ВВЕДЕНИЕ 6
1 Научные исследования в области имитационного моделирования 8
1.1 Математические модели, применяемые для анализа транспортной инфраструктуры 8
1.2 Пропускная способность автомобильных дорог 10
1.3 Основные теоретические положения метода имитационного моделирования движения транспортных потоков 12
1.4 Сравнение с представленными программными решениями на рынке 14
2 Графовая модель дорожной сети 16
2.1 Моделирование транспортной сети 16
2.2 Принципы изменения суточной активности городского трафика 18
3 Проектирование веб-приложения 27
3.1 Функциональные требования к системе 27
3.2 Модель данных 29
3.3 Инструменты реализации 33
3.4 Поведение системы 35
3.5 Разработка прототипа 49
4 Проектный этап разработки системы 62
4.1 Модуль работы с картами. 62
4.2 Модуль работы с базой данных 70
4.3 Вспомогательный модуль для проведения необходимых расчетов во время этапа подготовки к моделированию 72
4.4 Модуль работы с моделированием движения городского трафика. 73
4.5 Модуль обработки действий пользователя. 75

4.6 Модуль, содержащий начальные настройки и параметры системы..... 79

4.8 Модуль генерации рекомендаций по улучшению структуры УДС. 81

5 Реализация имитационной модели движения транспортного потока абстрактно и на примере нескольких районов города. 84

5.1 Постановка задачи моделирования и подготовка имитационной модели. 84

5.2 Имитационное моделирование дорожно-транспортных сетей 86

5.3 Результаты работы системы имитационного моделирования 91

ЗАКЛЮЧЕНИЕ 93

ВВЕДЕНИЕ

Современные автомобильные дороги представляют собой сложные инженерные сооружения, которые должны обеспечивать максимально возможное безопасное движение автомобилей с любыми скоростями. Тем не менее, ежегодно в России в дорожно-транспортных происшествиях погибает свыше 40 тысяч человек. Материальный ущерб от дорожно-транспортных происшествий достигает 10% годового национального дохода. Это свидетельствует о неблагоприятной дорожно-транспортной обстановке на дорогах страны.

Улично-дорожные сети (УДС) являются очень дорогим и трудноизменяемым элементом городской инфраструктуры. Их проектирование относят к числу наиболее сложных вопросов теории транспортной планировки субъектов и городов. Обоснование любых градостроительных решений, связанных с изменением дорожной сети, включает детальный анализ существующего состояния УДС, поэтому оценка ее состояния предшествует многим видам градостроительного проектирования, разработкам градостроительных регламентов и зонированию городских территорий, является обязательным элементом комплексных схем организации движения (КСОД), проектов реконструкции УДС и организации дорожного движения (ПОД).

Для оценки состояния и проектов реконструкции УДС можно применить имитационное моделирование транспортных сетей. Основная задача, которую решает имитационное моделирование – прогнозирование и определения основных параметров транспортной сети, таких как интенсивность движения, объемы пассажиро- и грузоперевозок, средняя скорость движения и т. д.

Целью проекта является разработка алгоритмов имитирования автомобильного потока дорожно-транспортной сети. Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать литературу и научные исследования в области имитационного моделирования.

2. Провести анализ конкурентов в выбранной области, изучить методы, принципы и технологии имитационного моделирования транспортных потоков.

3. Реализовать алгоритмы изменения суточной активности движения автомобилей и программный комплекс, работающий на его основе.

4. Реализовать имитационную модель движения транспортного потока абстрактно и на примере нескольких районов города.

Результатом моделирования станет созданная модель транспортной сети города, реализующая задачу ее качественного и количественного развития, обеспечивающую эффективную работу городского хозяйства и безопасность участников дорожного движения, и экологическую безопасность жителей.

1 Научные исследования в области имитационного моделирования

1.1 Математические модели, применяемые для анализа транспортной инфраструктуры

Транспортная инфраструктура - одна из важнейших инфраструктур, обеспечивающих жизнь городов и регионов. В последние десятилетия во многих крупных городах исчерпаны или близки к исчерпанию возможности экстенсивного развития транспортных сетей. Поэтому особую важность приобретает оптимальное планирование сетей, улучшение организации движения, оптимизация системы маршрутов общественного транспорта.

Решение таких задач невозможно без математического моделирования транспортных сетей. Главная задача математических моделей — определение и прогноз всех параметров функционирования транспортной сети, таких как интенсивность движения на всех элементах сети, объемы перевозок в сети общественного транспорта, средние скорости движения, задержки и потери времени и т.д. Математические модели, применяемые для анализа транспортных сетей, весьма разнообразны по решаемым задачам, математическому аппарату, используемым данным и степени детализации описания движения.

Основываясь на функциональной роли моделей, т.е. на тех задачах, для решения которых они применяются, можно условно выделить три основные класса: прогнозные модели, имитационные модели, оптимизационные модели. Прогнозные модели предназначены для решения следующей задачи. Пусть известны геометрия и характеристики транспортной сети, а также размещение потокообразующих объектов в городе. Необходимо определить, какими будут транспортные потоки в этой сети. Более подробно, прогноз загрузки транспортной сети включает в себя расчет усредненных характеристик движения, таких как объемы межрайонных передвижений, интенсивность потока, распределение автомобилей и пассажиров по путям движения и др.

При помощи этих моделей можно прогнозировать последствия изменений в транспортной сети или в размещении объектов. В отличие от этого имитационное моделирование ставит своей целью воспроизведение всех деталей движения, включая развитие процесса во времени. При этом усредненные значения потоков и распределение по путям считаются известными и служат исходными данными для этих моделей. Кратко это отличие можно сформулировать так: прогнозные модели отвечают на вопрос: «сколько и куда» будут ехать в данной сети, а имитационные модели отвечают на вопрос: как в деталях будет происходить движение, если известно в среднем, «сколько и куда». Таким образом, прогноз потоков и имитационное моделирование являются дополняющими друг друга направлениями.

Из сказанного следует, что к классу имитационных по их функциональной роли можно отнести широкий спектр моделей, известных под названием модели динамики транспортного потока. В моделях этого класса может применяться разная техника — от имитации движения каждого отдельного автомобиля до описания динамики функции плотности автомобилей на дороге. Для динамических моделей характерна значительно большая детализация описания движения и, соответственно, потребность в больших вычислительных ресурсах.

Применение этих моделей позволяет оценить динамику скорости движения, задержки на перекрестках, длины и динамику образования «очередей» или «заторов» и другие характеристики движения. Основные области практического применения динамических имитационных моделей — улучшение организации движения, оптимизация светофорных циклов и др. В настоящее время актуальной задачей является разработка систем автоматизированного оперативного управления движением, работающих в режиме реального времени. Такие системы должны использовать информацию с датчиков в сочетании с динамическим имитационным моделированием.

Однако помимо практических применений, развитие динамических моделей представляет большой научный интерес в связи с изучением

транспортного потока как физического явления со сложными и нетривиальными свойствами. Среди таких свойств - спонтанная потеря устойчивости, явления самоорганизации и коллективного поведения и др. Модели прогноза потоков и имитационные модели ставят своей целью адекватное воспроизведение транспортных потоков.

УДС изучались многими отечественными специалистами. Так различным аспектам проектирования и оценки УДС посвящены исследования: А.А. Агасьянца, В.Ф. Бабкова, Ю.П. Бочарова, С.А. Ваксмана, Ф.Г. Глика, Т.А. Глухаревой, Г.Е. Голубева, З.И. Горбанева, Д.Р. Гришкявючене, И.И. Демкина, С.Г. Елизарова, Г.Н. Зубкова, Ю.В. Игнатьева, А.М. Костина, А.Н. Красникова и др.

Критерии эффективности функционирования УДС и качества организации дорожного движения (ОДД) рассмотрены в работах: М.Б Афанасьева, Р.О. Брайловского, В.Е. Верейкина, Б.Н. Грановского, О.А. Дивочкина, В.В Зырянова, И.Е. Капитанова, Г.И. Клинковштейна, В.И. Коноплянко, Ю.Ф. Кременца и др.

1.2 Пропускная способность автомобильных дорог

Пересечения автомобильных дорог являются одним из участков, на которых сосредотачиваются дорожно-транспортные происшествия, значительно уменьшается пропускная способность, наблюдается снижение скорости, а, зачастую, и полная остановка движения автомобильного транспорта (заторы).

Пропускная способность транспортного узла автомобильных дорог – это наибольшее количество автомобилей, которое может пропустить узел в единицу времени по всем направлениям. За единицу времени обычно принимают один час. Большинство узлов в одном уровне имеют меньшую пропускную способность чем подходящие к нему дороги вследствие наличия на узле опасных

точек, задержки автомобилей перед светофорами, слияния потоков движения на соединительных линиях в пределах узлов и ряда других причин.

Каждой интенсивности движения по главному направлению соответствует определенное число автомобилей второстепенного направления, которые могут пересечь или влиться в основной поток. В связи с этим понятие пропускная способность пересечения означает возможные соотношения интенсивностей движения на пересекающихся улицах или дорогах. Это соотношение определяется интенсивностью движения по главному направлению. Число автомобилей, прошедших по второстепенному направлению, зависит от того, насколько полно используются интервалы между автомобилями основного потока. В связи с этим следует различать несколько значений пропускной способности пересечений.

Теоретическая пропускная способность нерегулируемого пересечения — это максимальная интенсивность второстепенного направления (при конкретной интенсивности главного направления), которая может быть достигнута при идеальных условиях движения на пересекающихся дорогах и на пересечении, и при полном использовании всех интервалов в основном потоке. Последнее условие выполнимо лишь при постоянном наличии на второстепенном направлении очереди автомобилей в количестве, достаточном для заполнения любого интервала в основном потоке.

Возможная пропускная способность нерегулируемого пересечения — это максимальная интенсивность движения второстепенного направления (при конкретной интенсивности главного направления) с учетом реальных условий движения на пересекающихся дорогах и на пересечении при полном использовании всех интервалов в основном потоке. Для достижения возможной пропускной способности также необходимо наличие постоянной очереди ожидающих автомобилей на второстепенном направлении достаточно большой длины.

Практическая пропускная способность нерегулируемых пересечений — это максимальная интенсивность движения второстепенного направления (при конкретной интенсивности главного направления) с учетом реальных дорожных условий, состава транспортного потока и практического (не всегда полного) использования интервалов в основном потоке. При этом могут наблюдаться очереди на второстепенном направлении, но не постоянные, как при теоретической или возможной пропускной способности, и меньшей длины.

1.3 Основные теоретические положения метода имитационного моделирования движения транспортных потоков

Под имитацией, по определению Т. Нейлора, понимается численный метод проведения на цифровых вычислительных машинах экспериментов с математическими моделями, описывающими поведение систем в течение продолжительных периодов времени. Движение автомобилей по дороге представляет собой вероятностный процесс, который можно описать с помощью стохастических моделей. Математическая модель реального процесса – это совокупность соотношений, которые связывают характеристики процесса функционирования системы с исходной информацией и начальными условиями.

Впервые математическая теория транспортных потоков была предложена в работе известного американского специалиста Ф. Хейта, где установлено соотношение между положениями теории транспортных потоков и теорией вероятности, а также обобщённые математические модели движения транспорта. В известной монографии Д. Дрю [8] приведены основные математические модели движения транспортных потоков, введено понятие времени ожидания на участке въезда на автомагистраль в зоне слияния транспортных потоков и дано решение в интегральной форме. Дальнейшее развитие теория транспортных потоков получила в работах известных Российских учёных В.Ф. Бабкова, В.В. Сильянова, Е.М. Лобанова, где разработаны теоретические основы нового

направления изучения транспортных потоков – унифицированное имитационное моделирование на ЭВМ движения транспортного потока как сложной системы.

Прогнозирование режимов движения транспортных потоков может осуществляться на основе информационной модели восприятия дорожных условий и учётом стохастического характера процесса движения. Алгоритм прогнозирования содержит в себе элементы статистического моделирования, информационные математические модели, а также динамические модели движения.

Многие из вышеперечисленных методов исследования транспортных потоков представляют собой сложные унифицированные имитационные модели, весьма трудные для понимания и малопригодные для решения практических задач. Поэтому в нашей работе представлены более простые математические модели на основе метода статистических испытаний (метод Монте-Карло), но как показали полученные результаты, вполне удовлетворительные для решения различных практических задач.

Сущность метода статистического моделирования или метода статистических испытаний (его также называют методом Монте-Карло), состоит в построении для исследуемого процесса моделирующего алгоритма, имитирующего поведение элементов сложной системы и взаимодействия между ними. Одно такое воспроизведение (попытка) функционирования системы называют реализацией или испытанием, которые представляют собой, как бы, срез во времени. После каждого испытания регистрируют совокупность параметров, характеризующих случайный исход реализации.

В результате многократного повторения реализаций получают статистический материал, позволяющий выявить устойчивые закономерности процесса функционирования системы. Чем больше таких реализаций, тем точнее можно определить конечный результат. Результаты моделирования, полученные при воспроизведении единственной реализации процесса в силу действия случайных факторов, будут реализациями случайных величин и не могут

объективно характеризовать процесс. Поэтому искомые величины при исследовании процессов методом статистического моделирования необходимо определять, как статистические оценки по данным большого числа реализаций процесса.

