархическое связывание ребер

Метод, предназначенный для решения проблемы визуального шума в плотных графах путем группировки смежных ребер. Он использует иерархическую структуру данных (дерево) для направления ребер [12].

Основные этапы работы следующие.

1) Построение иерархии (если она не задана, может использоваться алгоритм кластеризации).

2) Определение пути для каждого ребра графа: ребро между вершинами P и Q строится вдоль пути в иерархии от P до Q через их наименьшего общего предка.

3) Моделирование ребра как B-сплайна, где контрольные точки определяются узлами иерархии.

4) Регулировка степени натяжения параметром β. При $\beta = 0$ ребра прямые, при $\beta = 1$ они сильно прижаты к иерархическому дереву, образуя пучки.

2.1.10 Топологический рыбий глаз

Метод навигации и визуализации очень больших графов, основанный на гибридном представлении. В отличие от геометрического искажения, здесь используется топологическое упрощение удаленных областей [13].

Основные этапы работы следующие.

1) Предварительное вычисление иерархии огрубленных графов путем последовательного стягивания вершин.

2) Выбор пользователем фокусной вершины.

3) Построение «гибридного графа» на лету: окрестность фокуса берется из исходного детального графа, а удаленные регионы заменяются соответствующими частями из огрубленных уровней иерархии.

4) Применение геометрического искажения к полученному гибридному графу для обеспечения плавного перехода между уровнями детализации.

Пример применения метода топологический рыбий глаз представлен на рисунке 5.

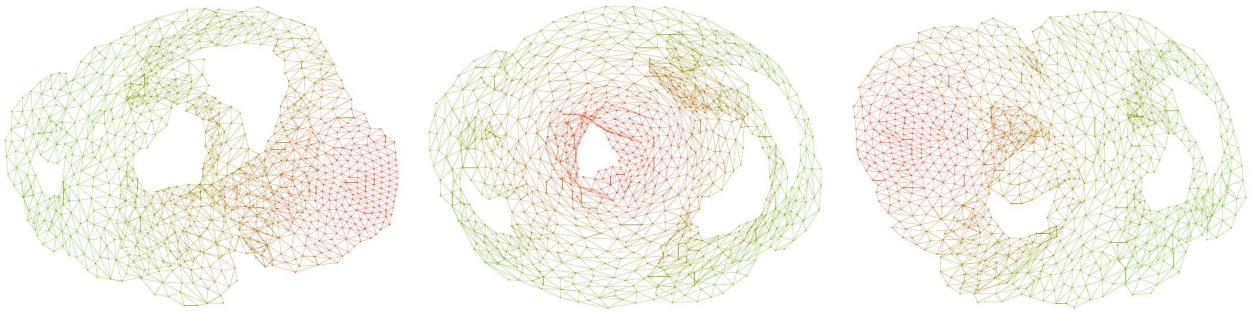

Рисунок 5 – Пример применения метода топологический рыбий глаз с центром в различных местах графа [13]

2.1.11 Алгоритм FM^3

Данный алгоритм представляет собой многоуровневый подход к силовой укладке, специально разработанный для визуализации больших графов. Он решает проблему медленной сходимости и попадания в локальные минимумы, характерную для классических силовых методов при большом количестве вершин [2].

Основные этапы работы следующие.

1) Огрубление — исходный граф последовательно сжимается путем объединения смежных вершин в супер-узлы, создавая иерархию графов уменьшающегося размера.

2) Укладка на нижнем уровне — для самого маленького графа в иерархии рассчитывается начальная укладка с использованием силового алгоритма.

3) Интерполяция и уточнение — координаты вершин переносятся с грубого уровня на более детальный. На каждом уровне применяется локальная силовая оптимизация (релаксация) для уточнения позиций. Для ускорения расчета сил отталкивания используется аппроксимация с помощью мультипольного метода (аналогично алгоритму Барнса — Хата).

2.1.12 Алгоритм OpenOrd

Алгоритм укладки, основанный на силовом подходе, но модифицированный для выявления кластерной структуры в очень больших графах (до миллионов узлов). Ключевой особенностью является агрессивное отсечение длинных ребер в процессе оптимизации [8].

Основные этапы работы следующие.

1) Жидкая фаза: допускаются большие перемещения вершин, высокая температура симуляции.

2) Фаза расширения: граф растягивается для улучшения разделимости.

