МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Формирование карты рейтинга жилых районов

Курсовая работа

09.03.04 Программная инженерия

Допущен к защите

Зав. кафедрой _____ С.Д. Махортов, д.ф.- м.н., доцент __.__.2021

Обучающийся _____ А.В. Калиткин, 3 курс, д/о

Руководитель _____ Н.К. Самойлов, ст. преподаватель

Воронеж 2021

Содержание

Содержание 2

Введение 3

1 Постановка задачи 4

2 Анализ предметной области 5

2.1 Обзор существующих решений 5

2.2 Выбор источника данных об инфраструктуре 7

3 Реализация 8

3.1 Выбор средства доступа к данным 8

3.2 Отрисовка тайлов 9

3.3 Выбор технологий и архитектуры для приложения 10

3.4 Структура данных OSM 11

3.5 Использование Overpass API 11

3.6 Выбор данных об инфраструктуре 11

3.7 Оптимизация запросов 13

3.8 Расчёт рейтинга 16

3.9 Построение тепловой карты рейтинга 17

3.10 Оптимизация алгоритмов 18

Заключение 21

Список использованных источников 22

Введение

Выбор подходящего места для покупки или аренды жилья – одна из задач, с которой сталкиваются люди при планировании или уже в процессе переезда на новое место жительства. Как правило, на выбор всегда оказывает влияние большое количество факторов. К ним можно отнести ограниченность предложения на рынке, отсутствие вариантов с удобным расположением и приемлемой ценой, наличие поблизости социальной инфраструктуры и остановок общественного транспорта, уровень преступности в данном районе, экологическую обстановку и многие другие.

В случае, если возможных вариантов всего несколько, каждый из них можно досконально изучить самостоятельно. В том случае, если выбор достаточно велик или будущее место жительства не привязано к конкретному району города, на самостоятельное изучение обстановки придётся потратить во много раз больше времени.

Один из способов упросить анализ – использовать некую метрику для оценки всех возможных вариантов. Эта метрика может быть как субъективной и формироваться на основе данных социологических опросов, так и объективной и основываться на данных из открытых источников. Но в любом случае, единого варианта существовать не может, так как каждый человек ведёт свой образ жизни и имеет разные потребности, по-разному оценивает важность тех или иных критериев.

Критерий, который очень важен проживающим в городе семьям – наличие поблизости инфраструктуры и возможность удобно добираться из дома до неё и до места учёбы или работы всех членов семьи. Таким образом, метрика, вычисляемая на основе этого критерия будет подходить большинству, кто столкнётся с необходимостью оценить качество того или иного места в городе с точки зрения комфортного проживания.

Для удобства восприятия, полученную метрику логично будет представлять в виде тепловой карты, отображаемой поверх карты города.

1 Постановка задачи

Цель данной работы – разработать приложение, позволяющее пользователям строить тепловую карту качества района на основе данных об инфраструктуре из открытых источников.

Конечные пользователи – люди, изучающие другой город с целью выбора наилучшего возможного будущего места жительства в связи с планируемым переездом.

Разрабатываемое приложение должно соответствовать следующим требованиям:

- Не должно требовать установки
- Использовать только открытые данные
- Должно быть интерактивным
- Не должно требовать от пользователя специальных знаний

Для выполнения поставленной задачи необходимо:

- Провести анализ предметной области
- Определить преимущества и недостатки существующих решений
- Выбрать источник открытых данных об инфраструктуре
- Выбрать средства реализации и архитектуру приложения
- Разработать приложение

2 Анализ предметной области

2.1 Обзор существующих решений

В ходе анализа предметной области было выяснено, что в большинстве случаев рейтинги районов городов составляются по результатам опросов общественного мнения, в которых жителей просят оценить их текущее место проживания в городе по нескольким критериям. Примерно половину из этих критериев составляют вопросы об обстановке в районе в целом с упором на такие аспекты, как чистота, экология, безопасность и работа коммунальных служб; остальные – о различных видах инфраструктуры: магазинах, развлечениях, инфраструктуры для детей, общественном транспорте и т. д.

