**Введение**

Оптимизация маршрута – задача, возникающая перед любыми организациями в самых различны отраслях, которые характеризуются наличием транспортной и/или логистической сети. Задача нахождения оптимального маршрута которая впервые освещалась в 1832 году в книге «Коммивояжера». Она вовсе не нова и является актуальной на сегодняшний день, рассматриваются и разрабатываются новые алгоритмы, программы и приложения, которые позволяют работать с количеством с большим количеством узлов и обладают достаточно большой эффективностью, что позволяет достаточно быстро решать поставленные задачи. Существует множество онлайн и офлайн сервисов, решающих задачу построения оптимального маршрута. Рассмотрим данную задачу на нашем примере: существует определенное число датчиков загрязнения воздуха, находящихся в черте города, задача приложения просчитать оптимальный маршрут между двумя точками на карте.

Вопросы оптимизации маршрута поднимаются в трудах таких известных деятелей математики как Уильям Роуэн Гамильтон, Джордж Данциг, Ричард Мэннинг Карп, Дэвид Аплгейт, Герхард Райнельт[1]

Цель данной работы заключается в интегрировании алгоритма задачи коммивояжера в Google.Maps с использованием технологии на примере приложения поиска оптимального маршрута между двумя точками на карте в условиях загрязнении окружающей среды.

Дипломная работа разделена на четыре главы. В первой главе «Обзор научно-технической литературы» представлен обзор предметной области, описывается функциональность приложения, определяются границы проекта, проводится анализ алгоритмов для решения задачи. Во второй главе «Проектирование системы» представлено проектирование программного средства, выбор необходимого алгоритма поиска оптимального пути и обоснование выбора. В третьей главе «Разработка системы» рассказано, какие инструменты использовались при создании проекта, анализ выбора языка программирования и элементы реализации. Четвертая глава посвящена испытанию программного средства посредством unit-тестирования.

ГЛАВА 1. Обзор научно-технической литературы

Первая глава состоит из обзора предметной области, включающего в себя описание и выбор алгоритма решения задачи поиска оптимального пути, а также анализа уже существующих приложений. Ставится задача диплома, формулируются требования к ее реализации.

Раздел 1.1 Анализ нормативной базы в отношении загрязнения окружающей среды.

Раздел 1.2 Алгоритмы поиска оптимального пути и их сравнение

Алгоритмы для решения задачи коммивояжера можно разделить на точные (exact algorithm) и неточные (non-exact algorithm). Точные алгоритмы включают в себя перебор всех возможных вариантов, в частных случаях решения могут быть быстро найдены, но в целом осуществляется перебор 𝑛! циклов. Вторые в общих случаях применяются для задач, которые невозможно решить точно (вычисление определенных интегралов, решение нелинейных уравнений, извлечение квадратного корня…), если существующие точные решения требуют значительных и неоправданных временных затрат при высокой сложности задачи, и как часть более сложного алгоритма, с помощью которого задача решается точно. [2]

1.2.1. Точные алгоритмы

В свою очередь существует две группы точных алгоритмов — одна из них использует методы релаксации линейного программирования TSP: алгоритм Гомори, метод внутренней точки, метод ветвей и границ; вторая, меньшая группа, использует методы динамического программирования. Характерная особенность методов обеих групп — гарантия нахождения оптимальных решений при общей трудоемкости процесса. [3]

Полный перебор (Brute Force)

Один из самых очевидных методов решения задачи коммивояжера – метод полного перебора или грубой силы. Его суть заключается в переборе всех возможных вариантов путей, алгоритм решения можно записать как:

1. найти общее число возможных гамильтоновых контуров;
2. найти вес каждого гамильтонова контура, сложив вес всех его ребер;

выбрать гамильтонов контур с минимальным весом, который и будет оптимальным. Метод полного перебора обладает рядом преимуществ – он гарантирует нахождение решения задачи TSP, при этом он прямолинеен и прост в исполнении. В то же время, алгоритм считается неэффективным при работe с большим объемом данных, так как для нахождения оптимального маршрута требует найти вес (𝑛 − 1)! гамильтоновых контуров.

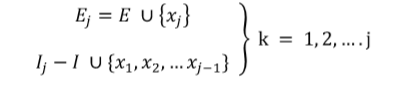
Метод ветвей и границ (Branch and Bound)

Метод ветвей и границ часто используется для нахождения оптимального решения задач комбинаторной оптимизации. Его суть заключается в разбитии множества на подзадачи и исключении заведомо неоптимальных решений.

Пусть граф 𝑉 содержит все города, 𝛱 – множество всех перестановок городов , покрывающее все возможные решения. Рассмотрим перестановку 𝜋∈𝛱, в которой каждому городу назначается преемник – 𝑖 для 𝜋𝑖 города. Таким образом, тур можно записать как (1,𝜋  (1),𝜋  (𝜋  (1)),…,1). Если число городов в туре равно 𝑛, тогда перестановку называют циклической. Задача о назначениях ставит перед собой цель найти циклические перестановки, а задача коммивояжера преследует ту же цель, но с ограничением, что у этих перестановок должна быть минимальная стоимость. Метод ветвей и границ в первую очередь находит решение задачи о назначениях, стоимость которой для 𝑛 городов довольно большая и асимптотически равна 𝑂 (). [4]

Если был найден полный тур, то полученное значение также является решением задачи коммивояжера. В противном случае проблема разделяется на несколько подобластей, каждая из которых исключает некоторые дуги подтура, таким образом исключая сам подтур. Метод, с помощью которого высчитывается, какую дугу следует удалить, называют правилом ветвления. Важное замечание – не должно существовать дублированных подзадач, их общее количество должно быть минимизировано.

Будем использовать критерий, гарантирующий независимость подзадач – рассматривается включенный набор дуги и выбирается минимальное число дуг, которые не принадлежат набору. Обозначим 𝐸 как множество исключенных дуг и 𝐼 как множество включенных. Разложим 𝐼. Выберем t дуги подобластей 𝑥5𝑥6…𝑥7 которые не принадлежат 𝐼. Задача разделена на 𝑡 потомков так, чтобы у потомка были исключенных дуг и включенных дуг. Запишем в виде формулы:



Но исключенная дуга подзадач и включенная дуга в области . Это значит тур, полученный решением задачи, может иметь дугу, но тур, полученный решением не содержит эту дугу. Это гарантирует отсутствие дублирующихся маршрутов.

Количество возможных решений равно (𝑛−1)!/2, для n=50 это приблизительно 3×1062. Этот метод наиболее часто используется при количестве узлов от 40 до 60. [5]

Необходимость целиком решать задачи линейного программирования во всей области допустимых решений можно считать главным недостатком вышеописанного метода. Для задач с большим объемом данных метод ветвей и границ является неоправданно трудоемким, в то же время алгоритм является надежным методом решения целочисленных задач.