3) Фаза охлаждения и отсечения: температура снижается, и длинные ребра, соединяющие удаленные кластеры, игнорируются при расчете сил притяжения. Это позволяет кластерам обособиться друг от друга.

4) Завершающая фаза: локальная доводка позиций для улучшения эстетического вида внутри кластеров.

2.1.13 Метод LinLog

Энергетическая модель укладки, предложенная Ноаком как альтернатива модели Фрухтермана — Рейнгольда для лучшего выявления кластеров. Название происходит от типа используемых сил: сила притяжения логарифмическая, а сила отталкивания — линейная (в некоторых интерпретациях, наоборот, зависит от контекста формулировки энергии или силы) [14].

Основные этапы работы следующие.

1) Определение целевой функции энергии, где штраф за длину ребра растет линейно (что сильно стягивает удаленные связанные узлы), а отталкивание препятствует коллапсу. В терминах энергии минимизируется выражение (5).

$$E = \sum_{(u,v)\in E} ||p_u - p_v|| - \sum_{(u,v)\in V\times V} \ln ||p_u - p_v||. \tag{5}$$

2) Оптимизация этой функции приводит к тому, что узлы с высокой плотностью связей (кластеры) группируются очень тесно, а расстояние

между кластерами становится значительным, что делает структуру сообществ визуально очевидной.

2.1.14 Размещение «Линия темы»

Специализированный алгоритм укладки, используемый для анализа социальных сетей или событийных графов, где важно выделить ключевые объекты («темы») и их связи с окружением. Метод эффективен для визуализации хронологии или иерархии влияния [15].

Основные этапы работы следующие.

1) Выбор ключевых вершин (тем), которые будут размещены на базовой горизонтальной линии.

2) Размещение линии темы: ключевые вершины выстраиваются в линию, длина которой может варьироваться.

3) Размещение окружения: остальные вершины, связанные с темой, размещаются перпендикулярно линии (сверху или снизу).

4) Маршрутизация ребер: связи между второстепенными вершинами и линией темы рисуются вертикально, минимизируя пересечения с другими элементами.

Пример размещения «Линия темы» представлен на рисунке 6.

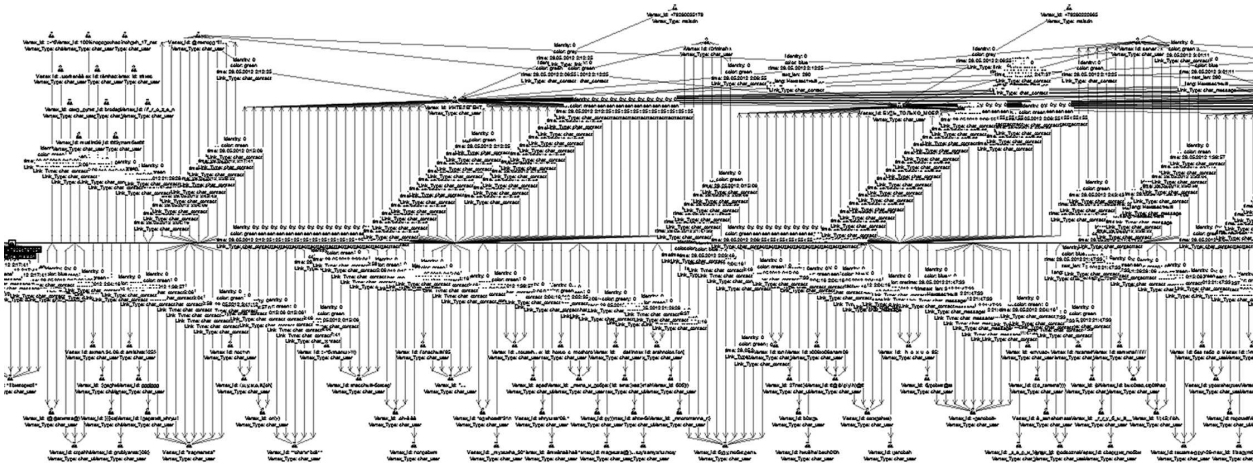

Рисунок 6 – Пример размещения «Линия темы»

2.1.15 Матричное упорядочение SlashBurn

Метод, предназначенный для визуализации и сжатия матриц смежности больших разреженных графов реального мира (социальные сети, веб-графы). Он использует свойство степенного распределения степеней вершин для оптимизации порядка строк и столбцов матрицы [14].