Рисунок 1 – Рейтинг районов Воронежа по результатам опросов жителей

Такие опросы позволяют оценить в целом район города, получить некоторую цифру, по которой этот район можно сравнить с другими, а также выявлять тенденции к изменению тех или иных показателей, но они имеют недостаточно высокую точность. Для того, чтобы привязка полученного рейтинга к географическому положению жителей была максимально точной, а погрешность из-за субъективности в ответах на опросы была минимальной,

необходимо проводить опрос, имея довольно большую выборку, что определяет большую сложность данной задачи.

Из общего ряда выделяются исследования Яндекса, посвящённые Москве и Санкт-Петербургу с точки зрения доступности инфраструктуры для жизни. Эти исследования были проведены аналитически, используя данные Яндекс.Карт об организациях. В ходе исследования наиболее заселённые районы городов были разделены на квадраты со стороной 300 метров, и затем каждый из квадратов относили к одному из трёх классов – «Недостаточно инфраструктуры», «Удобно для жизни» или «Очень удобно для жизни» на основании доступности для жителей этого квадрата всей необходимой для жизни инфраструктуры в пешей доступности.

Рисунок 2 – Проведённое Яндексом исследование Москвы с точки зрения доступности инфраструктуры для жизни

Аналогичные карты были составлены для инфраструктуры для развлечений, или учитывая все виды инфраструктуры вместе. Затем полученные по квадратам данные были сгруппированы по районам, к которым они относятся, и по полученным результатам были сформированы интерактивная карта и таблица с рейтингами и особенностями районов – относительной долей квадратов каждого из классов, среднее время пешком до заведений и организаций разных видов и комментарии исследователей.

Стоит так же упомянуть сервис Яндекс.Недвижимость, который кроме отображения на карте доступного для продажи или аренды жилья даёт так же даёт возможность посмотреть их собственный рейтинг инфраструктуры в виде тепловой карты. Но, к сожалению, он доступен только для Москвы и Санкт-Петербурга.

2.2 Выбор источника данных об инфраструктуре

На данный момент есть несколько сервисов, предоставляющих доступ к картам. Большой популярностью в России пользуются Яндекс.Карты и 2GIS, на западе – Google Maps. Однако это всё закрытые проекты, которые требуют регистрации и соблюдения некоторых ограничений даже в случае некоммерческого использования и не подходят требования проекта.

В качестве источника данных об инфраструктуре был выбран проект OpenStreetMap, предоставляющий большое количество разнообразных географических данных, которые могут быть использованы совершенно свободно.

Наполнение карты (количество обозначенных объектов, актуальность, количество метаданных), а так же её качество (соответствие метаданных принятым стандартам) может варьироваться от города к городу, но в большинстве городов России и почти всех городах Европы достаточно хорошее, чтобы рассчитывать свой собственный рейтинг на основе этих данных.

3 Реализация

3.1 Выбор средства доступа к данным

Основу приложения должна составлять интерактивная карта, позволяющая выбрать на ней некоторый интересующий пользователя участок, на котором в последствии будет просчитаны рейтинги района и отрисована тепловая карта. Но для того, чтобы мы могли провести какие-то вычисления, необходимо сначала получить необходимые для этого данные.

Есть два пути решения этой проблемы. Первый – развернуть свой собственный сервер базы данных, который будет содержать данные того региона, с которым мы работаем. Один из плюсов данного подхода – возможность отправлять запросы на выборку данных или производить поиск маршрутов, обращаясь напрямую к серверу базы данных. Однако, объём данных OSM в несжатом виде со всеми индексами достаточно велик, что потребует наличия на сервере соответствующего количества места на жёстких дисках и довольно много времени для выполнения импорта и предобработки данных. Второй существенный минус – необходимость регулярного обновления базы данных для поддержания данных в актуальном состоянии, что расточительно при небольшом количестве пользователей системы.

Второй путь – не разворачивать свой сервер, а использовать открытые API. OpenStreetMap предоставляет два API – первый, editing API, используется только приложениями для редактирования карт, так как позволяет извлекать самые актуальные данные и вносить изменения на сервер. Он не позволяет извлекать данные с больших территорий и довольно ограничен. Второй, Overpass API, позволяет писать в декларативном стиле довольно сложные запросы к базе и извлекать с их помощью данные сразу на большой территории.

В своём приложении для извлечения необходимых данных мы будем обращаться к одному из общедоступных серверов Overpass API. По заверению их администраторов, они достаточно производительны, чтобы их можно быть использовать, совершенно не волнуясь, что мы можем помешать работе

других приложений, если наше приложение совершает в сутки не более 10 000 запросов и выгружает в сутки не более 5 ГБ данных.