Алгоритм Гомори (The Cutting Plane)

В 1954 году была представлена работа Данцига, Фалкерсона и Джонсон, описывающая новый метод решения задачи коммивояжера, который также может быть использован для решения любой проблемы



 где 𝑐≠0, 𝑆 – конечное подмножество некоторого , и таким образом это мы сможем найти точки 𝑆. Это итерационный алгоритм – каждое повторение начинается с линейной программной релаксации. Запишем в виде формулы:



где многогранник P, определенный как {𝑥∶  𝐴𝑥  ≤  𝑏}, содержит S и ограничен. Так как P ограничен, мы можем найти оптимальное решение 𝑥∗ как экстремальную точку 𝑃. Если 𝑥∗ принадлежит 𝑆, то оптимальное решение найдено (2.2); в противном случае некоторое линейное неравенство удовлетворяет все точки 𝑆 и нарушает 𝑥∗. Такое неравенство называют алгоритмом Гомори, который подробно описал его 1958, методом отсекающих плоскостей или просто отсечений. [6]

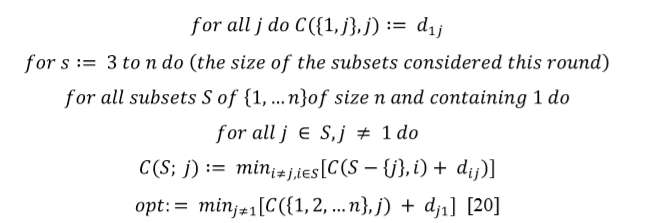
Данный метод используется для построения точных или приближенных задач, особенно часто встречается в сочетании с методом ветвей и границ и тогда называется методом ветвей и отсечений. Оба метода основаны на решении последовательности релаксированных подзадач линейного программирования. В алгоритме Гомори релаксированные подзадачи постепенно улучшают аппроксимацию целочисленной задачи, уменьшая окрестность оптимального решения. Если оптимальность не удалось получить, тогда ищется приближенное решение с погрешностью.

У метода отсечений есть преимущество над методом ветвей и границ – первые более удобны для аппаратного вычисления, так как для их решения не требуется большой объем оперативной памяти для хранения дерева решений. [7]

Метод динамического программирования (Dynamic Programming)

Рассмотрим задачу с 𝑛 городами и расстояниями между любыми двумя городами, путь начинается и заканчивается в городе . TSP может быть решена за время 𝑂(𝑛!) методом полного перебора, но алгоритм динамического программирования позволяет сократить то время до 𝑂.

Обозначим подзадачу: пусть 𝑆 будет подмножеством городов, содержащим 1 и как минимум еще один город, 𝑗 будет городом отличным от 1, обозначим через С(𝑆,𝑗) самый короткий путь, начинающийся в 1, проходящий все города 𝑆 и заканчивающийся в 𝑗. Запишем алгоритм в виде псевдо-кода.

[8]

 Данный метод используется для повышения эффективности вычислительных повторений, храня промежуточные результаты и снова используя их при необходимости.

1.2.2. Неточные алгоритмы

В целом алгоритмы данной группы предлагают потенциально неоптимальные, но быстрые решения. В свою очередь приближенные алгоритмы можно разделить на две категории: приближенные (Approximation Algorithms) и эвристические (Heuristic Algorithms).

Алгоритм Кристофидеса (Christofides’ Algorithm)

Алогоритм Кристофидеса используется для решения метрических TSP – с дополнительным условием, что для матрицы расстояний выполнено неравенство треугольника:

∀𝑖,𝑗,𝑘   ⩾ +

  Большая часть эвристических алгоритмов относятся к 2-приближенному классу. Профессор Никос Кристофайдс в 1976 году доработал один из существующих алгоритмов (метод двойного минимального остовного дерева, 𝑂()) так, что время решения задачи не превышает оптимальное время более чем на 3/2. [9]

Решение оригинального алгоритма можно записать так: найти минимальное дерево из множества всех городов, продублировать все ребра и построить эйлеров граф, построить гамильтонов цикл, пройдя каждый узел только один раз и выбирая наикратчайших путь, ведущий из каждого узла.

Алгоритм Кристофидеса состоит из последовательности следующих действий:

1. найти минимальное дерево из множества всех городов;
2. найти паросочетание с минимальным весом множества вершин нечетной степени и построить эйлеров граф;
3. найти эйлеров обход и построить гамильтонов цикл, избегая посещаемых узлов. [10]

Основное различие – дополнительное вычисление паросочетания с минимальным весом. Эта часть также самая трудоемкая, поэтому время выполнения алгоритма возрастает до 𝑂(). Проведенные тесты показали, что алгоритм Кристофидеса на 10% выше нижней границы Хелд-Карп. [11]

Алгоритм ближайшего соседа (NearestNeighbour)

Один из самых простых эвристических методов решения TSP. Главное правило алгоритма – всегда выбирать близлежащий город (соседа). Решение задачи складывается из следующих шагов:

1. выбрать любой город;
2. найти близлежащий город, не включенный в маршрут, и перейти в него;
3. проверить остались ли города, не включенные в маршрут, если ответ положительный – повторить второй шаг.
4. чтобы завершить тур добавить ребро между последним выбранным городом и первым.

В общем случае трудоемкость решения задачи равна O(). Нижняя граница стоимости оптимального маршрута на 10% выше нижней границы Хелд-Карп. [12]

Жадный алгоритм (Greedy)

Чтобы решить TSP использование жадный алгоритм, мы исследуем все ребра, выходящие из города-узла, и выбираем 𝑛 самых коротких дуг. Если те 𝑛 самых коротких дуг формируют гамильтонов цикл, тогда мы нашли оптимальное решение. [13] Трудоемкость решения задачи жадным алгоритмом равна O(𝑛6). Нижняя граница стоимости оптимального маршрута выше нижней границы Хелд-Карп на 15-20%. [10]

Алгоритм Кернигана – Лина (Lin-Kernighan)

Алгоритм Кернигана – Лина считается один из самых эффективных методов поиска оптимальных или почти оптимальных решений задачи коммивояжера. Однако разработка и реализация алгоритма не проста, так как алгоритм состоит из множества шагов, большинство из которых сильно влияет на работу алгоритма. [14] Создание алгоритма Кернигана было вдохновлено наблюдением, что статическое 𝐾 в -оптимальном методе не дает наилучшее решение. Появилась идея использовать различные стадии -оптимального метода в выполнении эвристического алгоритма. На практике было показано, что практически невозможно заранее предугадать какое 𝐾 следуюет использовать, чтобы достигнуть лучшего компромисса между трудоемкостью и качеством решения. Лин и Керниган убрали этот недостаток, введя оптимальную переменную, таким образом значение 𝐾 меняется во время выполнения алгоритма. [15] Трудоемкость при этом равна O(). []

Алгоритм поиска с запретами (TabuSearch)

Главная проблема алгоритма ближайшего соседа состоит в частом застревании в точке локального оптимума. Этого можно избежать, применив алгоритм поиска с запретами, в 1977 году предложенный Ф. Гловером. Данный метод позволяет переходить от одного локального оптимума к другому в поиске глобального оптимума, после перехода ребро попадает в список запретов и повторно не используется, кроме случаев, когда оно может улучшить построенный оптимальный путь. На практическом уровне запрещенный набор сохраняется как комбинация ранее посещаемых шагов, который позволяет построить дальнейший путь относительно текущего решения и соседних узлов. [16] Главным недостатком этого метода служит его время выполнения – трудоемкость алгоритма оценивается как 𝑂(). [11]

Муравьиный алгоритм (Ant Colony Optimization)