Основой метода имитационного моделирования движения транспортных потоков явилось положение об экспоненциальном законе распределения интервалов между движущимися автомобилями в этих потоках, впоследствии подтверждённое экспериментальными данными. Большим преимуществом экспоненциального распределения является то, что оно однопараметрическое. Им чрезвычайно удобно пользоваться, особенно при проектировании элементов автомобильных дорог, когда нам не известен действительный закон распределения.

1.4 Сравнение с представленными программными решениями на рынке

Существующие инструменты являются слишком сложными для использования инженерами-транспортниками, либо не предоставляют необходимых возможностей расширения. Графический конструктор разработанной среды упрощает процесс моделирования. В основу среды положена модель поведения участника движения.

Возрастающие автотранспортные потоки больших городов диктуют необходимость в реконструкции существующих и создании новых транспортных узлов. Имитационное моделирование – эффективный способ анализа пропускной способности и других характеристик небольших и средних транспортных систем.

В настоящее время существуют несколько средств моделирования транспортных систем. Все системы, кроме Transport AnyLogic, являются

зарубежными разработками и имеют высокую стоимость, часто неприемлемую для отечественных проектных организаций и муниципальных образований.

Моделирование во многих средах (VISSIM, Transmodeler) предоставляет богатый набор возможностей, но часто является неоправданно сложным и недоступным специалистам в предметной области, так как предполагает глубокое знание технологий имитационного моделирования. Некоторые (SIDRA Intersection) средства позволяют моделировать только отдельные типы объектов транспортной инфраструктуры.

В этих условиях представляется целесообразным создание новой среды, обладающей следующими свойствами:

- простота создания моделей, ориентированность на специалистов предметной области, а не на инженеров-имитационщиков;
- наглядность моделирования, максимальное визуальное сходство модели и моделируемой системы;
- реализация в виде web-приложения (доступность системы пользователю не зависит от операционной системы);
- расширяемость моделей, предоставление возможности обогащать набор средств разработки моделей.

Среду моделирования транспортных систем было решено реализовать, приняв за основу агентный подход. Возможности визуализации обеспечивают наглядность как на этапе создания модели, так и при проведении экспериментов. Разработана общая структура среды моделирования транспортных систем.

2 Графовая модель дорожной сети

2.1 Моделирование транспортной сети

Модель транспортной сети может быть представлена в виде графа. Граф – это фигура, состоящая из точек (вершин) и соединяющих их отрезков (звеньев).

Вершины графа – это точки на сети, наиболее важные для определения расстояний или маршрутов движения.

Звенья графа – это отрезки транспортной сети, характеризующие наличие дорожной связи между соседними вершинами. Звенья графа характеризуются числами, которые могут иметь различный физический смысл. Чаще всего это расстояние, но может использоваться, например, и время движения. Ориентированные по направлению звенья графа называются дугами. Всякое неориентированное звено графа включает в себя две равноценные, но противоположно направленные дуги.

Граф, каждая вершина которого может быть соединена некоторой последовательностью звеньев с любой его другой вершиной, называется связанным графом. Иначе говоря, каждая вершина связанного графа должна иметь как минимум одну входящую и одну выходящую дугу.

Граф, моделирующий транспортную сеть, обязательно должен быть связанным, чтобы всегда был путь из любой вершины в любую другую вершину. Числа, характеризующие звенья такого графа, обычно выражают протяженность пути, время или стоимость проезда.

Для моделирования транспортной сети необходимо:

- картографический материал; обычно это карты крупного масштаба, так как они позволяют с большой точностью делать замеры расстояний между пунктами;
- сведения о размещении основных объектов транспортной системы и ее среды (в зависимости от решаемой задачи: грузообразующие и

грузопоглощающие предприятия, жилые массивы, места приложения труда и т. п.);

– дополнительные сведения из коммунальных и дорожных организаций в виде перечня улиц с характеристикой их проезжей части;

– сведения по организации дорожного движения, т. е. схемы организации движения на перекрестках, площадях и транспортных развязках, а также сведения о различных ограничениях движения, связанных с установленными дорожными знаками.

Основной проблемой при моделировании транспортной сети является выбор уровня детализации. Здесь приходится искать компромисс между точностью и затратами на исследования. Вероятно, 100 % точность будет обеспечена, если будут учтены индивидуально маршрут поездок всех пользователей в течение всех дней в году. В то же время очевидна излишняя детализация такого подхода, так как во многих случаях будут совпадать точки отправления (остановочные пункты, гаражи, склады), точки прибытия и маршруты следования. В аспекте времени тоже будет проявляться общность поведения пользователей. Для преодоления этих противоречий используется транспортное зонирование.

Транспортное зонирование – это способ агрегирования индивидуальных потребностей пользователей в использовании транспортной сети для целей моделирования. Транспортное зонирование имеет две взаимосвязанные характеристики: количество зон, описывающих сеть, и их размер. Чем больше зон, тем соответственно меньше размер транспортных зон, которыми будет описываться транспортная сеть.

Как правило, для решения разных задач используется транспортное зонирование с различным количеством зон. При решении стратегических задач транспортного планирования используется меньшее количество зон, каждая зона покрывает достаточно большую площадь, например, несколько жилых

кварталов. При анализе конкретных транспортных проблем используют большое количество небольших по размеру зон для детализации ситуации на сети.

2.2 Принципы изменения суточной активности городского трафика

Эффективность автомобильных перевозок и безопасность дорожного движения в значительной мере определяются характером взаимосвязей в потоке движущихся транспортных средств и основными характеристиками этого потока. Незнание природы таких взаимосвязей ограничивает возможности управления транспортным потоком и предотвращение ДТП.

Транспортный поток — это совокупность транспортных средств, движущихся по проезжей части дороги. В зависимости от числа полос и разрешенных направлений движения транспортный поток подразделяют на следующие виды:

- однополосный односторонний;
- двухполосный односторонний или двусторонний;
- трехполосный односторонний или двусторонний;
- четырехполосный (и более) односторонний или двусторонний.

В зависимости от вида транспортного потока будут различаться возможности для маневрирования, условия движения транспортных средств и т. п.

Наиболее востребованными и часто применяемыми характеристиками транспортного потока являются интенсивность, скорость движения, плотность потока, его состав по типам транспортных средств.

Интенсивность движения $\boldsymbol{N_a}$ определяется числом транспортных средств (автомобилей), движущихся в определенном направлении или направлениях по данной полосе или дороге и проходящих через пункт наблюдения за фиксированный промежуток времени. Определение интенсивности движения составляет основу оценки состояния транспортного потока.

Интенсивность движения является главным показателем при определении уровня загруженности различных дорог.

При изучении интенсивности движения по времени суток движения определяют такой параметр, как неравномерность транспортного потока - его распределение по времени и направлениям (рисунок 2.1).

Рисунок 2.1 – Изменение интенсивности движения по времени суток

При расчетах обычно пользуются данными об интенсивности движения в часы пик и среднесуточной интенсивности движения за год. При определении эффективности дорожной сети различают два аспекта, один из которых связан с учетом основных характеристик отдельных дорог, а другой - с оценкой эффективности дорожной сети в целом.

Постоянный мониторинг интенсивности движения позволяет своевременно планировать работы по изменению организации дорожного движения, модернизации и реконструкции дороги. Данные работы проводятся в соответствии с Руководством по прогнозированию интенсивности движения на автомобильных дорогах (утверждено распоряжением Минтранса России 19.06.2003 № ОС-555-р), которое предназначено для расчета существующей и прогнозирования ожидаемой на соответствующий перспективный период

среднегодовой суточной интенсивности движения на участках сети автомобильных дорог общего пользования, дорожной сети городов и т.д.

Интенсивность движения — это непрерывно изменяющаяся величина, даже при очень низкой часовой интенсивности движения могут наблюдаться кратковременные интервалы, за которые через данный пункт будет проходить сравнительно много транспортных средств.

На дорогах с более высоким уровнем интенсивности движения меньше неравномерность движения и стабильнее интенсивность в пиковые периоды.

Для двухполосных дорог с движением в обоих направлениях общая интенсивность обычно характеризуется суммарным значением встречных потоков, так как условия движения и, в частности, возможность обгонов определяются загрузкой обеих полос.

Если же дорога имеет разделительную полосу и встречные потоки изолированы друг от друга, суммарная интенсивность встречных направлений не определяет условия движения, а характеризует лишь суммарную работу дороги как сооружения. Для таких дорог имеет значение интенсивность движения в каждом направлении.

При решении различных вопросов регулирования дорожного движения, особенно в городских условиях, большое значение имеет не только суммарная интенсивность потока по данному направлению, но и интенсивность движения, приходящаяся на одну полосу, - так называемая удельная интенсивность движения $\boldsymbol{M_a}$.

Если известно конкретное распределение интенсивности движения по полосам, и оно существенно неравномерно, в качестве расчетной интенсивности $\boldsymbol{M_a}$. можно принять интенсивность движения по наиболее загруженной полосе.

Во многих городах дорожная сеть не соответствует возросшей интенсивности движения. На перекрестках возникают заторы, жизнь пешеходов подвергается опасности, оставленные у тротуаров автомобили сильно затрудняют движение транспортных средств.

В оценку уровня загруженности дорог входят следующие взаимосвязанные факторы: скорость движения и время, затрачиваемое на поездку; непрерывность движения; свобода маневрирования; безопасность и удобство управления транспортным средством. Интенсивность движения влияет на все эти факторы, причем с увеличением интенсивности ее отрицательное влияние усиливается. Когда фактическая интенсивность движения по дороге приближается к максимально возможной, увеличивается опасность заторов.

Затор — это качественное понятие, которое связано с количественной характеристикой, называемой плотностью транспортного потока $\boldsymbol{q_a}$.

Плотность транспортного потока является пространственной характеристикой, определяющей степень стесненности движения на полосе дороги. Ее измеряют числом транспортных средств, приходящихся на 1 км протяженности дороги.

Предельная плотность транспортного потока достигается при неподвижном состоянии колонны транспортных средств, расположенных вплотную друг к другу на полосе.

Предельное значение плотности транспортного потока qmax составляет **170 - 200 авт./км** в зависимости от состава транспортного потока.

При разных значениях плотности движения могут складываться разные уровни эксплуатационных условий по степени стесненности. В зависимости от плотности транспортного потока движение по степени стесненности подразделяют на свободное, частично связанное, насыщенное и колонное.

Численные значения $\boldsymbol{q_a}$ в физических единицах (автомобилях), соответствующих этим состояниям транспортного потока, весьма существенно зависят от параметров дороги, в первую очередь от ее плана и профиля, коэффициента сцепления, а также состава транспортного потока по типам транспортных средств, что, в свою очередь, влияет на выбираемую водителями скорость движения.

Скорость движения $\boldsymbol{v_a}$ является важнейшим показателем транспортного потока, так как цель всех мероприятий по организации дорожного движения - обеспечение скорости транспортного потока, наиболее приближенной к максимально возможной из условий безопасности дорожного движения.

В практике организации дорожного движения в зависимости от методов измерения и расчета рассматривают:

мгновенную скорость движения $\boldsymbol{v_a}$ - скорость, фиксируемую в отдельных типичных сечениях (точках) дороги. Именно мгновенная скорость движения в значительной степени влияет на безопасность движения, поскольку определяет кинетическую энергию автомобиля, т. е. его тормозной путь и время, которое имеется у водителя для оценки опасной ситуации;

максимальную скорость движения $\boldsymbol{v_m}$ - наибольшую мгновенную скорость движения, которую может развить транспортное средство. Для дорожного движения большое значение имеет максимальная скорость движения транспортного средства, которая ниже разрешенной. Такие транспортные средства становятся препятствием для нормального движения транспортного потока;

крейсерскую скорость движения $\boldsymbol{v_k}$ - скорость, с которой водитель стремится ехать в данных условиях. Если транспортный поток движется более медленно или более быстро, водитель испытывает дискомфорт. В зависимости от типа личности водитель быстрее ощущает усталость, становится невнимательным или раздражительным;

разрешенную скорость движения $\boldsymbol{v}_{\text{раз}}$ - скорость, разрешенную на данном участке дороги нормативными документами или средствами регулирования дорожного движения;

рекомендуемую скорость движения $\boldsymbol{v}_{\text{рек}}$ - скорость, с которой рекомендуется двигаться водителю, и которая обеспечивает безопасность дорожного движения в данных условиях;

безопасную скорость движения $\boldsymbol{v}_{\text{б.д.}}$. - скорость, при которой водитель в состоянии предпринять необходимые действия при возникновении опасной ситуации. Соблюдение безопасной скорости движения с большой вероятностью позволяет гарантировать безопасность поездки;

экономичную скорость движения $\boldsymbol{v}_{\text{экн}}$ - скорость, при которой затраты на движение (в основном расход топлива) минимальны;

Между параметрами транспортного потока существуют определенные зависимости. Характер этих зависимостей достаточно сложный, на них влияет громадное количество факторов, связанных не только непосредственно с транспортным потоком, но и с условиями его движения по дороге, метеоусловиями, временем года и суток и т. п. При исследовании транспортных потоков влияющие на них факторы могут рассматриваться как детерминированные или как вероятностные величины. Вероятностный подход более близок природе транспортного потока, но сложен для математического описания. Детерминированный подход легче реализовать в инженерных методиках, и при тщательном анализе исходных данных он дает достаточно точные для практики результаты.