Для тестирования запросов к Overpass API во время выполнения работы использовался сервис Overpass Turbo, позволяющий писать и выполнять запросы, а также отображать результат запроса в текстовом виде и на карте прямо в окне браузера.

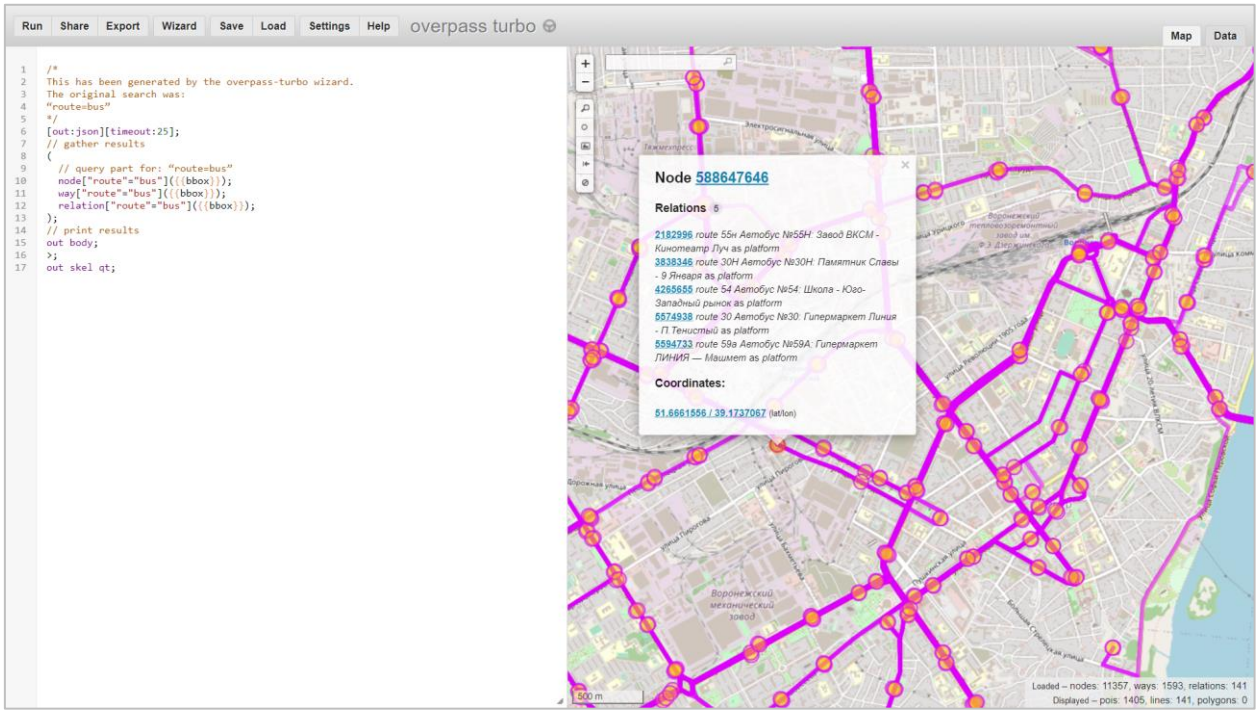


Рисунок 3 – Интерфейс Overpass Turbo с примером запроса к API

Стоит заметить, что несмотря на свои возможности, этот Overpass API позволяет только извлекать объекты, но не проводить с ними вычисления на стороне сервера-интерпретатора запросов. По этой причине невозможно использовать Overpass Turbo как решение поставленной задачи без его доработки.

3.2 Отрисовка тайлов

Аналогичная ситуация обстоит и с сервером для отрисовки тайлов – мелких кусочков изображения, из которых состоит отображаемая пользователю в растровом виде карта. Так как мы не имеем собственного сервера данных OSM и тайлового сервера, обращаться будем к общедоступному серверу MAPNIK, принадлежащего OSM.

3.3 Выбор технологий и архитектуры для приложения

Так как приложение не должно требовать от пользователей для своей работы его установки, логично реализовать его в виде одностраничного веб-приложения. Так же так как мы не имеем собственного сервера базы данных, а обращаемся за ними к общедоступным серверам, используя открытый API, у нас есть возможность производить все необходимые для расчёта рейтингов и отрисовки карты вычисления на стороне клиента, без необходимости иметь собственный сервер в принципе.