Муравьиный алгоритм – эффективный полиномиальных алгоритм, вдохновленный вдохновлена поведением настоящих муравьев. Впервые его принципы были описаны в 1991 Марко Дориго. Муравьям свойственно сотрудничать в поисках пищевые ресурсы, поэтому они оставляют след химического вещества, феромонов, на их пути от гнезда до источника пищи. [15] Этот тип невербальной коммуникации называют стигмергия – стимуляция, основанная на опыте предыдущих муравьев и направленная на повышение производительности. [17]

Для решения задачи коммивояжера как правило используют около 20 муравьёв. Их размещают в случайные города и отправляют в другие города. Им не позволяют дважды посещать один и тот же город, только если они не завершают маршрут. Тот муравей, который выбрал самый короткий тур, будет оставлять след феромонов обратно пропорциональный длине маршрута. Этот след феромонов будет считываться следующим муравьем при выборе города, и с большой вероятность он пойдет тем же путем, еще сильнее укрепив след. Этот процесс будет многократно повторен пока не будет найдет маршрут, достаточно короткий, чтобы быть оптимальным. Среди недостатков алгоритма хочется выделить, что первое полученное решение может оказаться одним из худших в плане оптимизации, однако при повторном решении метод выдает достаточно точный результат. [18]

Нижняя грань Хелд-Карпа (The Held-Karp Lower Bound)

Самый распространенный способ измерить эффективность эвристического алгоритма для решения TSP – это сравнить результаты с нижней гранью Хелд-Карпа. Эта нижняя грань является решением TSP, найденным за полиномиальное время при помощи симплекс-метода. Нижняя грань Хелд-Карпа приблизительно 0.8% ниже оптимальной продолжительности тура. [19] В то же время она гарантировано не превышает оптимальное время более чем на 2/3. [11]

По состоянию на 2015 алгоритм Кристофидеса считается самым эффективных методом для решения задачи коммивояжера на общих метрических пространствах, хотя известны лучшие приближения для частных случаев. Также хорошо в тестах себя показали алгоритм Кернигана-Лина и жадный эвристический алгоритм. [11]

Алгоритм Дейкстры - алгоритм на графах, изобретённый нидерландским ученым Э. Дейкстрой в 1959 году. Находит кратчайшее расстояние от одной из вершин графа до всех остальных. Работает только для взвешенных графов без рёбер отрицательного веса. Сложность алгорима , что делает его одним из самых эффективных. Он может работать как с однонаправленными, так и с двунаправленными графами.

Раздел 1.3 Анализ существующих приложений и их недостатки

Популярные картографические сервисы вроде Яндекс.Карты и Google Maps предоставляют пользователям возможность поиска оптимально пути однако в условиях загрязнения окружающей среды они не работают. При вводе нескольких координат сервис выстраивает маршрут в том порядке, в котором данные были введены. Пользователи могут выбирать средства передвижения (на машине, пешком, на наземном транспорте), но все эти настройки влияют исключительно на варианты построения маршрута между его фиксированными точками.

Анализ, провели путем сравнением десятков картографических сервисов, как русскоязычных, так и зарубежных, показал, что среди самых популярных вариантов ни одного из них не работает построение маршрута в условиях загрязнения окружающей среды. Ниже представлен подробный отчет о трех наиболее достойных внимания сервисах и их недостатках. Среди русскоязычных сервисов в первую очередь можно выделить Poncy-ru[10]. Он включает в себя конструктор карт, визуализацию GPS треков и построение оптимальных маршрутов.

Следующий русскоязычный сервис Логист [20] обладает меньшим количеством недостатков, но все они существенны:

1. среди средства передвижения возможен выбор между автомобилем (построение маршрута по проезжим дорогам) и вертолетом (прямой путь между пунктами маршрута);
2. сервис в первую очередь рассчитан для решения задач логистики и построения маршрута перевозок продукции между городами (Рис. 1.2).

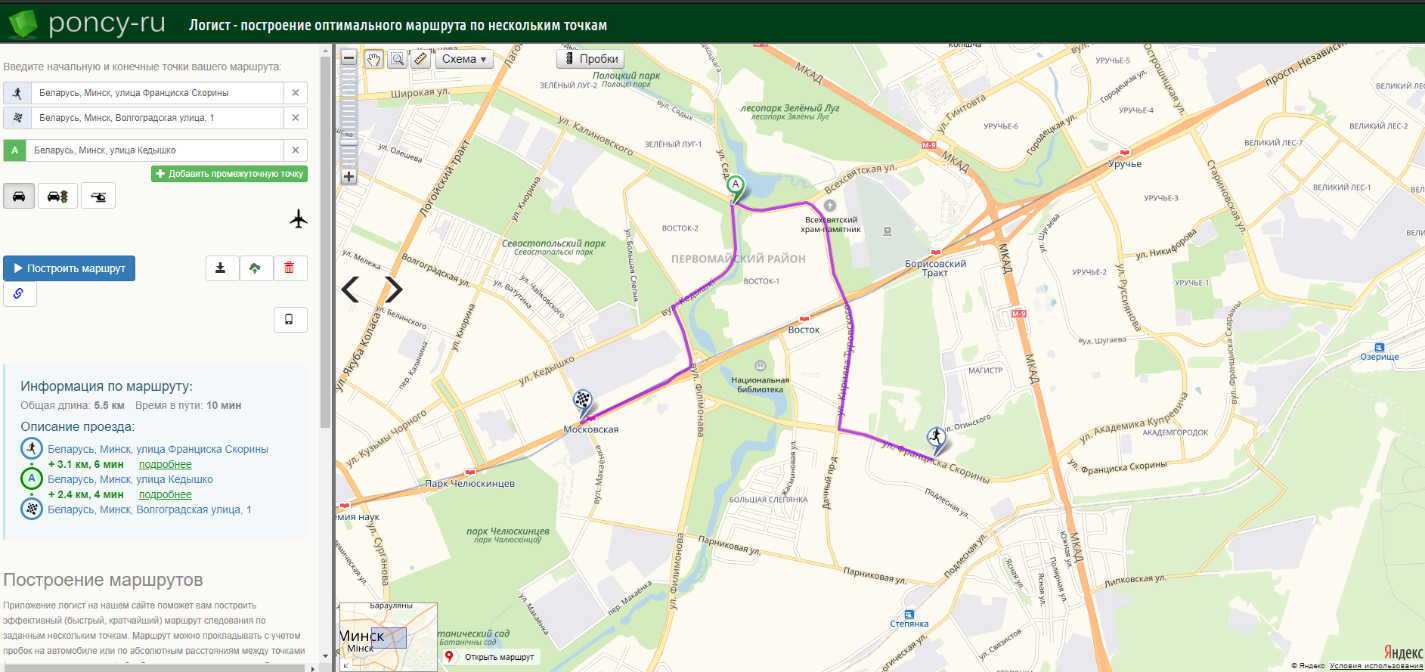


Рисунок 1.1 Сервис Логист.

Среди зарубежных картографических сервисов с функцией построения оптимального маршрута наиболее популярен Speedy Route [21]. В описании продукта сказано, что сервис рассчитывает наиболее эффективный по количеству затрат топлива маршрут между несколькими локациями, где исходная и конечная точка едины (Рис. 1.3). В первую очередь можно выделить такие недостатки как:

1. Speedy Route рассчитан исключительно для автомобилистов;
2. при вводе данных возникают трудности перевода русскоязычных названий на английский язык;
3. у сервиса предусмотрено минимальное количество пунктов маршрута – 5; маршруты, состоящие из 4 пунктов построить невозможно;
4. на сайте представлена урезанная версия карты для ознакомления, для доступа к полной версии требуется оформить платную подписку.