Подход к изучению транспортных потоков рассматривает его как целостный процесс, характеризуемый только внешними параметрами. При таком подходе создаются макроскопические модели, которые рассматривают такие характеристики транспортного потока как скорость, интенсивность, плотность движения и т.п.

Между скоростью движения, плотностью и интенсивностью движения существует соотношение $N_a = U_a * q_a$, которое графически может быть изображено в виде так называемой основной диаграммы транспортного потока (рис. 2.2).

Рисунок 2.2 – Основная диаграмма транспортного потока

Диаграмма отражает изменение состояния однорядного транспортного потока преимущественно легковых автомобилей в зависимости от увеличения его интенсивности и плотности. Левая часть кривой (показана сплошной линией) отражает устойчивое состояние транспортного потока, при котором по мере увеличения плотности транспортный поток проходит фазы свободного, затем частично связанного и связанного движения, достигая точки максимально возможной интенсивности - пропускной способности (точка $N_{a\,max} = P_a$). В процессе этих изменений скорость транспортного потока падает - она характеризуется тангенсом угла наклона α радиуса-вектора, проведенного от точки 0 к любой точке кривой, характеризующей изменениеN_a.

Соответствующие точке $N_{a\,max} = P_a$ значения плотности и скорости транспортного потока считаются оптимальными по пропускной способности ($q_{a\,opt} = u_{a\,opt}$). При дальнейшем росте плотности (за точкой p_a перегиба кривой) транспортный поток становится неустойчивым (эта ветвь кривой показана штриховой линией).

Переход транспортного потока в неустойчивое состояние происходит вследствие снижения плавности движения, например при появлении препятствия на участке дороги, неблагоприятных метеоусловиях и т. п. Снижение скорости движения автомобилем - лидером группы требует торможения разной интенсивности последующих автомобилей, а затем и разгонов, что создает пульсирующий, неустойчивый поток.

Резкое торможение транспортного потока (находящегося в режиме, соответствующем точке А) и переход его в результате торможений к состоянию по скорости и плотности в положение, соответствующее, например, точке В, вызывает так называемую «ударную волну» (показана тонкой штриховой линией АВ), распространяющуюся навстречу направлению транспортного потока со скоростью, характеризуемой тангенсом угла β.

Еще одним важным параметром, характеризующим транспортный поток, является наличие в его составе транспортных средств различного типа - состав транспортного потока. Состав транспортного потока влияет на загрузку дорог (стесненность движения), что объясняется, прежде всего, существенной разницей в габаритных размерах автомобилей: от длины 4...5 м для легковых автомобилей, до длины 24 м, характерной для автопоездов. Необходимо учитывать и различие в динамических характеристиках автомобилей различного типа.

Оценка состава транспортного потока осуществляется по процентному составу или доле транспортных средств различных типов.

Знание состава транспортного потока позволяет учесть степень занятости дороги движущимся автомобилем, выражением которой является динамическая площадь автомобиля S_a. Этот показатель характеризует сложность обгона данного автомобиля другим.

Низкие значения S_a незначительно влияют на безопасность движения, а высокие вызывают ее снижение. Динамическая площадь определяется как произведение длины автомобиля l_a на динамический габарит G_a:

$$S_a = G_a * l_a$$

$$G_a = (B_a + K_a + 0{,}5)/2$$

Где B_a - ширина автомобиля, м; K_a - ширина колеи (по центрам колес), м; 0,5 - запас для обеспечения разъезда по 0,25 м на сторону.

Для учета в фактическом составе транспортного потока влияния различных типов транспортных средств на загрузку дороги применяют коэффициенты приведения $K_{\text{пр}\,i}$ к условному легковому автомобилю.

В действительности грузоподъемность не имеет прямого отношения к динамике автомобиля. Этот вывод тем более справедлив для современных более динамичных, скоростных транспортных средств. Автомобиль с высокими динамическими характеристиками независимо от грузоподъемности создает меньше помех движению и способствует повышению пропускной способности дороги.

Корректная оценка динамики автомобилей в транспортном потоке позволяет повысить достоверность расчета пропускной способности дороги.

3 Проектирование веб-приложения

3.1 Функциональные требования к системе

Программный комплекс планируется использовать для создания динамических транспортных моделей с возможностью прогнозирования характера транспортного потока в зависимости от текущей транспортной ситуации: возникновение ДТП, ремонт дороги и т.д. в режиме реального времени.

Данное приложение использует один актор – администратор системы. Его возможности представлены на Use-case диаграмме приложения (рисунок 3.1)

Рисунок 3.1 – Use-case диаграмма WEB-приложения

Пользователь системы имеет возможность управлять параметрами генерации потока автомобилей. Сюда входят:

– Редактирование общего числа автомобилей в городе

– Минимальная, средняя и максимальная скорости движения автомобилей во время работы программы

Также есть возможность введения дополнительных параметров движения автомобилей, которые включают в себя резкое увеличение числа машин в городе, создание аварийных ситуаций и полное закрытие любого ребра графа дорожной сети.

Кроме того, администратор может редактировать маршруты следования автомобилей по карте. Данная функция включает в себя изменение графа дорожной сети (добавление и удаление рёбер графа).

Администратор имеет возможность взаимодействовать с базой данных приложения, загружая список имитационных моделей, выбирая конкретную, сохраняя, удаляя любую из них. Кроме того, изменяя любой элемент графа дорожной сети (вершина или ребро), пользователь в любой момент может сохранить внесенные им изменения.

Подробное описание возможностей пользователя системы представлены в таблице реестра вариантов использования (таблица 3.1).

Таблица 3.1 – Описание возможностей пользователя системы

Код	Основной актор	Наименование	Формулировка
A1	Администратор	Редактирование общего числа автомобилей в городе	Этот вариант использования позволяет администратору системы редактировать общее число автомобилей в городе, что повлияет на количество автомобилей, которые будут находиться на дороге в каждый момент времени суток
A2	Администратор	Изменение показателей скорости автомобилей	Этот вариант использования позволяет администратору системы изменить показатели минимальной средней и максимальной скорости движения автомобилей в городе, что повлияет на время их движения от точки старта до точки финиша их запланированного маршрута

Окончание таблицы 3.1

A3	Администратор	Введение доп. параметров движения автомобилей	Этот вариант использования позволяет администратору системы вводить дополнительные параметры движения автомобилей, которые включают в себя резкое увеличение числа машин в городе, создание аварийных ситуаций и полное закрытие любого ребра графа дорожной сети
A4	Администратор	Редактирование маршрута следования автомобилей	Этот вариант использования позволяет администратору системы редактировать маршруты следования автомобилей, позволяя добавлять новые ребра графа или удалять их
A5	Администратор	Сохранение графа дорожной модели	Этот вариант использования позволяет администратору системы сохранить созданную дорожную модель и использовать ее в последующих экспериментах
A6	Администратор	Удаление графа дорожной модели	Этот вариант использования позволяет администратору системы удалить из базы данных созданную дорожную модель
A7	Администратор	Загрузка и изменение графа дорожной модели	Этот вариант использования позволяет администратору системы загрузить из базы данных уже существующую дорожную модель и проводить необходимые эксперименты

Данная таблица описывает возможности действий, которые предоставляет система для администратора.

3.2 Модель данных

Все данные о созданных пользователями системы моделях хранятся в базе данных Firebase. Firebase – это облачный сервис, сочетающий в себе множество функций: аутентификацию, базу данных в реальном времени, хранение файлов, уведомления и прочие.

База данных состоит из трех таблиц: Nodes, Edges и Cars. Таблицы Nodes и Edges связаны между собой связью «один-ко-многим» (каждая вершина графа может принадлежать нескольким ребрам). Таблицы Cars и Edges связаны между собой связью «один-к-одному» (каждый автомобиль может одновременно находиться на одном ребре).

Таблица **Nodes** содержит 8 полей:

– **Node_id** – уникальный номер вершины графа

– **Label** – название вершины графа, которое отображается при наведении на него курсора мыши

– **X** – относительная координата вершины графа на оси абсцисс

– **Y** – относительная координата вершины графа на оси ординат

– **Latitude** – абсолютная координата широты вершины графа

– **Longitude** – абсолютная координата широты вершины графа

– **Size** – отображаемый размер вершины графа на экране пользователя

– **Color** - отображаемый цвет вершины графа на экране пользователя

Таблица **Edges** содержит 19 полей:

– **Edge_id** – уникальный номер ребра графа («автомобильной дороги»)

– **Source** - уникальный номер вершины графа, с которого начинается ребро графа

– **Target** - уникальный номер вершины графа, которым заканчивается ребро графа

– **Distance** – длина ребра графа для расчета времени движения по нему автомобилей

– **Color** – цвет ребра графа

– **Bandwidth** – пропускная способность ребра графа

– **Cars** – количество автомобилей на ребре графа в каждую минуту часа

– **Is_closed** – поле, показывающее, является ли ребро графа закрытым и недоступным для движения по нему автомобилей

– **Is_has_accidents** - поле, показывающее есть ли на ребре графа ДТП, что повлияет на скорость движения по нему автомобилей

– **Quality** – качество дорожного покрытия (влияет на скорость движения автомобилей)

– **Class -** класс автомобильной дороги (автобан, трасса федерального значения и т.д.)

– **Pavement_type** – тип дорожного покрытия автомобильной дороги (асфальтобетон, цементобетон и т.д.)

– **Load** – моментальная загруженность автомобильной дороги (рассчитывается из нескольких параметров: класс автомобильной дороги, тип дорожного покрытия, пропускная способность ребра, количество автомобилей на ребре)

– **Points** – «штрафные» баллы, которые ребро получает, если коэффициент его загруженности больше 90%

– **Reduction_factor** – коэффициент понижения пропускной способности ребра (зависит от типа дорожного покрытия, качества дорожного покрытия, дорожных ситуаций на ребре, количества полос движения, минимальной и максимальной скорости движения автомобилей)

– **Speed_reduction_factor -** коэффициент понижения скорости движения автомобиля на ребре (зависит от типа дорожного покрытия, качества дорожного покрытия, дорожных ситуаций на ребре)

– **Accident_coefficient_nerf** – коэффициент, показывающий на сколько процентов дорожные ситуации снижают коэффициент пропускной способности ребра

– **Speed_coefficient_nerf** – коэффициент, показывающий на сколько процентов дорожные ситуации снижают максимальную скорость автомобилей на ребре

– **Has_accidents** – параметр, показывающий есть ли на ребре дорожные ситуации, снижающие его пропускную способность и максимальную скорость автомобилей на нем

Таблица **Cars** содержит 5 полей:

– **Car_id** – уникальный номер автомобиля

– **Speed** – средняя скорость движения автомобиля

– **Route** – маршрут следования автомобиля, включающая в себя последовательность вершин графа

– **Number** – гос. номер машины («A000AA00»)

– **Is_onroad** – величина, показывающая находится ли машина на каком-либо графе в данный момент

Связь между таблицами представлена на рисунке 3.2.

Рисунок 3.2 – ER-диаграмма приложения

Данная структура БД позволяет легко динамически изменять поля таблиц во время процесса моделирования дорожного движения и обеспечивать необходимую легкость и доступность к данным БД при работе с сервером в JSON-формате.

3.3 Инструменты реализации

В качестве основного языка для разработки был выбран JavaScript - объектно-ориентированный язык программирования для написания сценариев. Web-браузер интерпретирует код сценария языка JavaScript и на основе описанных в сценарии действий производит манипуляции с разметкой web-страницы. Таким образом, посредством языка JavaScript реализуется возможность программирования на стороне клиента. Язык JavaScript предоставляет возможность доступа к элементам разметки web-страницы посредством объектов.

Вся полученная информация выводится в понятном пользователю виде на Web-страницах. Web-страницы описываются на специальном языке, называемом HTML (HyperText Markup Language язык разметки гипертекстовой информации), который позволяет осуществлять простое форматирование текста, вставку графики, а также составление таблиц и цветовое оформление документов.

В качестве серверной части приложения будет использоваться стек технологий (Linux + Nginx + Firebase)

Для хранения всех данных проекта была выбрана база данных Firebase. Основной сервис — облачная СУБД класса NoSQL, позволяющая приложений хранить и синхронизировать данные между несколькими клиентами. Поддержаны особенности интеграции с приложениями под операционные системы, предусмотрено API для шифрования данных.

Операционная система Linux для сервера обслуживает контент для клиентских устройств. Соответственно, операционные системы для серверов имеют инструментарий для создания простых серверов. Поскольку серверы, в основном, работают в автоматическом режиме, то графический интерфейс с пользователем (GUI) в операционной системе Linux для сервера имеет менее важную роль.

Firebase служит базой данных, которая изменяется в реальном времени и хранит данные в JSON. Любые изменения в базе данных тут же синхронизируются между всеми клиентами, или девайсами, которые используют одну и ту же базу данных. Другими словами, обновление в Firebase происходят мгновенно.

Вместе с хранилищем, Firebase также предоставляет пользовательскую аутентификацию, и поэтому все данные передаются через защищенное соединение SSL.