Для создания клиентского приложения я решил использовать фреймворк Vue и язык программирования TypeScript, для создания пользовательского интерфейса – библиотеку Vuetify, для отображения интерактивной карты – библиотеку Leaftet.

Выбор языка обусловлен наличием в нём строгой типизации и возможности писать код в привычном объектно-ориентированном стиле, что существенно ускоряет разработку и уменьшает вероятность ошибок. На этапе сборки проекта код на TypeScript компилируется в JavaScript-код, который сможет выполняться в браузере, но в котором уже отсутствует информация о типах. Совместимость TypeScript с JavaScript позволяет использовать JavaScript-библиотеки из TypeScript кода, и наоборот.

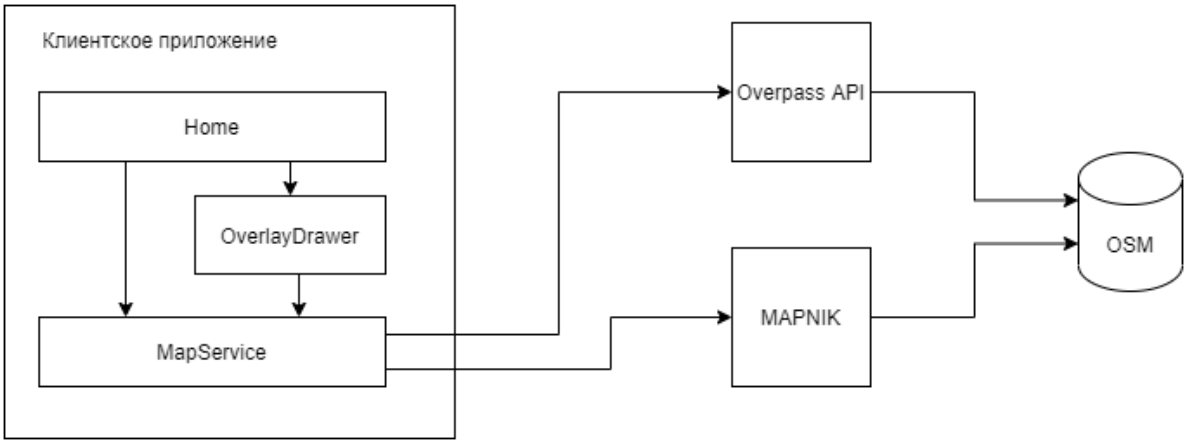


Рисунок 4 – Взаимодействие между компонентами приложения

3.4 Структура данных OSM

В OSM используется только три вида базовых элементов, которыми можно описать любую сущность, требующую представления на карте. Эти элементы – это точка (node), путь (way) и отношение (relation). Точка – элемент, содержащий широту и долготу. Путь – это ломаная линия, соединяющая точки в заданном порядке. Ломаная линия может быть замкнутой (полигон) или не замкнутой. Их часто используют для улиц и других объектов, имеющих геометрию. Отношения задают принадлежность точек, путей или полигонов к определённой группе.

У каждого элемента могут быть заданы атрибуты (теги), которые представляют собой пары ключ=значение. По ним можно понять, что перед нами объект какого-то определённого типа либо извлечь метаинформацию, например, о названии и режиме работы заведения.

3.5 Использование Overpass API

Запросы к Overpass API позволяют получать с сервера данные, удовлетворяющим условиям запроса. Однако, Overpass API обрабатывает каждый возможный вид данных отдельно. Итого, например, чтобы выбрать все элементы с тегом amenity=school нам будет необходимо воспользоваться оператором объединения, поместив внутрь результаты выборки точек, путей и отношений с этим тегом.

На рисунке 3 показан пример запроса к Overpass API, в результате которого возвращаются все автобусные маршруты, которые попали в отображаемую на интерактивной карте в правой половине окна зону.

3.6 Выбор данных об инфраструктуре

Большинство объектов инфраструктуры имеют тег amenities. Объекты, являющиеся магазинами, принято помечать тегом shop. Объекты, относящиеся к системе здравоохранения, обычно помечают тегом healthcare (например, healthcare=hospital или healthcare=pharmacy), но также часто встречаются больницы, поликлиники и другие объекты здравоохранения, помеченные только как amenity, без тега healthcare.