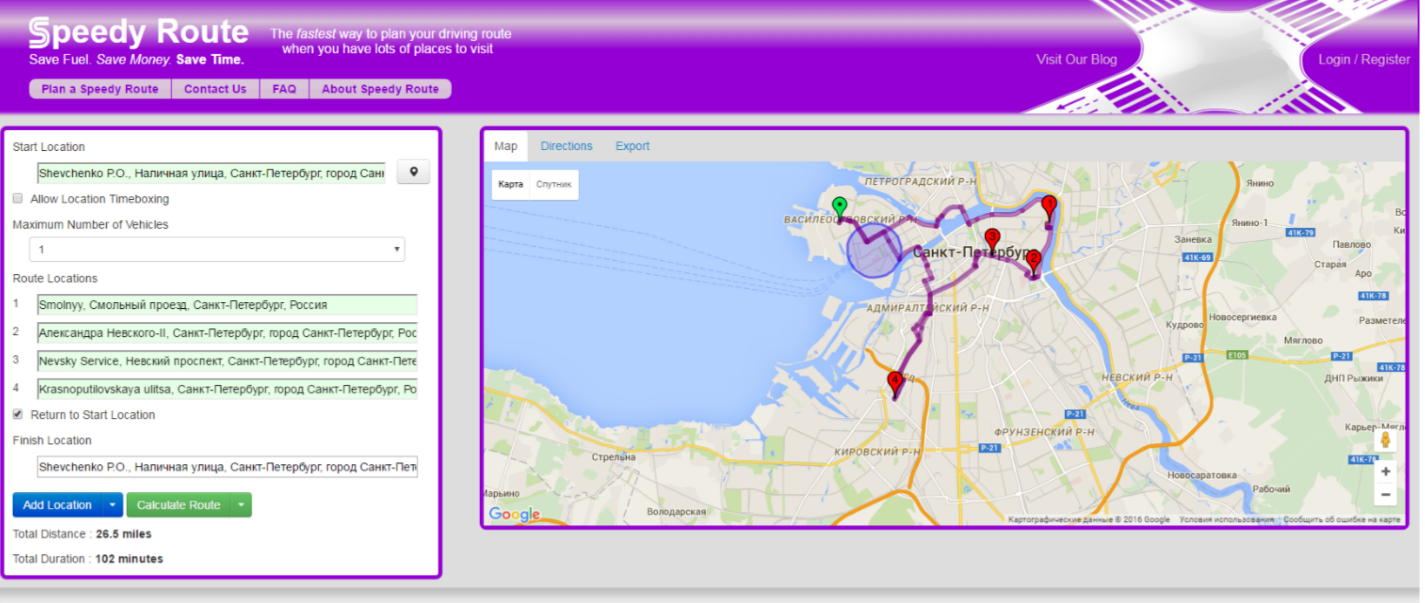


Рисунок 1.2 Сервис Speedy Route.

Суммируя все вышеперечисленное можно сказать, что пока не существуют сервисы, которые бы строили оптимальные пешеходные маршруты внутри города и одновременно были удобными и понятными для пользователей при взаимодействии с ними.

1 <https://sibac.info/journal/student/6/76841>

1.4 Выводы и постановка задачи.

Задача данного дипломного проекта – разработка алгоритма построения оптимального маршрута в условиях загрязнения окружающей среды. То есть необходимо разработать систему, которая бы учитывала различные параметры загрязнения воздуха сравшивало их с заданными ПДК (предельно допустимая контцентрация) разных газов и учитывая расстояние между сенсорами с учетом их координат прокладывало самый оптимальный марсшрут, самый короткий маршрут и самый чистый маршрут..

Для достижения поставленной цели необходимо создать клиентскую и серверную части приложения. Приложение будет размещено на удаленном сервере и предоставит клиенту пользовательскую UI в виде страницы в браузере.

В качестве алгоритма построения оптимального маршрута в условиях загрязнения окружающей среды был выбран Алгоритм Дейкстры, так как он имеет сравнительно простую реализацию и оптимальную производительность, даже с учетом работы алгоритма с большим количеством точек. Сложность алгоритма порядка . Для сравнения остальные алгоритмы имеют сложность порядка Следовательно выбранный алгоритм будет иметь большую производительность на той же аппаратной основе, ввиду своей легкости.

В процессе написания курсового проекта должны быть решены следующие задачи:

1. Разработка структурной модели для визуального представления решаемой задачи;
2. Разработка базы данных для хранения всей необходимой информации включая положение сенсоров и их показания;
3. Разработка сервера, взаимодействующего с базой данных для выполнения запросов клиента;
4. Разработка удобного и наглядного пользовательского интерфейса на клиентской части приложения, обеспечивающего эффективное использование функций приложения;
5. Тестирование разработанного ПО.

Глава 2 Проектирование приложение

2.1 Анализ и особенностей проектируемой системы поиска оптимальных маршрутов

Оптимизация маршрута – это задача которая может возникнуть в любой организации, имеющей даже косвенное отношение к логистике. Данная задача, а именно задача нахождения оптимального маршрута, была впервые освещена в 1832 году в книге «Коммивояжера». Над этой задачей работали многие ученые и не только математики и логисты. Достаточно вспомнить задачу о Кенигсбергских мостах. В которой необходимо пройти по всем мостам Кенигсберга, не пройдя по одному и тому же два и более раза. Так что задача по оптимизации маршрутов становится актуальнее с каждым днем, ввиду роста новой инфроструктуры и разрушении или реставрации новой. Пограмм и приложений, которые могут работать с большим количеством узлов и обладают достаточно большой эффективностью достаточно мало, это обусловленно наличием большого количества внешних факторов и достаточно большой сложностью алгоритмов. В виду того что большенство алгоритмов занимаются перебором ребер графа по определенным правилам – это позволяет уменьшить трудоемкость алгоритма и повысить его точность, что позволяет решать поставленные задачи быстрее, чем программы использующие более трудоемкие алгоритмы. К примеру алгоритм дискретного перебора. Который без сомнения является точным, но в виду того, что рост графа приводит к росту времени обработки, которое растет несоизмиримо сильнее. От алгоритмов подобного типа отказались.

В разрабатываемом приложении будет реализована возможность обмена данными сенсорами и пользователем. В данном случае администратор должен будет внести коордиаты сенсоров и даличие дуг между сенсорами. Для того, чтобы обеспечить эффективность и надёжность работы приложения целесообразно реализовать следующие компоненты:

1. Обеспечение доступа к данным, расположенным как на локальным, так и на удалённом компьютерах;
2. Обеспечить целосность данных на базе данных.
3. Реализовать удобный и понятный пользовательский интерфейс:
4. Возможность интеграции разрабатываемого программного модуля с другими компонентами логистических систем.
   1. Анализ требования к системе

Необходимо учесть следующие требования:

* Программа должна быть реализована с использованием языка программирования Java;
* Необходимо обеспечить сетевое взаимодействие клиентской и серверной частей приложения с помощью протокола TCP/IP(Sockets);
* В качестве источника данных должна использоваться реляционная база данных, создаваемая с помощью СУБД PostgreSQL 9.0 и выше;
* Обеспечение разделения реализации и бизнес-логики;
* К приложению должен быть разработан пользовательский GUI-интерфейс;
* GUI-интерфейс должен быть реализован посредством веб сайта

Выдвинутые требования позволят создать программный продукт, архитектура которого будет полностью соответствовать наиболее эффективной модели реализации поставленной задачи.