Nginx — мощный инструмент для развертывания веб-сервера, который при правильной настройке превосходит Apache. Области применения Nginx весьма обширны — от кэширования HTTP до создания инвертированного прокси-сервера. Nginx работает на ОС Unix-типа и был успешно протестирован на OpenBSD, FreeBSD, Linux, Mac OS X, Solaris.

В отличие от обычного веб-сервера, Nginx не создаёт один поток под каждый запрос, а разделяет его на меньшие однотипные структуры, называемые рабочими соединениями. Каждое такое соединение обрабатывается отдельным рабочим процессом, а после выполнения они сливаются в единый блок возвращающий результат в основной процесс обработки данных. Одно рабочее соединение может обрабатывать до 1024 запросов одного вида одновременно.

3.4 Поведение системы

Транспортные потоки складываются из отдельных передвижений, совершаемых участниками движения, или пользователями транспортной сети. Основными факторами, определяющими количество совершаемых передвижений и их распределение по транспортной сети города, являются:

Потокообразующие факторы, т.е. размещение объектов, порождающих передвижения, таких как места проживания, места приложения труда, культурно-бытового обслуживания и др.

Характеристики транспортной сети, такие как количество и качество улиц и дорог, параметры организации движения, маршруты и провозные способности общественного транспорта и др.

Поведенческие факторы, такие как мобильность населения, предпочтения при выборе способов и маршрутов передвижений и др.

Для построения математических моделей необходимо формальное описание указанных факторов. Основа такого описания — транспортный граф, узлы которого соответствуют перекресткам и станциям внеуличного транспорта, дуги — сегментам улиц и линий внеуличного транспорта. Отдельной составляющей транспортного графа является маршрутный граф общественного транспорта. Узлами маршрутного графа являются остановочные пункты, дугами — сегменты маршрутов между остановочными пунктами. С обычными узлами графа узлы-остановки соединены дугами-посадками и дугами-высадками.

Для описания распределения потокообразующих объектов необходимо разделить город на некоторое количество условных районов прибытия и отправления (ПО). Каждый район ПО включается в граф как узел, соединенный с обычными узлами графа специальными дугами-связями. Общий объем передвижений из одного района ПО в другой (независимо от конкретных путей передвижения) называется межрайонной корреспонденцией. Основой для моделирования поведения пользователей является математическая

формулировка критерия, на основании которого пользователь оценивает альтернативные пути и способы передвижения.

Данный критерий принято называть обобщенной ценой пути. Увеличение обобщенной цены снижает привлекательность пути. Обобщенная цена пути складывается из обобщенных цен, входящих в него дуг. Кроме того, в цену пути может добавляться цена переходов с дуги на дугу, например, цена поворота при движении по улично-дорожной сети (УДС) или цена посадки при переходе с дуги-пересадки на дугу, соответствующую поездке. Обобщенная цена определяется как взвешенная сумма слагаемых, выражающих влияние факторов различной природы на оценку пути. В общем случае она может включать в себя следующие слагаемые: время передвижения, которое вычисляется на основе заданной функции зависимости скорости движения от загрузки дуги.

Используются различные функции скорости для дорог с разными физическими характеристиками и условиями регулировки движения; дополнительные задержки на различных элементах транспортной сети (время парковки, время ожидания и др.); денежные затраты (платные магистрали, плата за въезд в определенные зоны города и др.); условные штрафные добавки времени, используемые для моделирования различных особенностей транспортной сети и мер по управлению транспортом. Как показывают обследования, время — основной фактор, определяющий цену пути. Другие факторы являются корректирующими и количественно выражаются в условных минутах, добавляемых к времени передвижения.

Поэтому путь между двумя точками сети, имеющий минимальную обобщенную цену среди всех возможных путей, часто для простоты называют кратчайшим путем. Важнейшей и фундаментальной особенностью формирования загрузки транспортной сети является то, что выбор способов и путей передвижения пользователями сети влияет на тот же выбор, осуществляемый другими пользователями. Математически это взаимное влияние описывается функциями зависимости цены дуги от суммарного потока

по этой дуге. Данное обстоятельство создает обратную связь в процессе формирования загрузки: выбор путей, формирующих загрузку, основан на сопоставлении цен различных путей, в то время как цены сами определяются сложившейся загрузкой. Транспортные потоки, реально наблюдающиеся в сети, представляют собой некоторое равновесное состояние этого процесса. Для поиска этого равновесного состояния применяются итеративные алгоритмы, описанные ниже. В задаче моделирования транспортных потоков в сети крупного города традиционно выделяют четыре основных этапа:

– оценка общих объемов прибытия и отправления из каждого района города (Trip generation);

– расщепление по способам передвижений, таким как пешие передвижения, передвижения с использованием общественного транспорта, передвижения на личном автомобиле и др. (Modal split);

– определение матриц корреспонденций, определяющих объем передвижений между каждой парой расчетных районов города (Trip distribution);

– распределение корреспонденций по транспортной сети, т.е. определение всех путей, выбираемых участниками движения, и определение количества передвижений по каждому пути (Trip assignment).

Разделение задачи моделирования на эти четыре этапа является условным, так как все этапы взаимосвязаны и не могут, вообще говоря, быть решены как отдельные задачи в силу отмеченных выше обратных.

Так, большинство моделей расчета корреспонденций используют в качестве важного фактора обобщенные цены межрайонных передвижений. Аналогично, расщепление передвижений по видам (например, между частным и общественным транспортом) зависит от соотношения цен при использовании этих видов транспорта. Следовательно, расчет корреспонденций и их расщепление может быть выполнено корректно, если уже известна итоговая загрузка сети.

Алгоритм реализации создания объекта «автомобиль» представлен на рисунке 3.3.

Рисунок 3.3 – алгоритм реализации создания объекта.

Описание шагов алгоритма создания объекта «автомобиль»

– Получение запроса на создание нового объекта «автомобиль»

– Проверка наличия на карте соединенных ребер, которые в будущем образуют маршрут движения нового объекта

– Если соединенные дороги существуют, происходит создание нового объекта с начальными параметрами

a. Средняя скорость движения: 60 км/ч

b. Длина автомобиля: 2,5 м

c. Находится ли автомобиль на дороге: нет

d. Гос. номер автомобиля: <<null>>

– Создание индивидуального гос. номера автомобиля вида «A000AA000RUS” выполняется автоматически, генерируясь системой

– Создание маршрута следования для нового объекта

a. Случайная генерация точки старта автомобиля

b. Случайная генерация точки финиша автомобиля

c. Проверка на несовпадения точек финиша и точек старта автомобиля

d. Построение маршрута следования от точки старта до точки финиша

– Динамическое задание параметров автомобиля с некоторой вероятностью отклонения от начальных значений

– Запуск автомобиля по заданному маршруту, изменение его свойста «Находится ли автомобиль на дороге» на «да»

– После достижения точки финиша свойство «Находится ли автомобиль на дороге» меняется на «нет»

– Автомобиль снова доступен для постановки на новый маршрут с той точки, которая являлась конечной для предыдущего маршрута

Диаграмма последовательности для общего алгоритма работы программы приведена на рисунке 3.4.

Рисунок 3.4 – Диаграмма последовательности

При работе с системой пользователь создает запрос на создание вершин и ребер графа, которые записываются в БД приложения. После чего идет передача данных в SigmaJS, который занимается отрисовкой созданных вершин и ребер графа, совмещая их с картой реального города. После начала процесса генерации автомобилей и запуска, идет процесс передачи данных о вершинах графа в объект автомобиля.

После чего динамически создаются автомобили с заданными параметрами с некоторой вероятностью их появления и отклонения от начальных параметров. Передача данных об этих изменениях и текущем состоянии каждого автомобиля передается в систему и моментально отрисовывается на экране пользователя с помощью SigmaJS.

Диаграмма классов приложения представлена на рисунке 2.5. Она демонстрирует общую структуру иерархии классов системы, обеспечивающих ее быстродействие и удобство работы пользователя.

Рисунок 3.5 – Диаграмма классов

Система состоит из 9 основных классов и 3 вспомогательных (описаны в ER-диаграмме):

– **AppMainModule** – класс, описывающий главный модуль системы. Атрибуты класса:

App_graph – атрибут, содержащий информацию о построенном графе (вершины, ребра, маршруты)

World_hours – атрибут, содержащий информацию о текущем «времени суток» системы (часы)

World_minutes - атрибут, содержащий информацию о текущем «времени суток» системы (минуты)

World_cars - атрибут, содержащий информацию о всех сгенерированных машинах в «городе» (системе)

Cars_on_road - атрибут, содержащий информацию об автомобилях, в данный момент времени находящихся на «дороге» (ребре)

Is_app_running - атрибут, содержащий информацию о запуске процесса моделирования

Методы класса:

choseCarsToStart() – метод, случайно выбирающий из доступного количества автомобилей те, которые в следующий момент времени «выедут в город»

runApp() – метод, запускающий процесс моделирования транспортного потока

runEachChosenCar() – метод, запускающий автомобиль на ранее сгенерированный методом generateRoadTrip() маршрут

isCarOnRoadNow() – метод, в каждый тик проверяющий массив сгенерированных автомобилей и выделяющий из них те, которые в данный момент находятся на «дороге».

checkEngine() – метод, срабатывающий до метода runApp() и проверяющий условия для возможности запуска процесса моделирования (соединение всех вершин ребрами, наличие сгенерированных маршрутов и некоторого количества автомобилей, отличного от 0)

startDay() – метод, запускающий смену «времени суток». «Время суток» приложения протекает минимум в 1000 раз быстрее (24 минуты реального времени (зависит от настроек пользователя)).

– **UserActionsHandlerModule** – класс, описывающий модуль взаимодействия пользователей с системой. Данный класс не содержит атрибутов.

Методы класса:

startApp – метод, позволяющий пользователю запустить процесс имитационного моделирования

pauseApp – метод, позволяющий пользователю приостановить процесс имитационного моделирования с возможностью его последующего продолжения.

stopApp – метод, позволяющий пользователю остановить процесс имитационного моделирования без возможности его последующего продолжения (при повторном запуске процесс моделирования начнется заново).

selectModel – метод, позволяющий пользователю выбрать желаемую модель для работы (только созданную или загруженную из БД)

deleteModel – метод, позволяющий пользователю удалить выбранную модель без возможности последующего восстановления

createModel – метод, позволяющий пользователю создать новую модель с выбранными начальными настройками

showNotification – метод, позволяющий пользователю видеть на экране уведомление об успешном выполнении некоторого действия системы (создание модели, удаление частей графа и т.д.)

changeEdgeParameters – метод, позволяющий пользователю системы взаимодействовать с рёбрами графа (добавление, удаление, изменение свойств)

changeNodeParameters – метод, позволяющий пользователю системы взаимодействовать с вершинами графа (добавление, удаление)

mapContainerOperations – метод, позволяющий пользователю взаимодействовать с генерируемой картой (изменение координат, масшабирование)

– **GenerateRecommendatiosModule** – класс, описывающий модуль создания рекомендаций по улучшению пропускной способности проблемных участков автомобильных дорог.

Атрибуты класса:

App_edges – атрибут, содержащий информацию о всех ребрах графа дорожной сети

Методы класса:

fetchEdges – метод, отвечающий за сбор информации о каждом ребре графа каждый момент времени

checkEdgePoints – метод, проверяющий количество «штрафных баллов» у каждого ребра графа (после получения их критического количества ребро помечается как «рекомендованное к улучшению пропускной способности»)

generateVerdict – метод, генерирующий советы по улучшению пропускной способности каждого «рекомендованного» ребра на основе его свойств (пропускная способность, количество полос и т.д.)

generateVerdictCard – метод, выводящий на экран пользователя все советы по улучшению пропускной способности дорожной сети, сгенерированной методом generateVerdict

updateVerdict – метод, обновляющий информацию о пропускной способности рёбер графа в зависимости от действий пользователя над ними

– **MapActionsModule** – класс системы описывающий модуль работы карты (Stamen Maps) и графовой модели (Sigma JS). Атрибуты класса:

Map_container – атрибут, содержащий информацию о текущих координатах центра карты (необходимо для корректного отображения графа дорожной сети при любом масштабировании)

Graph_container – атрибут, содержащий информацию о созданном пользователем графе дорожной сети

Методы класса:

renderMap – метод, отрисовывающий карту в момент после загрузки выбранной пользователем имитационной модели

renderGraph - метод, отрисовывающий граф дорожный сети в момент после загрузки выбранной пользователем имитационной модели

redrawMap – метод, перерисовывающий карту выбранной местности, в случае если пользователь изменил координаты ее центра

redrawGraph – метод, каждый момент времени перерисовывающий граф дорожной сети, в случае если он был изменен пользователем (добавление или удаление вершин или рёбер) или системой (во время процесса моделирования)

– **RoadTrafficGenerationModule** – класс системы, описывающий модуль генерации дорожного трафика с заданными пользователем параметрами. Атрибуты класса:

Car_object – атрибут, содержащий информацию о каждом созданном объекте «автомобиль»

Map_edges – атрибут, содержащий информацию о всех ребрах графа дорожной сети

Map_nodes – атрибут, содержащий информацию о всех вершинах графа дорожной сети

Graph_directions – атрибут, содержащий информацию о всех сгенерированных маршрутах следования автомобилей графа дорожной сети