Так же этими тегами помечены большое количество объектов, такие как скамейки, мусорные баки и велосипедные парковки, которым стоит быть обозначенными на картах, но которые явно не стоит учитывать при расчёте рейтинга района с точки зрения инфраструктуры.

В таблице ниже приведены несколько групп объектов, и сценарии использования приложения, при которых эти категории используются при расчёте рейтинга:

Категория	Атрибуты	Сценарии использования
Общественный транспорт	highway=bus_stop	Все
Магазины и рынки	highway=bus_stop shop=convenience shop=supermarket amenity=marketplace	Все
Аптеки	amenity=pharmacy healthcare=pharmacy	Все
Кафе и быстрое питание	amenity=fast_food amenity=cafe	Любой студент
Детские сады	amenity=kindergarten	Семьянин
Школы	amenity=school	Семьянин
Больницы и поликлиники	amenity=school amenity=clinic amenity=hospital amenity=doctor amenity=doctors healthcare=clinic healthcare=hospital	Все

	healthcare=doctor healthcare=doctors healthcare=centre	
Парикмахерские	shop=hairdresser	Семьянин
Кино	shop=hairdresser	Безответственный студент
Пабы и бары	amenity=pub amenity=bar	Безответственный студент
Ночные клубы	amenity=nightclub	Безответственный студент

Так как довольно часто тег может быть установлен не на точке, а на полигоне или отношении, в данном важно извлекать не информацию о геометрии, а о центре объекта.

3.7 Оптимизация запросов

Самый простой способ запросить через Overpass API данные об объектах, удовлетворяющих хотя бы одному из условий – воспользоваться оператором объединения, передав по одному набору данных для каждого отдельного условия. Однако, интерпретатор Overpass не выполняет оптимизацию таких запросов перед исполнением, в результате чего для извлечения требуется в несколько раз больше времени, чем если бы запрос использовал всего одно условие, использовавшее бы для фильтрации данных регулярные выражения.

Overpass API поддерживает два варианта фильтрации атрибутов с помощью регулярных выражений. Первый – атрибут имеет фиксированный ключ, но значение проверяется с помощью регулярного выражения. Второй – и ключ, и значения проверяются с помощью двух отдельных заданных им регулярных выражений.

Используя первое свойство, можно оптимизировать запрос на выборку данных, у которых значение одного из атрибутов может принимать несколько значений. Ниже приведён пример, показывающий суть оптимизации:

```
//Использование оператора объединения, 3 раздельных условия
[out:json][bbox:{{bbox}}];
(  node[amenity=school];
  node[amenity=kindergarten];
  node[shop=hairdresser];);
out;

//Объединение + регулярное выражение для группировки значений, 2 условия
[out:json][bbox:{{bbox}}];
(  node[amenity~"^(school|kindergarten)$"];
  node[shop=hairdresser];);
out;
```

Учитывая, что множества возможных значений для атрибутов с ключами amenity, shop и healthcare практически не пересекаются, можно дополнительно ускорить запрос, отказавшись от использования оператора объединения и используя всего один большой запрос с двумя регулярными выражениями. В этом случае сервер вернёт некоторое небольшое количество лишней информации, но запрос всё равно будет выполнен заметно быстрее. Пример такого запроса показан ниже:

```
//Использование всего одного условия с регулярными выражениями
[out:json][timeout:25][bbox:{{bbox}}];
```

```
node[~"^(amenity|shop)$"~"^(school|kindergarten|hairdresser)$"];
out;
```

Использование этой особенности интерпретатора запросов позволило ускорить обработку запроса на сервере более чем в семь раз по сравнению с запросом, выбирающим данные без использования регулярных выражений и использующим оператор объединения.

На графиках ниже показано среднее время выполнения и объём возвращённых в ответ данных нескольких вариантов запросов, извлекающих данные об инфраструктуре в европейском городе средней величины.

Рисунок 5 – Среднее время выполнения сформулированных разными способами запросов

Рисунок 6 – Объем передаваемых данных в результате выполнения запросов

3.8 Расчёт рейтинга

Пусть максимальный допустимый рейтинг для категории будет равен 5 и выставляется только в том случае, если до ближайшего объекта заданной категории можно дойти пешком по прямой не более чем за две с половиной минуты, считая, что средняя скорость пешехода равна 5 км/ч.