* 1. Моделирование алгоритма Дейкстры для поиска оптимального пути

В первой главе дипломной работы было представлено сравнение алгоритмов. Для написания программного продукта для построения оптимального пути в условиях загрязнения окружающей среды был выбран алгоритм Дейкстры. В виду сравнительно небольшой сложности алгоритма.

Рассматриваемый алгоритм состоит из следующий шагов:

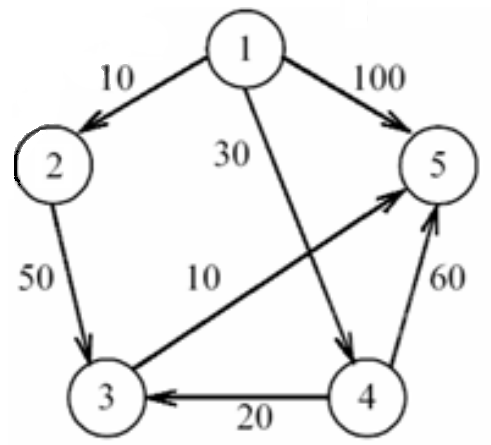
1. Всем вершинам за исключением первой, присваивается вес равный бесконечности, а первой вершине 0.
2. Все вершины не выделены.
3. Первая вершина объявлена текущей
4. Вес всех невыделенных вершин пересчитывается по формуле. Вес невыделенной вершины есть минимальное число старого веса данной вершины, суммы веса текущей вершины и веса ребра, соединяющего текущую вершину с невыделенной.
5. Среди невыделенных вершин ищется вершина с минимальным весом. Если таковая не найдена, то есть вес всех вершин равен бесконечности, то маршрут не существует. Следовательно, выход из цикла. Иначе текущей становится найденная вершина. Она же выделяется.
6. Если текущей вершиной является конечная, то путь найден и его вес есть вес конечной вершины.
7. Переход на шаг 4.

В программной реализации альгоритма Дейкстры построим множество S вешин для которых кратчайшие пути от начальной вершины уже известны. На каждом шаге к множеству S добавляется та их оставшихся вершин расстояние до которой от начальной вершины меньше чем до других оставшихся вершин. При этом дубем использовать массив D, в который записываются длинны кратчайших путей для содержание длинны кратчайших путей от начальной вершины до каждой вершины.

Помиом указанных массивов будем использовать матрицу длины С где С[I,j]-фактическиеой длинны ребер. Фактически матрица С представляет собой матрицу смежности, в которой все нулевые элеменнты заменены на бесконечность.

Для определение самого кратчайшего пути введем массив Р вершин, где P[v] будет содерщать вершину непосредственено предшествующую вершине v в кратчайшем пути.

Пседокод алгоритма представлен на рисунке.



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Итерация | S | W | D[2] | D[3] | D[4] | D[5] |
| Начало | {1} | - | 10 | ∞ | 30 | 100 |
| 1 | {1,2} | 2 | 10 | 60 | 30 | 100 |
| 2 | {1,2,4} | 4 | 10 | 50 | 30 | 90 |
| 3 | {1,2,4,3} | 3 | 10 | 50 | 30 | 60 |
| 4 | {1,2,4,3,5} | 5 | 10 | 50 | 30 | 60 |

Список использованных источников:

1 Бердж. К Задача о кратчайшем пути\\ Теория графов и ее применение = Theorie des graphes et ses applications/ Под ред И.А.Вайнштейна – Москва. Издательство иностранной литературы. 1962-С. 75-81. – 320 с.

2 Алексеев В.Е, Таланов В.А Нахождение кратчайших путей в графе. Графы. Модели вычислений. Структуры данных. – Нижний Новгород. Издание Нижегородского гос. Университета 2005 – с 236-237. – 307с.

3 Галкина В.А Построение кратчайших путей в ориентированном графе. Дискретная математика. Комбинаторика оптимизация на графах. – Москва, Издательствао “Гелиос АРВ”, 203 – с 75-94 –232с.

4. Евстигнеев В.А. Интеративные алгоритмы глобального анализа графов. Путе и покрытия. Применение теории графов впрограммировании . Под ред А.П.Ершова – Москва. Наука. Главная редакция физико-математической литературы, 1985 – С. 138-150 – 352.

2.4 Критерии оптимальности при проектировании алгоритма

По гигиеническому нормативу «Гигиенический норматив содержания загрязняющих химических веществ в атмосферном воздухе, обладающих эффектом суммации» от министерства здравоохранения Республики Беларусь 30.03.2015 №33. Установлены нормы ПДК для наиболее распространенных загрязняющих веществ.

Настоящий Гигиенический норматив устанавливает величины предельно допустимых концентраций (далее – ПДК), ориентировочных безопасных уровней воздействия (далее – ОБУВ) загрязняющих веществ в атмосферном воздухе населенных пунктов и мест отдыха населения, обладающих эффектом суммации.

Требования настоящего Гигиенического норматива обязательны для соблюдения государственными органами, иными организациями, физическими лицами, в том числе индивидуальными предпринимателями, при проектировании, строительстве, реконструкции, техническом перевооружении и эксплуатации производственных и других объектов хозяйственной деятельности.

Настоящий Гигиенический норматив используется в целях охраны атмосферного воздуха, профилактики неблагоприятного воздействия загрязняющих веществ на здоровье населения в населенных пунктах и местах отдыха населения.

Основой настоящего Гигиенического норматива являются комплексные токсиколого-гигиенические и эпидемиологические исследования с учетом международного опыта проведения данных исследований.

Для целей настоящего Гигиенического норматива используются основные термины и их определения в значениях, установленных Законом Республики Беларусь от 7 января 2012 года «О санитарно-эпидемиологическом благополучии населения» (Национальный реестр правовых актов Республики Беларусь, 2012 г., № 8, 2/1892), Законом Республики Беларусь от 16 декабря 2008 года «Об охране атмосферного воздуха» (Национальный реестр правовых актов Республики Беларусь, 2009 г., № 4, 2/1554), а также следующие термины и их определения:

лимитирующий показатель вредности – показатель, характеризующий направленность биологического действия загрязняющего вещества;

ОБУВ – временный гигиенический норматив максимального допустимого содержания загрязняющего вещества в атмосферном воздухе населенных пунктов;

ПДК – концентрация, не оказывающая на протяжении всей жизни человека прямого или косвенного неблагоприятного воздействия на настоящее или будущие поколения, не снижающая работоспособности человека, не ухудшающая его самочувствия и санитарно-бытовых условий жизни;

резорбтивное действие загрязняющего вещества – возможность развития общетоксических, гонадотоксических, эмбриотоксических, мутагенных, канцерогенных и других эффектов, возникновение которых зависит не только от концентрации вещества в атмосферном воздухе, но и от длительности его вдыхания. С целью предупреждения развития резорбтивного действия устанавливается среднесуточная ПДК (далее – ПДКс.с.);

рефлекторное действие загрязняющего вещества – реакция со стороны рецепторов верхних дыхательных путей (ощущение запаха, раздражение слизистых оболочек, задержка дыхания и другие реакции). Данные эффекты возникают при кратковременном воздействии вредных веществ, поэтому рефлекторное действие лежит в основе установления максимальной разовой ПДК (далее – ПДКм.р.);

рефлекторно-резорбтивное действие загрязняющего вещества – действие загрязняющего вещества, при котором эффекты на организм реализуются за счет влияния на нервные окончания в месте контакта с веществом и в результате всасывания вещества и поступления его в общий кровоток.