Методы класса:

getCarPreviousNode – метод, получающий информацию о предыдущей вершине, которую пересек каждый автомобиль (необходима для корректного расчета маршрута его следования)

getCarNextNode - метод, получающий информацию о следующей вершине, которую пересек каждый автомобиль (необходима для корректного расчета маршрута его следования)

generateNewCarObject – метод, генерирующий объект «автомобиль» со случайно заданными параметрами

generateRoadTrip – метод, генерирующий маршрут для каждого автомобиля (состоит из вершин, которые автомобиль должен пересечь)

refreshEdgeParameters – метод, обновляющий параметры ребра графа (загруженность, цвет) во время процесса моделирования

redrawEdge – метод, обновляющий параметры ребра графа (цвет, размер), которые визуально отображаются на экране пользователя

createNewCarRoute – метод, генерирующий новый маршрут для автомобиля, в тот момент, когда он достигает точки финиша предыдущего маршрута

– **DataBaseActionsModule** – класс, описывающий модуль взаимодействия системы с базой данных. Атрибуты класса:

User_data - атрибут, содержащий информацию о данных, которые необходимы для сохранения в БД (вершины и ребра графа, сгенерированные автомобили)

Методы класса:

addData – метод, позволяющий сохранить необходимую информацию в базу данных

removeData – метод, позволяющий удалить необходимую информацию из базы данных

changeData – метод, позволяющий изменить необходимую информацию в базе данных

getData – метод, позволяющий получить необходимую информацию из базы данных

– **DefaultSettingsModule** – класс, описывающий модуль, который отвечает за хранение в приложение начальных настроек. Атрибуты класса:

Is_app_running – атрибут, содержащий информацию о запуске процесса моделирования

World_hours - атрибут, содержащий информацию о «времени суток» в системе во время процесса моделирования (часы)

World_minutes - атрибут, содержащий информацию о «времени суток» в системе во время процесса моделирования (минуты)

Nodes_number - атрибут, содержащий информацию о количестве вершин графа

Minutes_in_real - атрибут, содержащий информацию о том, за сколько минут пройдет цикл «суток» моделирования

Road_coefficients - атрибут, содержащий информацию о количестве машин на дороге в каждый «час суток» процесса моделирования

Cars_in_city - атрибут, содержащий информацию об общем количестве автомобилей в городе

Edge_refresh_interval - атрибут, содержащий информацию об интервале обновления параметров рёбер графа во время процесса моделирования

Canvas_array_interval - атрибут, содержащий информацию об интервале обновления статистики рёбер графа (средняя загруженность, количество машин, пропускная способность) во время процесса моделирования

Road_accidents - атрибут, содержащий информацию обо всех возможных дорожных ситуациях, которые пользователь может установить для каждого ребра

Quality_types - атрибут, содержащий информацию о качестве дорожного покрытия, которые пользователь может установить для каждого ребра

Pavement_types - атрибут, содержащий информацию о типе дорожного покрытия, которые пользователь может установить для каждого ребра

Road_classes - атрибут, содержащий информацию о классе автомобильной дороги, которые пользователь может установить для каждого ребра

– **DrawChartsModule** – класс, описывающий модуль, который отвечает за отрисовку статистики рёбер графа. Атрибуты класса:

Graph_labels - атрибут, содержащий информацию о точках на оси X («часы и минуты суток»)

Radar_chart_data - атрибут, содержащий информацию о данных для отрисовки допустимой и фактической загруженностях ребра графа

Edge_chart_data - атрибут, содержащий информацию о данных для отрисовки суточной интенсивности движения автомобилей на каждом ребре графа

Bar_chart_data - атрибут, содержащий информацию о данных для отрисовки процента времени, когда ребро было фактически перегружено

Statistics_chart_data - атрибут, содержащий информацию о данных для отрисовки статистики всех ребер графа

Методы класса:

drawEdgeRadarChart – метод, позволяющий вывести на экран пользователя информацию о суточной интенсивности движения автомобилей на выбранном ребре графа

updateEdgeRadarChart – метод, позволяющий в реальном времени обновлять информацию о суточной интенсивности движения автомобилей на выбранном ребре графа

drawEdgeBarChart – метод, позволяющий вывести на экран пользователя информацию о проценте времени, когда выбранное ребро было фактически перегружено

updateEdgeBarChart – метод, позволяющий в реальном времени обновлять информацию о проценте времени, когда выбранное ребро было фактически перегружено

drawStatisticsChart – метод, позволяющий вывести на экран пользователя информацию о загруженности всех ребер графа одновременно

updateStatisticsChart – метод, позволяющий в реальном времени обновлять информацию о загруженности всех ребер графа одновременно

– **AuxiliaryModule** - класс, описывающий модуль, который отвечает за вспомогательные функции во время процесса моделирования. Данный класс не имеет атрибутов.

Методы класса:

degreesToRadians - метод, позволяющий перевести градусы в радианы (для получения X и Y координат вершин)

distanceInKmBetweenEarthCoordinates – метод, позволяющий рассчитать расстояние между двумя вершинами на карте и найти длину ребра, которое их соединяет

dijkstraAlgorithm – алгоритм, позволяющий найти минимальное расстояние между двумя точками маршрута следования автомобилей

getRandomNumber – метод, позволяющий сгенерировать случайное число в выбранном диапазоне

getRandomInteger - метод, позволяющий сгенерировать случайное целое число в выбранном диапазоне

carNumberGenerator - метод, позволяющий сгенерировать случайный гос. номер автомобиля (необходимо для его однозначной идентификации)

3.5 Разработка прототипа

Обычно под прототипом понимают поведенческую модель, в которой не реализуются все слои архитектуры системы, но которая обладает предполагаемым интерфейсом пользователя. Такой прототип называется горизонтальным, или поведенческим.

Горизонтальный прототип не реализует функциональности системы в действительности, он создает только ее видимость, поэтому трудоемкость

разработки горизонтального прототипа невелика. Экраны интерфейса пользователя, навигация между ними показывают функциональные возможности и структуру доступа к информации, что позволяет пользователю исследовать поведение системы в различных ситуациях, выяснить возможность выполнения необходимой работы и уточнить требования.

Вертикальный, или структурный прототип служит для проверки концепции, реализует часть функциональности системы на всех уровнях от интерфейса пользователя до сервисных функций. Такой прототип действует как настоящая система и позволяет оптимизировать алгоритмы, оценить базу данных, определить критические временные требования. Вертикальный прототип полезен для исследования критически важных требований к интерфейсу и времени исполнения, для сокращения рисков на этапе проектирования системы.

Точкой входа «А» в приложение является начальный экран, который пользователь видит при загрузке веб-приложения. Дизайн этого экрана представлен на рисунке 3.6.

Рисунок 3.6 – Точка входа в приложение

Начальная страница приложения состоит из двух элементов – его названия и кнопки «Начать работу», которая призывает пользователя начать работу по созданию имитационной модели (рисунок 3.7).

Рисунок 3.7 – Дизайн начального экрана приложения

После нажатия на кнопку пользователь попадает на экран выбора имитационной модели, с которыми можно проводить манипуляции. Кроме того, есть карточка, которая позволяет создать новую имитационную модель (рисунок 3.8).

Рисунок 3.8 – Карточка создания новой модели

Кнопка «Создать» является недоступной до тех пор, пока пользователь в поле ввода «Название модели» не введет фразу длиннее трех символов на любом

языке и в любом регистре. После чего кнопка станет доступной для нажатия (рисунок 3.9).

Рисунок 3.9 – Активная кнопка «Создать» на карточке создания новой модели

По нажатии на активную кнопку «Создать» открывается модальное окно, которое содержит необходимые настройки для нормального функционирования имитационной модели - Режим распределения интенсивности движения автомобилей по дороге в течение суток, коэффициенты усредненных характеристик движения и график, который отражает суточную интенсивность движения автомобилей (рисунок 3.10). Кроме того, в нижней части модального окна присутствуют две кнопки: «Отмена» и «Сохранить».

Рисунок 3.10 – Модальное окно для задания режима интенсивности движения автомобилей на дорогах

Вся информация приложения обрабатывается и сохраняется на сервер, на котором используется база данных Firebase. Логика взаимодействия приложения и сервера с БД представлена на рисунке 3.11.

Рисунок 3.11 - Логика взаимодействия приложения и Firebase

После нажатия на кнопку «Сохранить модель» начальные данные имитационной модели отправляются на хранение в БД, после чего от сервера приходит ответ и на экране приложения появляется карточка с только что созданной моделью и её параметрами (рисунок 3.12).

Рисунок 3.12 – карточка имитационной модели

Каждая карточка имитационной модели на экране (только созданной или уже использовавшейся) содержит следующие элементы:

– Название модели (например «test»)

– Символ крестика, по нажатию на который пользователю будет предложено либо удалить текущую модель, либо отменить это действие

– Параметры имитационной модели: количество вершин, рёбер, маршрутов и режим распределения интенсивности движения автомобилей по дороге в течение суток.

– Кнопка «Выбрать», по нажатию на которую пользователь переходит на главный экран приложения (рисунок 3.13)

Рисунок 3.13 – Основной экран приложения

Основной экран приложения содержит два элемента – карту города, для которого будет создаваться симуляция движения и «гамбургер», который открывает боковое меню для взаимодействия пользователя с системой (рисунок 3.14).

Рисунок 3.14 – Открытие окна взаимодействия пользователя с системой

Боковое окно содержит название модели (например, «Модель №2»), выделенную жирным шрифтом и информацию о режиме распределения интенсивности движения автомобилей по дороге в течение суток (например, «Бимодальный»).

Ниже расположены вкладки («Симуляция», «Общая информация», «Вершины», «Рёбра», «Отчеты»), названия которых помогают пользователю понять, с каким элементом системы они будут работать, нажав на неё.

При нажатии на название любой вкладки, меняется динамически меняется содержимое страницы (рисунок 3.15). Активная вкладка подсвечивается отличным от других вкладок цветом.

Рисунок 3.15 – Динамическая смена активных вкладок

Первая вкладка «Симуляция» является основной и содержит следующие элементы (рисунок 3.16):

– Кнопка «Начать симуляцию»

– Полем для ввода максимального количества машин, которые появятся на дорогах города в период пиковой нагрузки

– Чекбокс, который отвечает за возможность изменения графа автомобильных дорог

– Информацию о «времени суток» в приложении

– Поле выбора желаемого «времени суток» для просмотра интенсивности автомобильного движения в определенный час

– Информация о количестве машин на дорогах города в каждый момент времени

Рисунок 3.16 – Содержимое вкладки «Симуляция»

Вторая вкладка «Общая информация» содержит следующие элементы (рисунок 3.17):

– Информацию об имитационной модели: количество вершин, ребер и сгенерированных маршрутов

– График, отображающий теоретическое изменение интенсивности движения автомобилей

Рисунок 3.17 – Содержимое вкладки «Симуляция»

Третья вкладка «Вершины» содержит следующие элементы (рисунок 3.18):

– Таблицу, отражающую информацию о всех вершинах графа, которые существуют на карте на данный момент

Рисунок 3.18 – Содержимое вкладки «Вершины»

Пользователь имеет возможность удалить каждую вершину, нажав на соответствующую кнопку в строке таблицы.

Третья вкладка «Вершины» содержит следующие элементы (рисунок 3.19):

– Поля выбора точки начала и конца ребра графа

– Кнопку добавления ребра

– Таблицу, отражающую информацию о всех ребрах графа, которые существуют на карте на данный момент

Рисунок 3.19 – Содержимое вкладки «Вершины»

ID каждого ребра является кнопкой, по нажатию на которую открывается модальное окно, дающее возможность просмотреть подробную информацию выбранного ребра (рисунок 3.20).

Рисунок 3.20 – Открытие окна подробной информации о ребре графа.

Данное окно содержит следующую полезную информацию о ребре:

– Класс автомобильной дороги
– Тип покрытия дорожного полотна
– Качество дорожного полотна
– Длина ребра
– Количество полос движения
– Минимальная и максимальная скорости движения автомобилей
– Загруженность ребра в реальном времени
– График изменения суточной активности движения автомобилей
– Графики, показывающие соотношения между пропускной способностью ребра и максимальной его загруженности в каждый час суток
– График, отражающий процент времени по часам, когда ребро было перегружено

Кроме того, некоторые параметры ребра также можно изменить, нажав на значок карандаша в предпоследней колонке. При этом появится модальное окно, с которым пользователь может взаимодействовать (рисунок 3.21).

Рисунок 3.21 - Открытие окна изменения параметров ребра графа.

Данное окно позволяет пользователю изменить следующие параметры ребра графа:

– Класс автомобильной дороги
– Тип дорожного покрытия
– Качество дорожного покрытия

Кроме того, есть возможность добавить (происшествие) на ребро (авария, ремонт дороги или препятствие (поломанный автомобиль)), которое повлечет за собой изменения остальных параметров ребра (количество полос движения, пропускную способность, минимальную и максимальную скорости движения автомобилей на нём).