Основные этапы работы следующие.

1) Идентификация хабов: поиск вершин с наивысшей степенью.

2) Удаление хабов: вершины-хабы и инцидентные им ребра временно удаляются из графа.

3) Выделение компонент связности: оставшийся граф распадается на множество мелких компонент.

4) Переупорядочивание: хабы помещаются в начало матрицы, а вершины компонент связности группируются вместе. Это создает характерную форму «стрелы» в матрице смежности, концентрируя ненулевые элементы и разрежая остальное пространство.

2.1.16 Алгоритм ForceAtlas2

Специализированный силовой алгоритм, разработанный для визуализации сетей «малого мира» и безмасштабных графов. Является алгоритмом по умолчанию в популярном инструменте Gephi. Он отличается линейно-логарифмической моделью взаимодействия (сила притяжения пропорциональна расстоянию, отталкивания — обратна расстоянию) и адаптивной скоростью сходимости, что позволяет пользователю интерактивно наблюдать за процессом укладки [16].

Основные этапы работы следующие.

1) Расчет сил: отталкивание между всеми узлами (оптимизировано через Барнса — Хата, $O(N \log N)$) и притяжение только между соседями (линейно от расстояния).

2) Учет весов ребер и степени вершин: узлы с высокой степенью отталкиваются сильнее, что выталкивает хабы на периферию кластеров.

3) Адаптивное охлаждение: шаг смещения зависит от локальной «температуры». Если узел колеблется, шаг уменьшается; если движется в одном направлении — увеличивается.

4) Предотвращение перекрытий: дополнительная стадия для разведения узлов на расстояние их радиуса.

Пример применения алгоритма ForceAtlas2 представлен на рисунке 7.

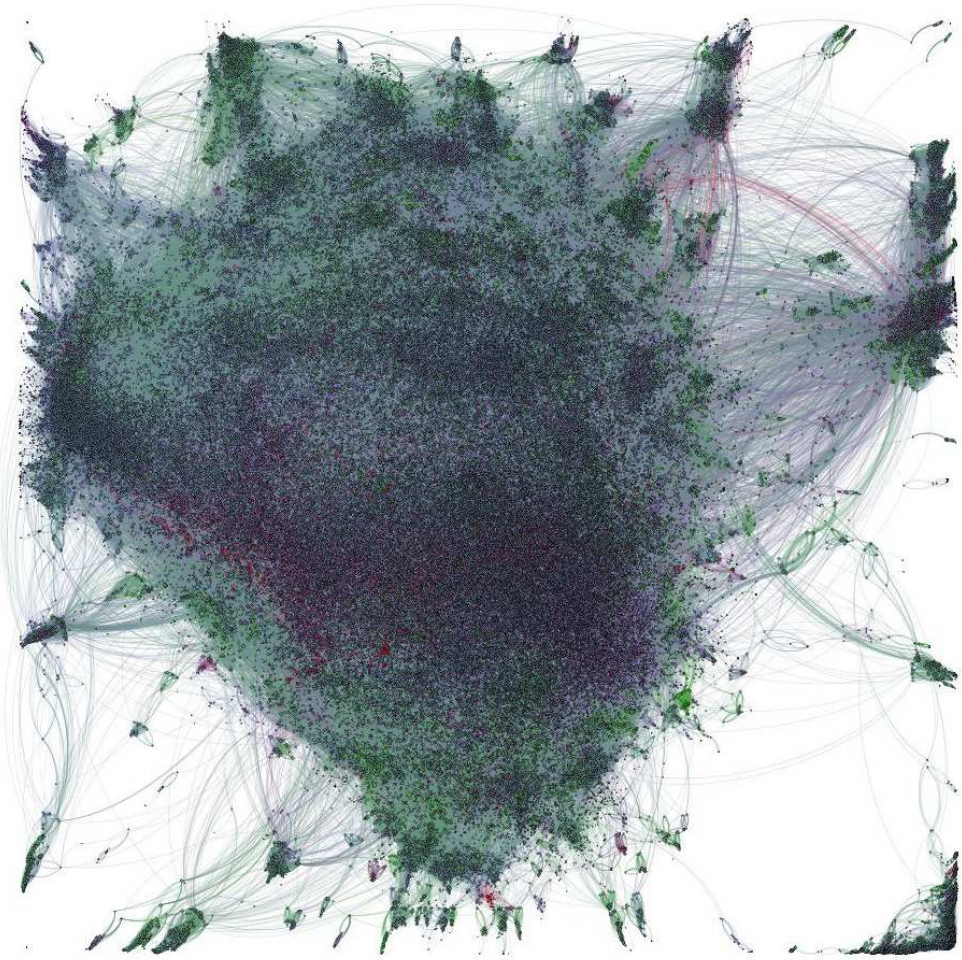

Рисунок 7 – Пример применения алгоритма ForceAtlas2 [8]