Предотвращение появления запахов, раздражающего действия и рефлекторных реакций у населения при кратковременном (не более 8 часов) воздействии загрязняющих атмосферный воздух веществ на человека обеспечивается соблюдением ПДКм.р..

Отсутствие развития вредных рефлекторно-резорбтивных и резорбтивных эффектов при воздействии на человека отдельных загрязняющих атмосферный воздух веществ в течение 24 часов обеспечивается соблюдением ПДКс.с..

Отсутствие развития вредных, в том числе канцерогенных, эффектов при длительном воздействии на человека отдельных загрязняющих атмосферный воздух веществ, обеспечивается соблюдением среднегодовой ПДК.

При одновременном содержании в атмосферном воздухе нескольких загрязняющих веществ, обладающих эффектом суммации, сумма отношений фактических концентраций каждого из них (К, К1...Кn) в воздухе к их ПДК (ПДК, ПДК1...ПДКn) не должна превышать единицы:

Перечень предельно допустимых концентраций загрязняющих веществ в атмосферном воздухе населенных пунктов и мест отдыха населения, обладающих эффектом суммации

Для

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Код | Наименование загрязняющего вещества | Химическая формула | Класс опасноссти | Максимальная разовая величина предельно допустимой концентрации (мкг/) |
| 0337 | Углерод оксид (окись углерода, угарный газ) | СО | 4 | 5000,0 |
| 0526 | Этилен | C2H4 | 3 | 3000,0 |
| 0502 | Бут-1-ен (бутилен) | C4H8 | 4 | 3000,0 |
| 2908 | Пыль неорганическая, содержащая двуокись кремния в %: менее 70 (шамот, цемент, пыль цементного производства - глина, глинистый сланец, доменный шлак, песок, клинкер, зола, кремнезем, доломит, пыль цементного производства - известняк, мел, огарки, сырьевая смесь, пыль вращающихся печей, боксит и др.) |  | 3 | 300,0 |
| 1401 | Пропан-2-он (ацетон) | С3Н6О | 4 | 350,0 |
| 0521 | Пропен (пропилен) | С3Н6 | 3 | 3000,0 |
| 1301 | Проп-2-ен-1-аль (акролеин | С3Н4O | 2 | 30,0 |

Список источников

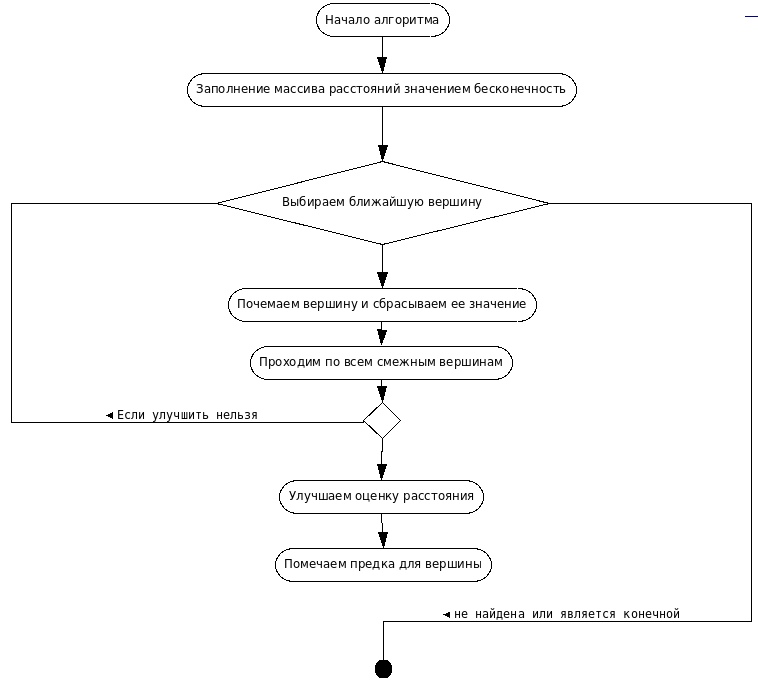
1 [Постановление №33](http://minzdrav.gov.by/upload/ru/static/acts/normativnye/postanovlenia_ministerstva/ob-utverzhdenii-gigienicheskogo-normativa-gigienicheskij-normativ-soderzhanija-zagrjaznjajuschix-ximicheskix-veschestv-v-atmosfernom-vozduxe-obladajuschix-effektom-summatsii_i_1960.html)   [Гигиенический норматив](http://minzdrav.gov.by/upload/dadvfiles/000352_158798_gig_normativ_33.doc)

* 1. Принцип построения маршрута

Для построения маршрута алгоритму Дейкстра необходим направленный взвешенный граф с неотрицательными значениями ребер. Это означает что у каждого ребра графа должно быть направление и неотрицательная стоимость прохода. Для выбора значения проход алгоритм может принимать расстояние между точками или значение загрязнения окружающей среды в сравнении с ПДК по определенному виду загрязнения. Также принимать взвешенное решение для него алгоритм просчитывает расстояние между точками и умножает на коэффициент, который зависит от загрязненности. Таким образом сенсоры, которые стоят рядом, но обладают большим уровнем загрязненности будут игнорироваться при построении маршрута.

В классическом исполнении алгоритм Дейкстры не способен воссоздать путь по которому он пришел к конечной точке для построения путей мы спроектировали класс Multilist который помимо значений вершины которая помечена алгоритмом будет хранить информацию текущей, предыдущей и следующей вершине. Что позволит алгоритму высчитать помимо общей ценности прохода из точки А в точку Б еще и путь прохождения алгоритма.