Пятая и последняя вкладка «Отчеты» содержит следующие элементы (рисунок 3.22):

– Кнопку «Посмотреть подробную статистику»
– Кнопку «Сохранить модель»
– Кнопку «Сгенерировать маршруты»

Рисунок 3.22 – Содержимое вкладки «Отчеты»

При нажатии на кнопку «Посмотреть подробную статистику», перед пользователем открывается окно, на котором он может увидеть информацию о количестве автомобилей, проехавших по каждому ребру графа за каждый час моделирования дорожной ситуации.

При нажатии на кнопку «Сгенерировать маршруты», система генерирует все возможные маршруты движения автомобилей по всем существующим рёбрам графа.

При нажатии на кнопку «Сохранить модель», сохраняются все изменения, сделанные пользователям во время работы с параметрами имитационной модели.

Кроме того, при каждом важном действии пользователя система оповещает его о совершенном действии и изменениях, которые произошли после этого (рисунок 3.23).

Рисунок 3.23 – Уведомления о действиях пользователя

4 Проектный этап разработки системы

Разработанная система «Имитационное моделирование транспортных потоков» состоит из восьми модулей:

– Модуль работы с картами
– Модуль работы с базой данных
– Вспомогательный модуль для проведения необходимых расчетов во время этапа подготовки к моделированию
– Модуль работы с моделированием движения городского трафика
– Модуль обработки действий пользователя
– Модуль, содержащий начальные настройки и параметры системы
– Модуль работы с визуализацией данных
– Модуль генерации рекомендаций по улучшению структуры УДС

4.1 Модуль работы с картами.

Один из самых важных модулей, реализация которого необходима для моделирования дорожной сети - картографический материал; обычно это карты крупного масштаба, так как они позволяют с большой точностью делать замеры расстояний между пунктами. В качестве картографического материала были выбраны карты студии Stamen. Для реализации интерактивного поведения веб-карт был использован стек библиотек: Leaflet.js + Sigma.js.

Stamen maps активно использует данные OpenStreetMap во многих своих визуализациях карт. Stamen предоставляет два стиля OpenStreetMap плиток: Toner и Terrain. Для работы был выбран стиль Toner.

Существенным достоинством Stamen является их стабильность, безотказность, простота подключения и работы, свободное использование тайлов (плиток) их карт, но ограничения могут применяться для активных пользователей.

Недостатком является нечастое обновление серверов OpenStreetMap, а, следовательно, и Stamen maps, что может сказаться на актуальности предоставленной информации.

Leaflet.js – это библиотека с открытым исходным кодом, с помощью которой мы можем развертывать простые, интерактивные и легкие веб-карты.

Библиотека Leaflet позволяет использовать маркеры, всплывающие окна, векторные слои, наложения изображений и GeoJSON.

Есть возможность взаимодействовать с картами Leaflet, просто перетаскивая её, увеличивая масштаб (двойным щелчком или прокручивая колесико), используя клавиатуру или обработку событий компьютерной мыши.

Leaflet поддерживает все современные десктопные и мобильные браузеры.

Для реализации связки библиотек Stamen и Leaflet был использован следующий алгоритм действий:

– Создание базовой HTML-разметки с необходимыми тегами.

– Загрузка CSS и JS скриптов Leaflet в проект для полного использования их возможностей.

– Создание контейнера для отображения пользователю системы выбранной карты.

– Задание необходимых параметров карты. Leaflet представляет широкие возможности гибкой настройки своих параметров, в том числе «тип управления картой», «параметры состояния карты», «параметры анимации». Для задания необходимых параметров был создан объект **«mapOptions»** и установлено два необходимых для функционирования карты параметра: **«center»** - опция, значение которой указывает место, в котором при начальной загрузке будет находиться центр карты, и **«zoom»** - опция, значение которой представляет собой уровень масштабирования карты по умолчанию.

– Создание объекта карты на странице веб-приложения. Используя API-интерфейс Leaflet, был создан объект карты путем создания экземпляра

класса карты Leaflet. При создании экземпляра класса необходимо передать два параметра: контейнер для хранения карты и опции с параметрами карты.

– Библиотека Stamen позволяет загружать и отображать разные типы карт (слоёв), создавая экземпляр класса TileLayer. При его создании необходимо указать шаблон URL-запроса, запрашивающий желаемый слой карты в виде строки. После чего необходимо добавить слой, созданный на шаге 6, к объекту карты, используя метод addLayer().

В результате выполнения шагов алгоритма при запуске приложения получаем вывод карты, представленный на рисунке 4.1.

Рисунок 4.1 – Вывод карты с использованием библиотек Leaflet и Stamen

Для отображения графа дорожной сети на карте была использована библиотека SigmaJS, которая может использовать одну из технологий для отображения вашего графика: svg, canvas и webgl. В приложении был выбран Canvas - элемент HTML5, совместимый со всеми современными браузерами. Он определяет область на веб-странице, где пользователь имеет возможность нарисовать несколько пикселей с помощью JavaScript API. Код JavaScript непосредственно интерпретируется браузером, и он решает, что и в какой момент времени нужно визуализировать.

Для работы с Sigma так же необходимо подключить CSS и JS файлы в проект. В результате получился следующий код:

```
sigmaInstance = new sigma({
    graph: g,
    container: 'graph-container',
    settings: {
        // style the nodes to spot them easily:
        nodeBorderSize: 1,
        defaultNodeBorderColor: '#fff',
        nodeOuterBorderSize: 2,
        minNodeSize: 1,
        maxNodeSize: 10,
        minEdgeSize: 1,
        maxEdgeSize: 5,
        autoRescale: true
    }
});
```

Экземпляр класса SigmaJS содержит три обязательных параметра: объект, содержащий информацию о построенном графе (вершины и рёбра), контейнер (HTML-элемент, на котором будет отображаться граф) и настройки по умолчанию (минимальный и максимальный размеры вершин и рёбер, параметры масштабирования).

После подготовки SigmaJS возникла необходимость добавить пользователю возможность редактировать граф: добавлять и удалять вершины и ребра, генерировать маршруты.

Объект новой вершины графа выглядит следующим образом:

```
let newNode = {
    id: 'n' + nodesNumber,
    label: 'node' + nodesNumber,
    x: Math.random(),
    y: Math.random(),
    lat: e.latlng.lat, // mandatory field
    lng: e.latlng.lng, // mandatory field
    size: 5,
    color: '#000'
};
```

Новая вершина содержит пять обязательных полей (уникальный номер, метка, x-координата, y-координата, широта, долгота) и два дополнительных (размер и цвет). При добавлении пользователем вершины происходит сразу несколько действий:

– Объект добавляется в массив вершин графа имитационной модели

– На вкладке «Вершины» добавится строка в таблице «Вершины графа»

```
btn.addEventListener("click", ev => {
    let newDirections = $.grep(g.directions, direction => {
        return !(direction.allNodes.includes(newNode.id));
    });
    showNotification('Удалено ' + (g.directions.length - newDirections.length) + '
маршрута(-ов)');
    g.directions = newDirections.slice();
    let newEdges = [];
    let newEdgesToDelete = [];
    newEdges = $.grep(g.edges, edge => {
        return !(edge.source === newNode.id || edge.target === newNode.id);
    });
    newEdgesToDelete = $.grep(g.edges, edge => {
        return (edge.source === newNode.id || edge.target === newNode.id);
    });
    showNotification('Удалено ' + (g.edges.length - newEdges.length) + ' ребра');
    g.edges = newEdges.slice();
    if (newEdgesToDelete.length) {
        newEdgesToDelete.forEach((value, index) => {
            document.getElementById('list-
edges').removeChild(document.getElementById('data-edge-id-' + value.id));
            sigmaInstance.graph.dropEdge(value.id);
        });
    }

    g.nodes = $.grep(g.nodes, node => {
        return (node.id !== newNode.id);
    });

    sigmaInstance.graph.dropNode(newNode.id);
    sigmaInstance.refresh();

    if (g.nodes.length !== 0) {
        leafletPlugin.fitBounds();
        sigma.plugins.relativeSize(sigmaInstance, 1);
    }
    tbody.removeChild(tr);
});
```

При создании вершины и добавлении строки в таблицу кнопке удаления присваивается сложная логика (при удалении вершины должны сначала должны удалиться маршруты, в которых присутствует эта вершина, после чего удаляются все ребра, началом или концом которых является выбранная вершина, и, наконец, удаляется сама вершина (рисунок 4.2))

Симуляция | Общая информация | Вершины | Рёбра | Отчеты

Вершины графа:

ID вершины	Название вершины	Удаление
n1	node1	×
n2	node2	×
n3	node3	×
n4	node4	×
n5	node5	×
n6	node6	×

Рисунок 4.2 – Содержимое вкладки «Вершины»

Объект нового ребра графа выглядит следующим образом:

```
let newEdge = {
    id: 'e' + g.edges.length,
    source: sNode.id,
    target: tNode.id,
    distance: distance,
    color: 'black',
    class: 6,
    pavementType: 1,
    quality: 2,//randomInteger(1, 5),
    load: 0,
    size: 0,
    cars: 0,
    reductionFactor: 1,
    speedReductionFactor: 1,
    accidentCoefficientNerf: 1,
    speedCoefficientNerf: 1,
    bandwidth: 1,
    hasAccidents: [
        {name: 'Авария', has: 0,},
        {name: 'Ремонт дороги', has: 0},
        {name: 'Препятствие (поломанный автомобиль)', has: 0}
    ]
}
```

Новое ребро графа имеет следующие обязательные поля:

– Уникальный идентификатор

– Точку начала

– Точку окончания

– Длина (расчёт расстояния между двумя точками на карте по координатам их широты и долготы)

– Цвет (изначально черный, во время процесса моделирования цвет ребра зависит от загруженности ребра)

– Класс ребра (класс автомобильной дороги (от «автобана» до «уличной дороги»))

– Тип дорожного покрытия (от асфальтобетона до грунта)

– Качество дорожного покрытия (от очень хорошего до очень плохого)

– Понижающий коэффициент пропускной способности (зависит от типа дорожного покрытия, качества дорожного покрытия и присутствия различных дорожных ситуаций)

– Понижающий коэффициент скорости (зависит от типа дорожного покрытия, качества дорожного покрытия и присутствия различных дорожных ситуаций)

– Пропускная способность (зависит от количества полос, типа дорожного покрытия, качества дорожного покрытия и присутствия различных дорожных ситуаций)

– Объект, которых хранит в себе информацию о ситуациях на дороге

ID каждого ребра является кнопкой, по нажатию на которую открывается модальное окно, дающее возможность просмотреть подробную информацию выбранного ребра. При создании ребра и добавлении строки в таблицу кнопке удаления присваивается сложная логика (при удалении ребра должны сначала должны удалиться маршруты, в которых присутствует это ребро, после чего удаляется само ребро).

После добавления или удаления вершины или ребра графа автоматически происходит обновление экземпляра класса SigmaJS и масштабирование нового графа (рисунок 4.3).

Рисунок 4.3 – Масштабирование карты

Пользователь имеет возможность сгенерировать маршруты следования автомобилей перед началом процесса моделирования. Для этого необходимо нажать кнопку «Сгенерировать маршруты» на вкладке «Отчеты» (рисунок 4.4).

Рисунок 4.4 – Возможность генерации маршрутов.

Основой для генерации маршрутов является алгоритм Дейкстры. Он в цикле перебирает все вершины графа и находит кратчайший маршрут до точки финиша, отличной от точки старта. Объект маршрута выглядит следующим образом:

```
let newDirection = {
    id: 'd' + g.directions.length,
    start: start.id,
    finish: finish.id,
    distance: distance,
    nodes: route,
    edges: [],
    allNodes: []
};
```

Маршрут содержит в себе информацию о расстоянии от точки старта до точки финиша, последовательности точек следования автомобиля, уникальных номерах ребер графа, составляющих его.

На этом этапе реализация модуля работы с картами была завершена. Программный код был готов к реализации модуля работы с моделированием движения городского трафика.

4.2 Модуль работы с базой данных

Для работы с графом дорожной сети, вершины и рёбра которого могут содержать до 15 параметров, каждый из которых может иметь собственную вложенность, встала необходимость использования нереляционной базы данных, одним из представителей которых является Firebase – это облачный сервис, сочетающий в себе множество функций: аутентификацию, базу данных в реальном времени, хранение файлов, уведомления и прочие.

Для подключения базы данных к веб-приложению необходимо было выполнить следующий алгоритм действий:

– Зарегистрировать приложение на сайте Google

– Получить ключ авторизации и параметры удаленного подключения к базе и инициализировать его в конце HTML-страницы приложения

После этого экземпляр класса firebase () будет доступен в приложении. Возможности БД были использованы для реализации следующих функций: загрузка всех существующих имитационных моделей, загрузка информации о выбранной пользователем имитационной модели, сохранение созданной модели,

сохранение параметров моделирования используемой модели, удаление выбранной модели из БД.