2.1.17 Алгоритм sfdp

Многоуровневый алгоритм силовой укладки, входящий в пакет Graphviz. Предназначен для визуализации очень больших графов (миллионы узлов). Основная идея — использование огрубления графа для быстрого нахождения глобальной структуры и последующая детализация. Алгоритм эффективно использует память и процессорное время [17].

Основные этапы работы следующие.

1) Построение иерархии: исходный граф сворачивается в серию уменьшающихся графов.

2) Начальная укладка: самый грубый граф укладывается силовым методом.

3) Уточнение: координаты переносятся на более детальный уровень. Применяется алгоритм укладки с использованием аппроксимации дальнодействующих сил для уточнения позиций.

4) Адаптация региона: алгоритм может адаптировать плотность укладки в разных регионах графа, чтобы избежать наложений в плотных кластерах.

2.1.18 Метод HDE

Метод быстрого размещения графа, основанный на проецировании из пространства высокой размерности. Он отличается высокой скоростью работы, что делает его пригодным для интерактивного анализа графов с сотнями тысяч вершин [5].

Основные этапы работы следующие.

1) Выбор опорных вершин: выбирается небольшое множество вершин (обычно 50–100).

2) Встраивание в многомерное пространство: для каждой вершины графа вычисляется вектор расстояний (кратчайших путей) до всех опорных вершин. Если выбрано K опорных точек, каждая вершина получает координаты в K-мерном пространстве.

3) Снижение размерности: применяется метод главных компонент (PCA) к полученной матрице координат размера $N \times K$ для проецирования данных на 2D плоскость таким образом, чтобы сохранить максимальную дисперсию.

2.1.19 Генетические алгоритмы укладки

Подход к размещению графа, основанный на принципах эволюционной биологии. Он позволяет находить глобально оптимальные укладки, учиты-

вая сложные и конфликтующие эстетические критерии, которые трудно формализовать в виде гладкой функции энергии [18].

Основные этапы работы следующие.

1) Создание начальной популяции: генерируется множество случайных вариантов раскладки графа.

2) Оценка приспособленности: для каждого варианта вычисляется оценка качества, учитывающая количество пересечений ребер, равномерность распределения вершин, длину ребер и другие критерии.

3) Селекция и скрещивание: отбираются лучшие варианты, и их координаты комбинируются для создания новых решений.

4) Мутация: вносятся случайные изменения в координаты вершин для предотвращения преждевременной сходимости. Процесс повторяется до достижения критерия остановки.

2.1.20 Гиперболическое дерево

Метод визуализации иерархических графов (деревьев), использующий неевклидову геометрию. Он реализует концепцию «фокус + контекст», позволяя отображать большие структуры данных в ограниченном пространстве [19].

Основные этапы работы следующие.

1) Размещение в гиперболической плоскости: узлы дерева укладываются на гиперболической плоскости (обычно используется модель диска Пуанкаре). Корневой узел помещается в центр.

2) Выделение пространства: каждому узлу выделяется сектор, размер которого экспоненциально уменьшается с удалением от центра. Это свойство гиперболической геометрии позволяет вместить экспоненциально растущее число узлов.

3) Проекция на экран: гиперболические координаты преобразуются в экранные евклидовы координаты. Узлы в центре (фокусе) отображаются крупно, а удаленные узлы сжимаются у границ диска.

2.2 Методы интерактивного взаимодействия и выбора вершин

Для эффективной работы с большими графами критически важны методы, позволяющие пользователю точно выбирать элементы (вершины и ребра) даже в условиях высокой плотности и перекрытия объектов.

2.2.1 Семантическое масштабирование

Метод взаимодействия, при котором изменение масштаба (зумирование) влияет не только на геометрический размер объектов, но и на способ их представления. При отдалении вершины могут отображаться как точки или агрегированные кластеры, а при приближении курсора к интересующей области детализация повышается: появляются метки, иконки и внутренние связи. Это позволяет пользователю выбирать вершину только тогда, когда она достаточно велика и различима [20].