Графическое исполнение работы алгоритма

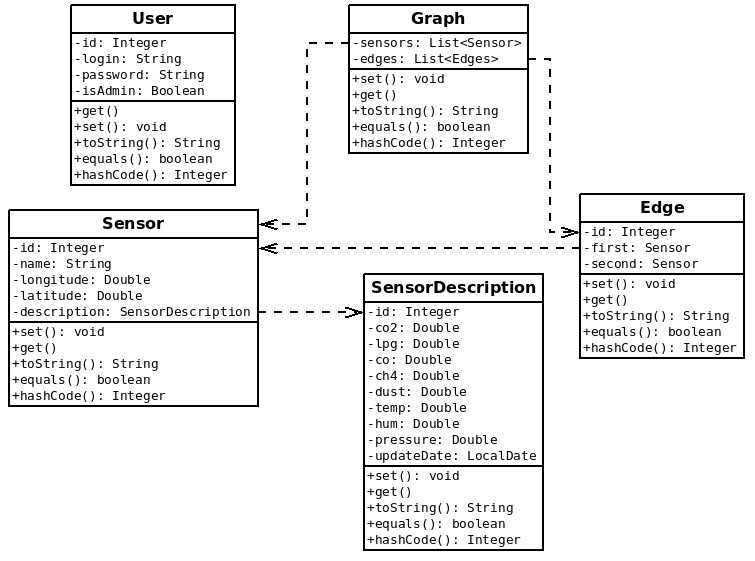


* 1. Разработка структуры данных

Моделирование системы в данном проекте будет осуществляться с помощью языка UML. Моделирование системы предполагает идентификацию сущностей, важных с той или иной точки зрения. Эти сущности составляют словарь моделируемой системы. В языке UML все сущности моделируются как классы. Класс – это абстракция сущностей, являющихся частью нашего словаря. Класс представляет не индивидуальный объект, а целую их совокупность. Чтобы изобразить систему с какой-либо точки зрения средствами UML, для организации представляющих интерес элементов используются диаграммы. В UML определено девять типов диаграмм, которые позволено комбинировать для создания нужного вида. Например, статические аспекты вида с точки зрения реализации системы удобнее визуализировать с помощью диаграмм компонентов, а динамические аспекты того же вида - с помощью диаграмм взаимодействия. Статические аспекты системной базы данных можно изобразить с помощью диаграмм классов, а динамические - с помощью диаграмм кооперации.

Диаграмма классов – основная для создания кода приложения. Для удобства представления я разбил диаграмму классов, на слои, которые они из себя представляют. По той причине, что спроектированных классов получилось более 33 и если разместить их на одной диаграмме классов, то она получилось бы слишком большой и запутанной.

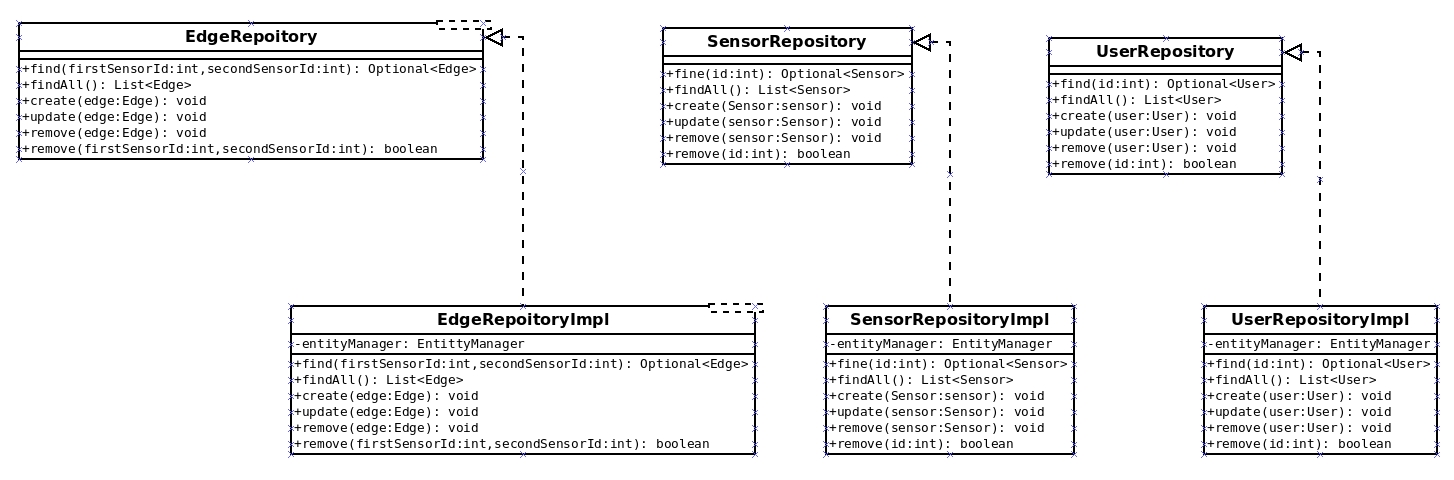
В первую очередь спроектируем диаграмму классов – сущностей. Основных объектов, с которыми программе предстоит работать.



(диаграмма сущностей)

На диаграмме есть такие классы как Sensor, Edge, SensorDescription и другие. Sensor - это основной класс проектируемой программы. В виду того что он содержит в себе всю основную информацию об реальном сенсоре его положение и координаты. Имеет параметр description, типа SensorDescription. В атрибуте данного типа хранится информация об уровне загрязненности по всем необходимым системе параметрам таким как CO, , о концентрации группы LPG газов (пропан, бутан, изобутан и другие), о содержании пыли в атмосфере, давлении воздуха, температуре и тд.

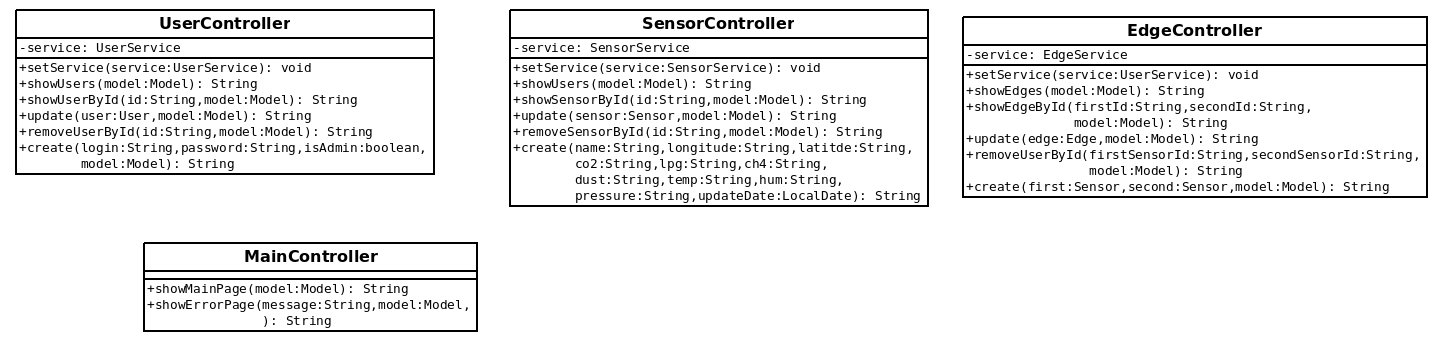
Спроектируем диаграмму классов, отвечающую за работу приложения с базой данных.



(диаграмма repository)

Данные классы проектируются для преобразования классов-сущностей в данные, хранящиеся в базе данный и наоборот. Также эти классы отвечают за все запросы к базе данных, выборке данных их обработке и преобразованию в объекты классов-сущностей, описанных выше.