В качестве примера будет работы с БД будет рассмотрена реализация функции сохранения имитационной модели из базы.

```
firebase.database().ref('graphs/' + tempModelParameters.name).set({
    graph: JSON.stringify(g),
    modelParameters: JSON.stringify(tempModelParameters)
}).then(() => {
    showNotification('Модель ' + tempModelParameters.name + ' сохранена');
    console.log(g, tempModelParameters)
});
```

Для доступа к функциям Firebase необходимо перед каждой из них обращаться к методу database() экземпляра класса firebase. Так как каждый объект, хранящийся в базе данных Firebase, имеет собственную относительную ссылку, то при каждом обращении к нему первым параметром в функции ref () необходимо её указать. В данном случае ссылка выглядит следующим образом - 'graphs/' + tempModelParameters.name. Здесь «graphs» - название таблицы в базе данных, «tempModelParameters.name» - переменная, которая хранит в себе название сохраняемой модели. После чего в функции set () необходимо передать изменяемые параметры модели в виде объекта (ключ: значение). Результат изменения параметров сохраняемой модели представлен на рисунке 4.5.

Рисунок 4.5 – Результат вызова функции сохранения новых параметров модели

В случае успешного изменения параметров модели и их сохранения в базе данных, можно вызвать обработчик события, результат выполнения которого уведомит пользователя о завершении выполнении операции сохранения.

4.3 Вспомогательный модуль для проведения необходимых расчетов во время этапа подготовки к моделированию

Вспомогательный модуль полностью состоит из собственноручно написанных функций, которые используются в других модулях системы. В качестве основного языка для разработки был выбран JavaScript - объектно-ориентированный язык программирования для написания сценариев. Web-браузер интерпретирует код сценария языка JavaScript и на основе описанных в сценарии действий производит манипуляции с разметкой web-страницы. Таким образом, посредством языка JavaScript реализуется возможность программирования на стороне клиента. Язык JavaScript предоставляет возможность доступа к элементам разметки web-страницы посредством объектов.

Список функций, реализованных в модуле:

– Перевод градусов в радианы

– Расчёт расстояния между двумя точками на карте по координатам их широты и долготы

– Реализация алгоритма Дейкстры

– Функция получения случайного числа из выбранного отрезка. Реализация сама выбирает начальное зерно для алгоритма генерации случайных чисел; оно не может быть выбрано или сброшено пользователем.

– Получение случайного гос. номера автомобиля перед запуском процесса генерации дорожного трафика.

Рассмотрим функцию генерации случайного гос. номера автомобиля.

```
function generateCarNumber() {
    let letters = ['A', 'B', 'C', 'E', 'H', 'K', 'M', 'O', 'P', 'T', 'X', 'Y'];
    let numbers = Math.floor(1 + Math.random() * (999 - 1 + 1));
    return letters[Math.floor(Math.random() * (letters.length))]
        + numbers.toString().padStart(3, '0')
        + letters[Math.floor(Math.random() * (letters.length))]
        + letters[Math.floor(Math.random() * (letters.length))]
        + '13'
}
```

Данная функция содержит две переменных: массив доступных для номера букв (имеющих латинские аналоги) и число, которое получается в результате случайной генерации чисел в промежутке от 1 до 999. В результате своей работы функция возвращает строку вида «X000XX00» по следующему сценарию

– Выбирает случайную букву из массива доступных букв

– Сгенерированное число преобразуется в строку, после чего заполняется нулями спереди, если его длина не равна трем

– После чего случайно выбираются еще две буквы из массива доступных букв

Пример результатов работы программы: «X856TT13», «Y484PY13», «K530TC13», «E043EX13», «A097YY13».

4.4 Модуль работы с моделированием движения городского трафика.

Одним из важнейших модулей системы является модуль для работы с моделированием движения городского трафика. Кроме уже рассмотренной функции генерации уникальных номеров автомобилей, есть еще несколько основных функций, которые обеспечивают симуляцию движения городского трафика.

Общее число машин, задаваемое перед началом симуляции, генерируется в следующем виде:

```
let car = {
    speed: null,
    route: {
        start: null,
```

```
        finish: null
    },
    number: null,
    onRoad: false,
    length: null,
    distanceBetweenCars: null,
};
```

Основными полями объекта созданного автомобиля являются скорость его движения (рассчитывается динамически и зависит от типа дорожного покрытия, качества дорожного покрытия и присутствия различных дорожных ситуаций на ребре, по которому он движется), ребро, на котором он находится в данный конкретный момент времени (точка начала и конца ребра), гос. номер, находится ли автомобиль на дороге или нет, длина автомобиля (от 50 до 20 метров) и расстояние между машинами (от 2 до 10 метров).

Рассмотрим упрощенную схему генерации дорожного трафика. После того, как машина попадает на дорогу, происходит следующее:

– Проверяется начальная точка движения автомобиля

– Рассчитывается скорость движения автомобиля на конкретном участке пути (ребре)

– Рассчитывается время движения автомобиля на конкретном участке пути (ребре)

– Если машина полностью ребро, в следующий такт он будет перенаправлена на следующее ребро с пересчетом ее динамических характеристик

– Если машина полностью проехала маршрут следования, она удаляется с карты и в случае последующего ее выбора, новый ее маршрут начнется с точки завершения предыдущего маршрута

Кроме того, каждый такт (заранее определенный пользователем момент времени) происходит обновление информации о ребре графа (остаток пропускной способности, текущая загруженность, максимальная загруженность каждый «час»), что отражается в виде перекрашивания ребра (рисунок 4.6).

Рисунок 4.6 – наглядная информация о загруженности ребра

Все предыдущие модули могут работать без наличия интерфейса для работы с системой и участия в их работе пользователя. Для наглядного отображения всех происходящих действий работает модуль обработки действий пользователя.

4.5 Модуль обработки действий пользователя.

Все действия пользователя во время пользования системой отслеживаются модулем обработки действий. В общем виде действия пользователя можно представить в виде Use-case диаграммы (рисунок 4.7).

Рисунок 4.7 – Use-case диаграмма приложения

Большинство действий пользователя отслеживается с помощью метода языка JavaScript EventTarget.addEventListener, который регистрирует определенный обработчик события, вызванного на EventTarget (должен быть либо существующим элементом в документе, либо Document, либо Window, либо любым другим объектом, который поддерживает события). В общем виде синтаксис метода следующий: target.addEventListener(type, listener[, options]), где type – строка, указывающая на типа прослушиваемого события, listener – объект, которые принимает уведомление, когда событие указанного типа произошло.

В качестве примера будет приведена функция, которая вызывается при клике на кнопку «Сгенерировать маршруты»:

```
document.getElementById('generate-directions').addEventListener("click", ev => {
    showNotification('Генерация маршрутов была начата');
    document.getElementById('generate-directions').setAttribute('disabled',
'disabled');
    generateDirections();
});
```

В данной функции метод document.getElementById() возвращает ссылку на элемент, который имеет атрибут id с указанным значением (в нашем случае id кнопки «Сгенерировать маршруты»), после чего указывается обработчик события (в нашем случае «клик») и тело самого события. В теле события вызывается три функции:

– Пользователю показывается уведомление о начале процесса генерации маршрутов

– Кнопке «Сгенерировать маршруты» присваивается атрибут «недоступна»

– Запускается функция генерации возможных маршрутов

Кроме обработчиков события есть вспомогательные функции. Одна из них – показ пользователю уведомления об изменениях, которые повлекли за собой его действия (рисунок 4.8)

Рисунок 4.8 – Уведомления о действиях пользователя

Код функции показа уведомлений:

```
function showNotification(text) {
    let noteOuter = document.getElementById('notification-container');
    let note = document.createElement('div');
    note.innerText = text;
    note.classList = 'notification';
    noteOuter.appendChild(note);
    setTimeout(() => {
        note.style.display = 'none';
```

```
        note.remove();
    }, 5000);
}
```

Данная функция принимает на вход один параметр – текст уведомления. После чего этот текст подкладывается в блочный элемент и отображается на экране. После 5 секунд данный блок постепенно теряет свою прозрачность и полностью удаляется из DOM-дерева страницы. Функция вызывается в паре с обработчиком события и выполняется в его теле.

Для более комфортной работы пользователя с системой был реализован виджет, позволяющий пользователю мгновенно выполнить наиболее часто повторяющиеся действия во время взаимодействия с системой и увидеть важные статистические данные, не открывая главное боковое меню приложения:

a) Возможность следить за «временем суток»

b) Возможность видеть количество машин на дорогах города

c) Возможность приостановить и продолжить симуляцию движения автотранспорта

d) Возможность сохранить созданную модель

e) Возможность изменить граф автомобильных дорог

Кроме того, реализованный виджет имеет возможность «drag-and-drop» - пользователь может переместить его в любое место на экране.

Пример виджета представлен на рисунке 4.9:

Рисунок 4.9 – Виджет, предназначенный для прямого взаимодействия системы с пользователем

Все начальные параметры системы хранятся в модуле, содержащем начальные настройки и параметры системы.

4.6 Модуль, содержащий начальные настройки и параметры системы

Данный модуль был создан с целью повышения читаемости кода проекта в целом и каждой страницы скриптов отдельно. В нем хранятся глобальные переменные, настройки и параметры проекта. Часть кода данного скрипта представлено ниже:

```
let run = false,
    hours = 0,
    minutes = 0,
    nodesNumber = 0,
    currentGeneratingModel = '',
    minuteVsReal = 100, //1000 - 24 минуты реального времени
    roadCoefs = [0.08, 0.02, 0.01, 0.02, 0.06, 0.14, 0.27, 0.30, 0.52, 0.68, 0.99,
0.84, 0.74, 0.75, 0.83, 0.97, 0.99, 0.95, 0.79, 0.47, 0.26, 0.24, 0.19, 0.12],
    carsInCity = 1000,
    edgeRefreshInterval,
    tempModelParameters,
    canvasArray = [],
    wasFilled = false,
    canvasArrayInterval = null,
    roadTripTimeOut,
    roadAccidents = [
        {name: 'Авария', speedCoefNerf: 0.9, bandwidthCoefNerf: 0.87},
        {name: 'Ремонт дороги', speedCoefNerf: 0.95, bandwidthCoefNerf: 0.8},
        {name: 'Препятствие (поломанный автомобиль)', speedCoefNerf: 0.99,
bandwidthCoefNerf: 0.9},
    ]
```

Каждый из этих параметров может быть изменен в процессе работы с приложением либо пользователем, либо системой. При сохранении каких-либо настроек настройки по умолчанию применяться не будут.

4.7. Модуль работы с визуализацией данных

Данный модуль был реализован для возможности получения пользователем информации о рёбрах графа (суточная интенсивность движения

автомобилей, соотношение допустимой и максимальной загруженности ребра, процент времени, когда ребро было перегружено) в режиме онлайн. Визуально модуль предоставляет пользователю информацию в трех различных видах:

– Линейный график – кривая, показывающая суточную интенсивность движения автомобилей для конкретного выбранного ребра или для всех рёбер вместе в отдельном окне. По оси X откладывается время суток (часы, минуты), по оси Y – количество автомобилей, проехавших по ребру за это время.

– Диаграмма типа «радар» - круговой график, на котором откладывается два значения. Первое – допустимая загруженность ребра (максимальная пропускная способность), второе –загруженность, которая программно рассчиталась в результате процесса моделирования. В случае, если второе значение превышает первое, пользователь получает уведомление от системы о том, что необходимо улучшить сформированную структуру УДС.

– Столбчатая диаграмма - диаграмма, представленная прямоугольными зонами (столбцами), высоты которых пропорциональны величине процента времени каждого часа, когда ребро было перегружено. Проанализировав эту диграму пользователь может понять, в какое время суток максимальная загруженность превышает допустимую загруженность.

Пример диаграммы приведён на рисунке 4.10:

Рисунок 4.10 – Диаграмма типа «радар»

При изменении параметров ребра, которые влияют на его пропускную способность, график изменится в режиме онлайн, что позволит пользователю своевременно сделать выводы о правильности внесенных изменений.

4.8 Модуль генерации рекомендаций по улучшению структуры УДС.

Данный модуль был реализован для того, чтобы пользователь в режиме онлайн получал информацию, о тех ребрах построенного им графа, у которых фактическая пропускная способность оказалась больше максимально допустимой (рассчитывается динамически).

Во время процесса моделирования администратор имеет возможность поменять параметры рёбер: класс автомобильной дороги, тип и качество дорожного покрытия, добавить дорожные ситуации, что повлияет на пропускную способность, количество полос движения, минимальную и максимальную скорость движения автомобилей на нем.

Для возможности своевременного исправления ошибок в построении графа дорожной сети администратор может ознакомиться с «вердиктом системы» - информацией о рекомендациях по улучшению структуры УДС. Пример списка рекомендаций представлен на рисунке 4.6.

Генерируя отчет, система основывается на полученных ребрами «штрафных баллов» - баллы, которые получает ребро, если ее пропускная способность была использована более, чем на 99%. Алгоритм работы следующий:

– Каждый момент времени система сканирует пропускную способность каждого ребра

– Если количество «штрафных баллов», набранных ребром больше 1, запускается процесс по составлению рекомендаций

– По очереди проверяются возможность увеличения качества дорожного покрытия, типа дорожного покрытия, возможность устранения дорожных ситуаций на ребре

– В случае, если в п.3 невозможно выполнить никаких действий, система предлагает добавить альтернативные маршруты между точками старта и финиша выбранного ребра

Алгоритм работы модуля по генерации рекомендаций:

```
g.edges.forEach((value, index) => {
    if (value.points > 1) {
        *******
        if (changeRoute) {
            if (value.quality !== "0") {
                *******
            }
            if (value.pavementType !== "0") {
                *******
            }
            value.hasAccidents.forEach((v, k) => {
                *******
            })
            if (value.class !== '0') {
                *******
            }
        }
        if (changeRoute) {
            *******
        }
        *******
    }
});
```

После изменении хотя бы одного свойства ребра, его «штрафные баллы» обнуляются, а ребро удаляется из списка рекомендованных к улучшению пропускной способности.