Рейтинг должен убывать достаточно быстро, чтобы на карте была видна разница между максимальным и почти максимальным рейтингами и области без инфраструктуры не были затронуты высоким рейтингом из соседних областей.

Под требования такой функции будет подходить ограниченная сверху гипербола, в числителе которой будет коэффициент, равный 2,5 * 5, а в знаменателе – время. Максимальное значение функции рейтинга – 5.

Время пешком от одной точки до другой по прямой легко находится из расстояния между точками, а расстояние между точками может быть вычислено по формуле гаверсинусов, зная координаты точек.

Общий рейтинг в точке можно представить в виде среднего арифметического рейтингов, рассчитанных для каждой из применимых к рассматриваемому случаю категорий.

3.9 Построение тепловой карты рейтинга

В случае, если мы хотим видеть на карте сразу много рейтингов одновременно, лучшим решением будет обозначить их цветом. Выбранная в моём случае цветовая гамма состоит из прозрачного и цветов радуги от синего до красного, причём прозрачный соответствует минимальному (нулевому) рейтингу в точке, а красный – максимальному.

Градиент, который будет использоваться при отрисовке карты рейтинга, просчитывается заранее и сохраняется в памяти в виде картинки шириной 1 и высотой 256 пикселей. Каждое возможное значение рассчитанного рейтинга будет соответствовать одному из цветов градиента.

Отрисовка тепловой карты выполняется в несколько этапов. Сначала рисуется картинка с низким разрешением, используя квадраты со стороной 8 пикселей, а затем детализация увеличивается с помощью перерисовки, используя квадраты со стороной 2 пикселя.

Сразу после каждого этапа отрисовки полученная картинка прикрепляется поверх отображаемой карты, замещая старую. Этот подход оказался более простым и производительным, чем использование сторонних плагинов.

При формировании тепловой карты работа ведётся не элементом Canvas, предоставляющим инструменты рисования в браузере, а напрямую с массивами байт, представляющих данную картинку в несжатом виде.