2.2.2 Искажение «Рыбий глаз»

Техника искажения экранного пространства, основанная на метафоре широкоугольной линзы. Область под курсором мыши увеличивается, раздвигая соседние элементы, в то время как периферия сжимается. Это позволяет пользователю точно выбрать мелкую вершину в плотном кластере, не теряя при этом глобального контекста графа и связи с окружающими узлами [21].

2.2.3 Эксцентричные метки

Специализированный метод выбора и идентификации вершин в очень плотных областях, где стандартные текстовые подписи перекрывают друг друга. При наведении курсора на скопление узлов система определяет окрестность фокуса и выводит список меток для всех попавших в нее вершин в виде упорядоченного списка рядом с курсором. Линии-выноски соединяют пункты списка с соответствующими узлами, позволяя точно выбрать нужный объект [2].

2.2.4 Инкрементальный просмотр

Подход к навигации, при котором изначально пользователю показывается лишь небольшая часть графа (например, корневой узел или выборка важных вершин). Остальная часть графа скрыта. При выборе вершины курсором система динамически подгружает и отображает ее соседей. Это позволяет работать с графами бесконечного размера, избегая перегрузки экрана и упрощая выбор следующего узла для перехода [19].

2.2.5 Связывание и кисть

Метод группового выбора, используемый при наличии нескольких представлений данных (например, граф и матрица смежности). Пользователь выделяет группу вершин курсором («кистью») в одном окне (например, обводя область на графе), и соответствующие этим вершинам элементы автоматически подсвечиваются во всех остальных представлениях. Это облегчает идентификацию и выбор вершин, которые могут быть скрыты или труднодоступны в одном из визуальных представлений [20].

2.2.6 Волшебные линзы

Интерактивный инструмент в виде подвижной области (линзы), перемещаемой курсором поверх графа. Внутри линзы меняется способ отображения данных: например, могут скрываться ребра для облегчения выбора вершин, изменяться цветовая схема или показываться скрытые атрибуты. Это позволяет производить селекцию в локальном контексте без изменения глобальных настроек визуализации [2].

2.2.7 Лассо-выделение

Инструмент для выбора множества вершин произвольной формы. Пользователь «рисует» курсором замкнутый контур вокруг интересующей группы узлов. Все вершины, попавшие внутрь контура, становятся выбранными. Этот метод значительно эффективнее прямоугольного выделения для графов со сложной геометрической структурой или при необходимости выделить конкретный кластер неправильной формы [2].

2.2.8 Приближение и переход

Техника навигации для выбора соседей вершины. При выборе узла его непосредственные соседи, которые могут находиться далеко на экране, плавно перемещаются («подтягиваются») ближе к выбранному узлу. Это позволяет пользователю легко выбрать следующую вершину для перехода без необходимости прокручивать экран и искать связи в большом количестве ребер [2].

2.2.9 Метафора «Резиновый жгут»

Метод геометрической трансформации, позволяющий пользователю растягивать область просмотра, словно резиновый лист. Захватив курсором точку на холсте и потянув ее, пользователь растягивает прилегающую область графа, увеличивая расстояние между вершинами. Это позволяет временно «разрядить» плотный участок графа для удобного выбора конкретной вершины без потери топологии [21].

2.2.10 Динамические запросы

Метод непрямого выбора вершин через элементы интерфейса (слайдеры, ползунки, чекбоксы). Пользователь фильтрует вершины по атрибутам (например, «показать вершины со степенью > 100»). Отфильтрованные вершины либо скрываются, либо затеняются, оставляя на экране только те объекты, которые соответствуют критериям. Это позволяет легко выбирать целевые вершины курсором, убрав визуальный шум [22].

3 Сравнительный анализ существующих решений

3.1 Сравнительный анализ методов укладки и агрегации графов

В таблице 1 приведен сравнительный анализ 20 рассмотренных методов по критериям:

— быстродействие — асимптотическая временная сложность алгоритма;

— тип графа;

— устойчивость — дает ли алгоритм схожие результаты при повторном запуске или при незначительном изменении данных;

— выделение кластеров — способен ли метод визуально отделить плотные группы вершин друг от друга.