Все рекомендации системы выводятся в понятном для пользователя виде (рисунок 4.11).

Рекомендации по улучшению структуры УДС

1) Возможные пути увеличения пропускной способности ребра e9

- Улучшить качество дороги до состояния "Хорошее"
- Улучшить тип дорожного покрытия на "Цементобетон"
- Увеличить количество полос до "2"
- Улучшить класс дороги до "Дорога без классификации"

2) Возможные пути увеличения пропускной способности ребра e11

- Улучшить качество дороги до состояния "Хорошее"
- Улучшить тип дорожного покрытия на "Цементобетон"
- Увеличить количество полос до "2"
- Улучшить класс дороги до "Дорога без классификации"

Рисунок 4.11 – Карточка рекомендаций по улучшению УДС

Все модули системы в совокупности при взаимодействии друг с другом обеспечивают надежную и бесперебойную работу приложения в целом, удобство и комфорт его использования пользователем.

5 Реализация имитационной модели движения транспортного потока абстрактно и на примере нескольких районов города.

5.1 Постановка задачи моделирования и подготовка имитационной модели.

Для проверки корректности работы созданной системы было принято решение создать две отличающихся друг от друга имитационных модели. Основное отличие которых будет состоять в пропускной способности участка дорожной сети, соединяющей соседние районы города Саранск.

Этапы создания имитационной модели:

– Создание новой модели, выбор ее названия, подбор желаемых параметров. В нашем случае это будет «Модель нескольких районов Саранска». После чего были заданы режимы распределения интенсивности движения автомобилей по дорогам в течение суток (бимодальный) и коэффициенты усредненных характеристик движения (от 0.01 в точках минимума до 0.86 в точке максимума) (рисунок 5.1)

Рисунок 5.1 – Начальная настройка имитационной модели

– После получения уведомления об успешном создании новой модели необходимо приступать к построению графа дорожной сети. В результате получилась модель, состоящая из 34 вершин, 44 рёбер и 1122 маршрутов. До начала процесса генерации трафика все рёбра имеют одинаковые параметры:

– Класс автомобильной дороги: уличная дорога (Дороги, которые служат доступом к жилью, без функции связывания населенных пунктов)

– Тип дорожного покрытия: цементобетон

– Качество дорожного покрытия: среднее

– Количество полос движения: 2 полосы

– Минимальная и максимальная скорость движения автомобилей: 40 и 60 км/ч соответственно

– Дорожные ситуации отсутствуют

Кроме того, начальные параметры ребра уменьшает его пропускную способность на 41.5% и на 28.5% скорость автомобилей на нём.

Граф автомобильных дорог визуально разделился на две части (рисунок 5.2).

Рисунок 5.2 – Построенный граф дорожной сети

В результате построения графа обнаружилось, что дороги, соединяющие районы города (рёбра №24, №25 и №41) теоретически будут более нагруженными в течение суток, что может повлечь за собой нехватку пропускной способности минимум одного из них.

– После построения графа дорожной сети последним этапом перед запуском процесса генерации трафика является настройка двух параметров: максимальное количество автомобилей, которое может находиться на дорогах города при условии, что коэффициент усредненных характеристик движения в этот час равен 1 (100%), и «время суток», с которого начнется моделирование. По умолчанию настройки принимают значения **1000** и **00:00** соответственно. Изменим количество автомобилей до 5000 и запустим процесс.

5.2 Имитационное моделирование дорожно-транспортных сетей

После прохождения системой моделирования «утреннего пика», приостанавливаем симуляцию и смотрим промежуточные результаты.

В каждый момент времени имеется возможность следить за загруженностью любого ребра дорожной сети и получать информацию вручную (основываясь на легенде карты) и автоматически (с помощью диаграмм рёбер). На рисунке 5.3 представлен график зависимости между допустимой и максимальной загруженности ребра №25.

Рисунок 5.3 – Диаграмма ребра типа «радар»

Из диаграммы видно, что с 7:00 до 10:00 утра максимальная загруженность ребра была больше величины допустимой загруженности (часовая активность движения автомобилей – 9625). Для исправления ситуации необходимо воспользоваться интерфейсом, который предоставляется модулем рекомендаций по улучшению УДС. Модуль предлагает пользователю несколько решений: от более простых до сложных с точки зрения дороговизны и трудозатрат по увеличению пропускной способности ребра (рисунок 5.4).

Рекомендации по улучшению структуры УДС

1) Возможные пути увеличения пропускной способности ребра Edge 25

- Улучшить качество дороги до состояния "Хорошее"
- Улучшить тип дорожного покрытия на "Цементобетон"
- Увеличить количество полос до "4"
- Улучшить класс дороги до "Дорога без классификации"

Рисунок 5.4 – Рекомендации по улучшению структуры УДС

Возможные пути увеличения пропускной способности ребра №25, предложенные системой:

– Улучшить качество дорожного покрытия
– Улучшить тип дорожного покрытия
– Увеличить количество полос
– Улучшить класс дороги

Наиболее легкий способ увеличения пропускной способности – улучшить качество дорожного покрытия. Это можно сделать, отредактировав свойства нужного ребра. После установления состояния качества дорожного покрытия как «отличное», пропускная способность ребра уменьшается на 10% от максимального вместо 41,5%, которые были ранее. Сохранив изменения, есть возможность в режиме онлайн просмотреть полученный результат (рисунок 5.5).

Рисунок 5.5 – Результат изменения качества дорожного покрытия

Из рисунка видно, что предпринятые действия не в полной мере дали желаемый результат. Следующее возможное действие – улучшение качества дорожного полотна. Повторив манипуляции, получаем график, на котором максимальная загруженность также превышает допустимую загруженность ребра графа. Следующий шаг для решения задачи увеличения пропускной способности УДС – увеличение количества полос движения автомобилей на выбранном участке дорожной сети. Данная задача является более трудоемкой и дорогостоящей для транспортной перепланировки городов.

Выбрав пункт «Изменение класса автомобильной дороги», выбираем «дорогу без классификации» - наименее важные дороги в системе страны - то есть второстепенные дороги более низкой классификации, но которые служат цели, отличной от доступа к собственности. Данный класс дорог отличается от начального («уличная дорога») количеством полос движения – 4 полосы (по 2 в

каждую сторону), что увеличивает пропускную способность ребра в 2 раза. На рисунке 5.6 показан результат применения изменений:

Рисунок 5.6 – Результат изменения класса автомобильной дороги

Из рисунка следует, что допустимая загруженность превышает максимальную загруженность ребра. После получения данного результата, система автоматически удаляет ребро из списка рекомендованных для улучшения структуры УДС и его обнуляет «штрафные» баллы.

Данное решение имеет существенное преимущество – низкое влияние дорожных ситуаций на пропускную способность выбранного ребра (рисунок 5.7).

Рисунок 5.7 – Пропускная способность ребра с возникшими дорожными ситуациями

Из рисунка видно, что одновременное возникновение трех дорожных ситуаций (авария, ремонт дороги, препятствие на полосе движения) понизило пропускную способность ребра на 37.4%, что в конечном итоге не повлияло на изменение соотношений допустимой и максимальной загруженности.

Кроме предложенного выше варианта существует еще один, наиболее дорогой, с точки зрения изменения структуры дорожной сети – построение объездных дорог для разгрузки наиболее загруженных частей города. Рассмотрим данный пример.

В качестве решения проблемы пропускной способности дорог, соединяющих два района города, добавим на карту еще 2 ребра (№№ 44,45). Каждое из них изначально будет иметь стандартные настройки (рисунок 5.7).

Рисунок 5.7 – Добавление новых ребер на карту дорожной сети

После добавления на карту новых рёбер, запускаем процесс генерации маршрутов и моделирование. Для того, чтобы не дожидаться большого появления автомобилей на дороги к 8:00 утра (моделирование начинается в 00:00), воспользуемся функцией выбора «времени суток». После чего, смотрим статистику ребра №25 (рисунок 5.8).

Рисунок 5.8 – Результат добавления новых участков УДС

Из рисунка видно, что максимальная почасовая загруженность ребра не превышает допустимую. Кроме того, максимальная интенсивность движения автомобилей не превышает 7000 в точке максимума (6531), что практически на 3000 ниже, чем в случае с увеличением количества полос движения на ребре графа.

5.3 Результаты работы системы имитационного моделирования

Результатом работы системы стало выявление проблемы перегрузки нескольких рёбер построенного графа. Пользователю было предоставлено на выбор несколько решений по улучшению структуры УДС:

– Интенсивный – улучшение параметров автомобильной дороги (класс, тип дорожного покрытия, качество дорожного покрытия, количество полос движения). Плюсы данного метода заключаются в относительной дешевизне его исполнения, в увеличении пропускной способности выбранного ребра и скорости движения автомобилей на нём. Минусы метода –влияние дорожных ситуаций на пропускную способность ребра, отсутствие альтернативных путей передвижения между районами города

– Экстенсивный – добавление на карту новых автомобильных путей. Плюсы данного метода – низкое влияние дорожных ситуаций на любом из рёбер графа, появление альтернативных путей передвижения, добавление новых

маршрутов. Минусы – дороговизна постройки дорожного полотна, затраты на проектирование и инфраструктуру. Данный метод не увеличивает пропускную способность выбранного ребра, а лишь распределяет нагрузку между ним и только созданными, что сказывается на более продолжительном сроке эксплуатации дорожного полотна, менее скором его износе.

Кроме того, система позволяет посмотреть почасовую статистику всех рёбер на одной карточке (рисунок 5.9)

Рисунок 5.9 – Подробная статистика всех рёбер графа

Данная функция позволит пользователю сделать выводы о загруженности каждого ребра и спланировать реконструкцию УДС без потери пропускной способности на соседних дорогах путем распределения нагрузки на каждое из них. Также администратор имеет возможность смоделировать в режиме онлайн наиболее частые дорожные ситуации, влияющие на пропускную способность ребра (авария, ремонт дороги, препятствие на полосе движения).

ЗАКЛЮЧЕНИЕ

Реализованный программный комплекс планируется использовать для создания динамических транспортных моделей с возможностью прогнозирования характера транспортного потока в зависимости от текущей транспортной ситуации: возникновение ДПТ, ремонт дороги, в режиме реального времени. Он интегрирует всех участников движения в единую математическую мультимодальную транспортную модель, которая в последствие служит инструментом для принятия стратегических решений относительно развития транспортной структуры города или региона.

В первой главе были проанализированы литература, научные исследования в области имитационного моделирования, понятия транспортной инфраструктуры населенного пункта - одной из важнейших инфраструктур, определены понятие и задача математических моделей - определение и прогноз всех параметров функционирования транспортной сети обеспечивающих жизнь городов и регионов, их классификация, основанная на задачах, для решения которых они применяются.

Во второй главе были рассмотрены понятие пропускной способности транспортного узла автомобильных дорог, бимодальные и унимодальные распределения интенсивности движения трафика в течение суток, а также экспресс-методы определения суточной интенсивности, основанные на статистике. Было выбрано формальное описание указанных факторов – транспортный граф, узлы которого соответствуют перекресткам и станциям внеуличного транспорта, дуги — сегментам улиц и линий внеуличного транспорта, сформулирован критерий, на основании которого пользователь оценивает альтернативные пути и способы движения – обобщенная цена пути.

В третьей главе были проанализированы теоретическая, возможная и практическая пропускные способности нерегулируемых пересечений автомобильных дорог, выяснено, что каждой интенсивности движения по

главному направлению соответствует определенное число автомобилей второстепенного направления, которые могут пересечь или влиться в основной поток. Были сформулированы функциональные требования к системе, предложена исчерпывающая модель данных, выбраны способ и инструменты реализации сервиса. Кроме того, было описано поведение системы, реализованы основные алгоритмы, образующие ядро приложения. После чего был описан разработанный горизонтальный прототип программного комплекса.

В четвертой главе были описаны модули и компоненты системы, которые в совокупности при взаимодействии друг с другом обеспечивают надежную и бесперебойную работу приложения в целом, удобство и комфорт работы с ним пользователя.

В пятой главе была описана работа системы генерации имитационной модели движения транспортного потока на примере нескольких районов города. Была поставлена задача моделирования, выявлена проблемная область. После чего системой автоматически было сгенерировано несколько предложений по улучшению структуры УДС и нормализацией соотношения максимальной и допустимой загруженностей автомобильных дорог. Кроме того, приложение позволяет спланировать реконструкцию УДС без потери пропускной способности на соседних дорогах путем распределения нагрузки на каждое из них.

Таким образом, приложение позволяет оценить транспортную ситуацию на участках УДС, выбрать оптимальную схему дорожного движения, анализировать пропускную способность. В будущем планируется реализовать функционал, позволяющий оптимизировать работы сигнальных устройств, а также проводить анализ пешеходного движения.