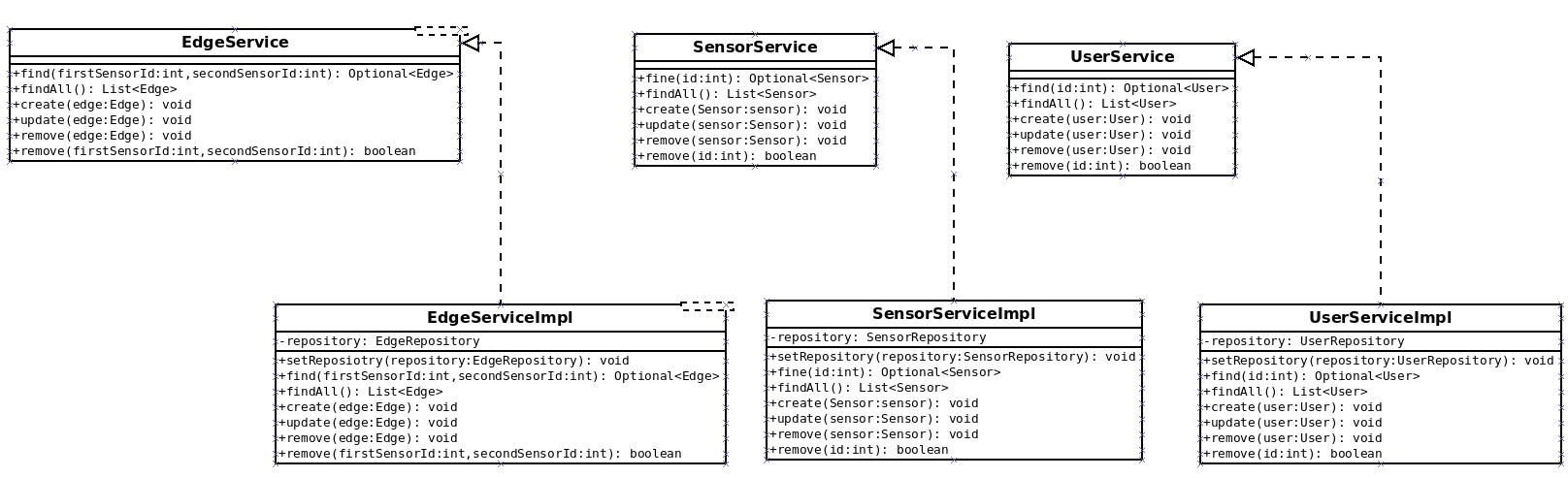
Спроектируем диаграмму классов, отвечающую за обработку запросов, приходящих с сервера.



(диаграмма контроллеров)

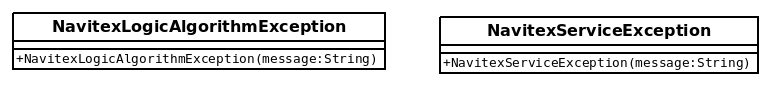
Эти классы ответственны за преобразование данных, полученных от пользователя. И предоставления обработанных данных пользователю посредством подготовленных Freemarker-ом страниц.

Спроектируем диаграмму классов, ответственных за обработку пользовательских данных и проверки их на валидность, также этот слой занимается логированием ошибок с последующей обработкой исключительных ситуаций.



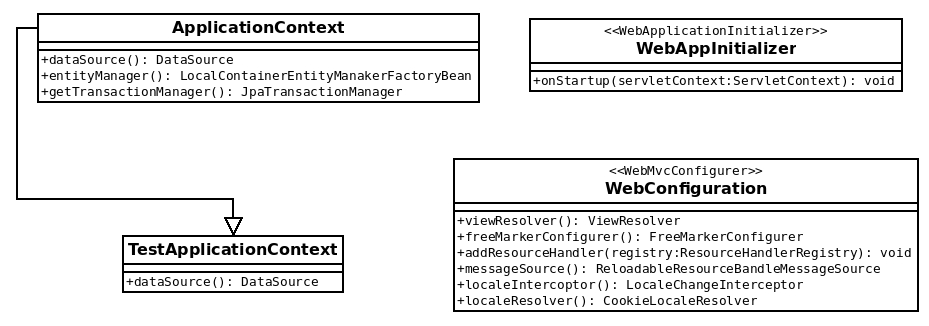
(Диаграмма service)

Спроектируем диаграмму классов, исключительных ситуаций, которые делаются для предоставления возможности обработки исключительных ситуаций, которые могут возникнуть в процессе работы приложения.



(диаграмма исключений)

Спроектируем диаграмму классов, которые отвечают за конфигурацию всего приложения в целом. Данные классы будут являться вспомогательными для построения контекста приложения.



(диаграмма config)

* 1. Проектирование базы данных

Для проектирования базы данных

2.7 Проектирование схемы работы сервера

Глава 3 Разработка приложения по теме диплома.

1. Выбор средств разработки
2. Описание разработанной системы
3. Используемые фреймворки
4. Диаграммы классов разработанной системы
5. Диаграммы вариантов использования разработанной системы

Глава 4 Испытание приложение   
4.1 Unit тестирование приложения

1. Левитин А. Алгоритмы. Введение в разработку и анализ. Вильямс, 2006. 35–36 с.
2. Kona H., Burde A., Dr. Zanwar D. R. A Review of Traveling Salesman Problem with Time Window Constraint // IJIRST – International Journal for Innovative Research in Science & Technology, 2015. Vol. 2, Issue 1. P. 253– 254.
3. Tannenbaum P. Excursions in Mathematics. University of Kansas, 2011. P. 25.
4. Clausen J. Branch and Bound Algorithms – Principles and Examples. University of Copenhagen, 1999. P. 5-6.
5. Tollis I. G. Algorithms and Complexity. University of Crete, 2000. P. 140–146.
6. Applegate D., Bixby R., Chvatal V., Cook W. TSP cuts outside the template paradigm. Donet, 2000. P. 1–10.
7. Захарова Е.М., Минашина И.К. Обзор методов многомерной оптимизации // Информационные процессы, 2014. Том 14, № 3. стр. 265– 266.
8. Papadimitriou C., Vazirani U. Efficient Algorithms and Intractable Problems. Berkeley EECS, 1999. Chapter 10.
9. Goodrich M., Roberto T. Algorithm Design and Applications, Wiley, 2015. P. 513–514.
10. Nilsson C. Heuristics for the Traveling Salesman Problem. Linkoping University, 2011. P. 1–6.
11. Alsalibi B.A., Jelodar M.B., Venkat I. A Comparative Study between the Nearest Neighbor and Genetic Algorithms: A revisit to the Traveling Salesman Problem // International Journal of Computer Science and Electronics Engineering (IJCSEE), 2013. Vol. 1, Issue 1.
12. Gutin G.,Yeo A. The Greedy Algorithm for the Symmetric TSP. University of London, 2002. P. 1–2.
13. Gutin G., Karapetyan D. Lin-Kernighan Heuristic Adaptations for the Generalized Traveling Salesman Problem. Royal Holloway London University, 2010. P. 1–5.
14. Gupta R., Chauhan C., Pathak K. Survey of Methods of Solving TSP along with its Implementation using Dynamic Programming Approach. // International Journal of Computer Applications, 2012. Vol. 52, No.4. P. 1–6.
15. Basu S. Tabu Search Implementation on Traveling Salesman Problem and Its Variations: A Literature Survey. Indian Institute of Management Calcutta, 2012. P. 1–8.
16. Dorigo M., Caro G. D. Ant Algorithms for Discrete Optimization // Artificial Life, 1999. Vol. 5, No 2. P. 139–140.
17. Dorigo M., Gambardella L. M. Ant colonies for the traveling salesman problem. Université Libre de Bruxelles, 1996. P. 1–4.
18. Johnson D. S., McGeoch L. A., Rothberg E. E. Asymptotic Experimental Analysis for the Held-Karp Traveling Salesman Bound. AT&T Bell Laboratories, 1999. P. 1–2.
19. Логист. http://logist.poncy.ru/
20. Speedy Route. <https://www.speedyroute.com/>