Введены следующие числовые обозначения:

— устойчивость: 0 — результат меняется при перезапуске, 1 — результат всегда одинаков, 2 — результат устойчив к малым изменениям, то есть при добавлении вершины граф не перестраивается целиком;

— выделение кластеров: 0 — отсутствует, 1 — неявное (кластеры видны, но не разделены), 2 — явное (четкое визуальное разделение).

При описании быстродействия используются следующие обозначения: N — число вершин, E — число ребер, K — высота дерева (для графов, являющихся деревом).

Таблица 1 – Сравнение методов укладки и агрегации

Метод	Быстро-действие	Тип графа	Устойчи-вость	Выделение кластеров
Фрухтермана-Рейнгольда	$O(N^2)$	Любой	0	1
Камады-Каваи	$O(N^3)$	Любой	1	1
Мажоризация напряжений	$O(N^2)$	Любой	1	1
t-SNE	$O(N \log N)$	Любой	0	2
LargeVis	$O(N)$	Любой	0	2
Node2Vec	$O(N)$	Любой	0	1
VERSE	$O(N)$	Любой	0	1
Упрощенная графовая свертка	$O(E)$	Любой	1	1
Иерархическое связывание ребер	$O(E \cdot K)$	Дерево	1	2
Топологический рыбий глаз	$O(N \log N)$	Любой	1	1
FM^3	$O(N \log N)$	Любой	2	1
OpenOrd	$O(N \log N)$	Разреженный	0	2
LinLog	$O(N \log N)$	Любой	0	2
Размещение «Линия темы»	$O(N)$	Событийный	1	0
SlashBurn	$O(N + E)$	Разреженный	1	2
ForceAtlas2	$O(N \log N)$	Любой	0	2
sfdp	$O(N \log N)$	Любой	0	2
HDE	$O(N)$	Любой	1	1
Генетические алгоритмы	$O(N^3)$	Любой	0	1
Гиперболическое дерево	$O(N)$	Дерево	1	1

3.2 Сравнительный анализ методов навигации и интерактивности

В таблице 2 приведен сравнительный анализ 10 рассмотренных методов по критериям:

— точность выбора вершины;

— сохранение контекста — изменяется ли видимость графа;

— тип — основное назначение метода.

Введены следующие числовые обозначения:

— точность выбора вершины: 0 — требует точного попадания в вершину, 1 — требует увеличения для выбора конкретной вершины, 2 — упрощает выбор перекрытых объектов;

— сохранение контекста: 0 — видна только часть графа, 1 — граф виден полностью, но искажен, 2 — граф виден полностью без искажений.

Таблица 2 – Сравнение методов интерактивности и навигации

Метод	Точность выбора вершины	Сохранение контекста	Тип
Семантическое масштабирование	1	0	Навигационный
Искажение «Рыбий глаз»	2	1	Навигационный
Эксцентричные метки	2	2	Атомарный
Инкрементальный просмотр	1	0	Навигационный
Связывание и кисть	1	2	Групповой
Волшебные линзы	2	1	Фильтрационный
Лассо-выделение	1	2	Групповой
Приближение и переход	2	1	Навигационный
Метафора «Резиновый жгут»	2	1	Навигационный
Динамические запросы	2	0	Фильтрационный

ЗАКЛЮЧЕНИЕ

В данной работе проведено исследование методов 20 визуализации от классических силовых алгоритмов до современных подходов на основе машинного обучения. Рассмотрены методы борьбы с информационной перегрузкой, такие как кластеризация и связывание ребер. Исследованы 10 методов навигации и выбора вершин курсором в условиях высокой плотности объектов.

Цель работы достигнута — были проанализированы существующие методы интерактивной визуализации больших графов. Были выполнены следующие задачи:

— проведён анализ предметной области, введены базовые определения области визуализации графов и обозначены ключевые проблемы отображения больших массивов данных;

— проанализированы 30 существующих программных решений алгоритмов укладки и способов интерактивного выбора вершин курсором;

— сформулированы условия и критерии сравнения найденных программных решений;

— построена сравнительная таблица для найденных решений на основе сформулированных условий и критериев.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Касьянов В. Н., Золотухин Т. А. Visual Graph — система для визуализации сложно структурированной информации большого объема на основе графовых моделей // Системная информатика. – 2015. – № 4. – С. 1–16.