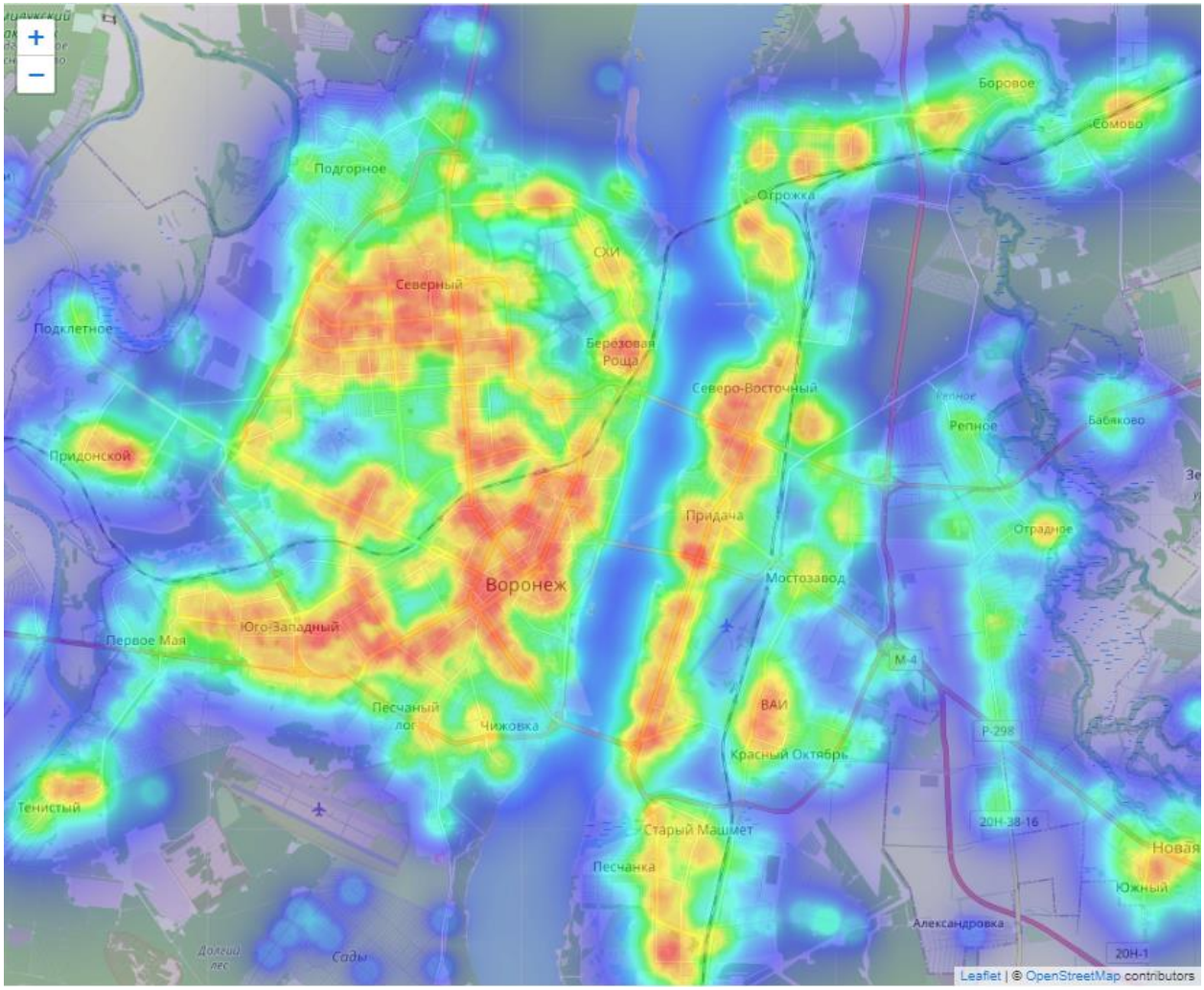


Рисунок 7 – Тепловая карта рейтингов районов с точки зрения доступности инфраструктуры

3.10 Оптимизация алгоритмов

Нахождение точного расстояния между двумя точками по их координатам – довольно затратная операция, которую не следует вызывать для каждой точки для каждой проверки. Поэтому, чтобы уменьшить количество вычислений, соотношение между расстоянием в метрах и расстоянием в градусах определяется один раз в момент перед загрузкой данных. Затем все вычисления проводятся для точек с координатами в полученных «условных» метрах.

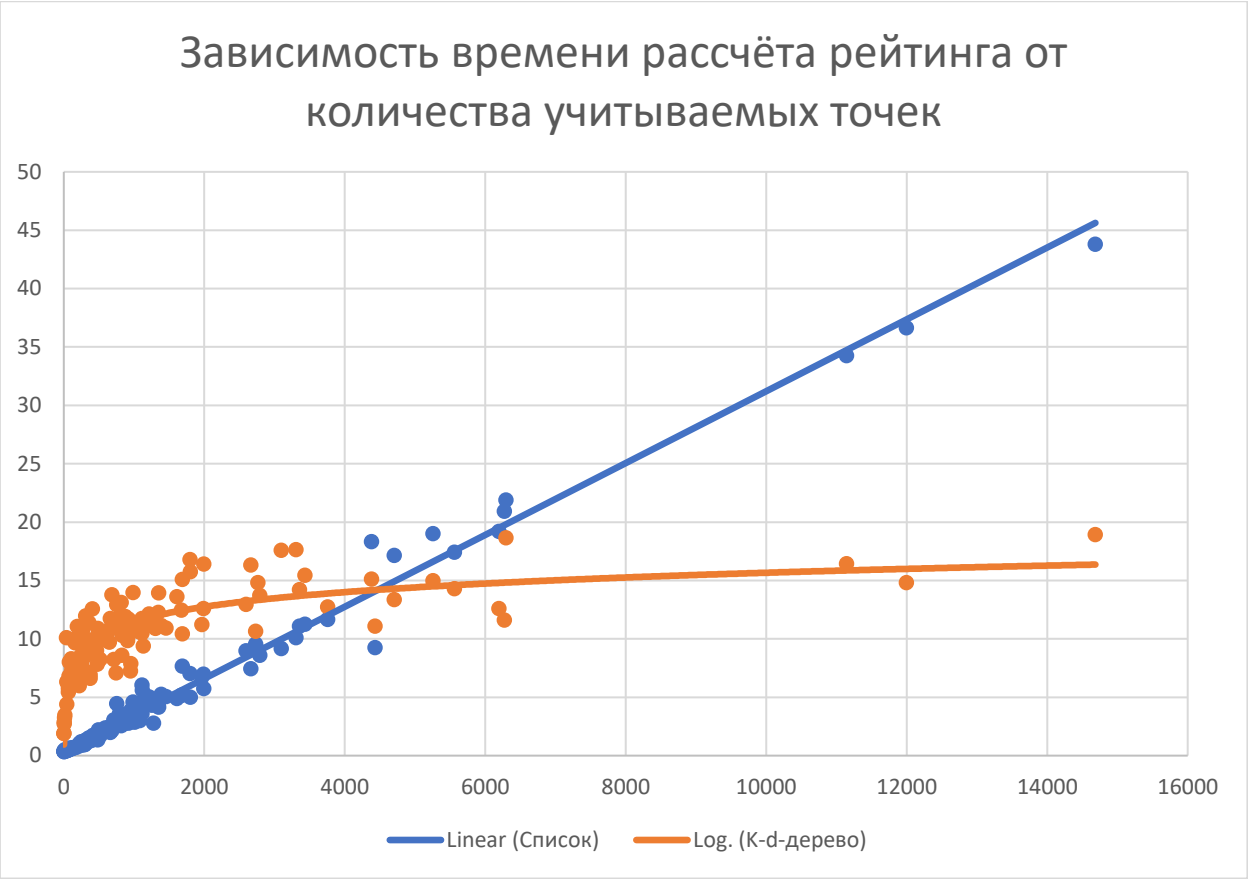
Метры «условные» так как текущее соответствие будет верным только для той широты, на которой было первоначально вычислено это соответствие.