2. von Landesberger T. Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges / T. von Landesberger, A. Kuijper, T. Schreck [и др.] // Computer Graphics Forum. – 2011. – Т. 30, № 6. – С. 1719–1749.

3. Наймушин А. В. – Эвристические методы укладки графов на плоскость: Магистерская диссертация: Тверской государственный университет. – Тверь, 2018. – Научный руководитель: к.ф.-м.н. Карлов Б. Н.

4. Kwon O. H., Crnovrsanin T., Ma K. L. What Would a Graph Look Like in This Layout? A Machine Learning Approach to Large Graph Visualization // IEEE Transactions on Visualization and Computer Graphics. – 2018. – Т. 24, № 1. – С. 478–488.

5. Gibson H., Faith J., Vickers P. A survey of two-dimensional graph layout techniques for information visualisation // Information Visualization. – 2013. – Т. 12, № 3-4. – С. 324–357.

6. Gansner E. R., Koren Y., North S. Graph Drawing by Stress Majorization // Graph Drawing. Lecture Notes in Computer Science. – Т. 3383. – Berlin, Heidelberg: Springer, 2005. – С. 239–250.

7. Tang J. Visualizing Large-scale and High-dimensional Data / J. Tang, J. Liu, M. Zhang [и др.] // Proceedings of the 25th International Conference on World Wide Web / International World Wide Web Conferences Steering Committee. – 2016. – С. 287–297.

8. Ковалев С. Визуализация больших графов для самых маленьких [Электронный ресурс]. – Open Data Science / Habr. – 2019. – URL: https://habr.com/ru/companies/ods/articles/464715/ (дата обращения: 20.11.2025).

9. Grover A., Leskovec J. node2vec: Scalable Feature Learning for Networks // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining / ACM. – 2016. – С. 855–864.

10. Tsitsulin A. VERSE: Versatile Graph Embeddings from Similarity Measures / A. Tsitsulin, D. Mottin, P. Karras [и др.] // Proceedings of the 2018 World Wide Web Conference / International World Wide Web Conferences Steering Committee. – 2018. – С. 539–548.

11. Wu F. Simplifying Graph Convolutional Networks / F. Wu, A. Souza, T. Zhang [и др.] // International Conference on Machine Learning / PMLR. – 2019. – С. 6861–6871.

12. Holten D. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data // IEEE Transactions on Visualization and Computer Graphics. – 2006. – Т. 12, № 5. – С. 741–748.

13. Gansner E. R., Koren Y., North S. C. Topological Fisheye Views for Visualizing Large Graphs // IEEE Transactions on Visualization and Computer Graphics. – 2005. – Т. 11, № 4. – С. 457–468.

14. Апанович З. В. Использование матриц смежности для визуализации больших графов // Электронные библиотеки. – 2019. – Т. 22, № 1. – С. 3–38.

15. Коломейченко М. И., Чеповский А. М. Визуализация и анализ графов больших размеров // Бизнес-информатика. – 2014. – № 4 (30). – С. 7–15.

16. Jacomy M. ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software / M. Jacomy, T. Venturini, S. Heymann [и др.] // PLoS ONE. – 2014. – Т. 9, № 6. – С. 1–12.

17. Hu Y. Efficient and High Quality Force-Directed Graph Drawing // The Mathematica Journal. – 2005. – Т. 10, № 1. – С. 37–71.

18. Костина М. А., Мельникова Е. А. Алгоритмы искусственного интеллекта в задаче визуализации графа // Вектор науки Тольяттинского государственного университета. – 2014. – № 3 (29). – С. 32–35.

19. Касьянов В. Н., Касьянова Е. В. Визуализация информации на основе графовых моделей // Научный сервис в сети Интернет: поиск новых решений. – 2013. – С. 148–156.

20. Yi J. S. Toward a Deeper Understanding of the Role of Interaction in Information Visualization / J. S. Yi, Y. Kang, J. T. Stasko [и др.] // IEEE Transactions on Visualization and Computer Graphics. – 2007. – Т. 13, № 6. – С. 1224–1231.

21. Апанович З. В. Методы навигации при визуализации графов // Вестник НГУ. Информационные технологии. – 2008. – Т. 6, № 3. – С. 36–47.

22. Shneiderman B. The eyes have it: A task by data type taxonomy for information visualizations // Proceedings 1996 IEEE Symposium on Visual Languages / IEEE. – 1996. – С. 336–343.