В то же время, вблизи этого места искажения будут довольно малы, чтобы эти значения можно было всё так же использовать для оценки расстояний.

При сравнении расстояний используется метрика, равная сумме квадратов разностей координат точек (расстояние в квадрате). Квадратный корень вычисляется только при необходимости посчитать точное значение уже непосредственно для расчёта рейтинга.

Другая операция, которая могла занимать продолжительное время в случае работы с большим количеством данных – это поиск ближайшего объекта, принадлежащего заданной категории. На малых объёмах данных линейный поиск справлялся за приемлемое время, но при попытке отрисовать карту рейтинга, например, для всей области этот процесс оказывался слишком медленным.

Чтобы проверить возможность ускорения отрисовки при большом объёме данных, вместе с линейными списками данные параллельно сохранялись в структуру, представляющую собой k-d-дерево. Затем была проведена отрисовка карты на разных масштабах, с разным количеством учитываемых данных. Полученные в ходе замеров производительности данные были усреднены для каждой величины рассматриваемого в момент поиска списка/дерева и использованы для построения графика.



Графики, построенные по точкам, полученным в ходе эксперимента, пересекаются при количестве элементов в списке/дереве немного большем 4000. Это означает, что начиная с этого значения, использование дерева требует меньше времени и ресурсов для поиска ближайшего элемента, чем при линейном поиске в списке.

Заключение

В ходе курсовой работы были изучены существующие решения, позволяющие оценить качество рассматриваемых районов города, проанализированы их преимущества, недостатки и ограничения.

С учётом полученных опыта существующих решений, было спроектировано, разработано и оптимизировано приложение, позволяющее построить карту рейтинга жилых районов любого города, используя данные об инфраструктуре, получаемые из OpenStreetMap и удовлетворяющее установленным до начала разработки требованиям.

Список использованных источников

1. Лучшие и худшие районы Воронежа
   https://www.domofond.ru/statya/luchshie_i_hudshie_rayony_voronezha/7557
2. Исследования Яндекса – Москва для жизни и для развлечений
   https://yandex.ru/company/researches/2017/moscow_districts
3. OpenStreetMap Wiki – Elements
   https://wiki.openstreetmap.org/wiki/Elements
4. OpenStreetMap Wiki – Overpass API
   https://wiki.openstreetmap.org/wiki/Overpass_API
5. OpenStreetMap Wiki – Overpass QL
   https://wiki.openstreetmap.org/wiki/Overpass_API/Overpass_QL
6. OpenStreetMap TagInfo
   https://taginfo.openstreetmap.org/