УДК: 004.021

РАЗРАБОТКА АЛГОРИТМА СОСТАВЛЕНИЯ МАРШРУТОВ ПЕРЕДВИЖЕНИЯ ПОЛЬЗОВАТЕЛЕЙ ОБЩЕСТВЕННОГО АВТОТРАНСПОРТА ПО НАИМЕНЬШЕМУ ПУТИ

Стрельников Е.М.

Белгородский государственный национальный исследовательский университет, Россия, Белгород, e-mail: info@bsu.edu.ru

Данная статья направлена на разработку алгоритма составления маршрутов передвижения пользователей общественного автотранспорта по наименьшему пути.
Ключевые слова: алгоритм составления маршрутов, общественный транспорт, планирование маршрута

DEVELOPMENT OF AN ALGORITHM FOR DRAWING ROUTES FOR MOVING USERS OF PUBLIC ROAD TRANSPORT BY THE LEAST WAY

Strelnikov E.M.

Belgorod State National Research University, Belgorod, e-mail: info@bsu.edu.ru

This article is aimed at developing an algorithm for compiling the routes of movement of users of public vehicles on the smallest path.
Keywords: route planning algorithm, public transport, route planning

Большинство мобильных приложений, которые позволяют следить за перемещением наземного общественного транспорта в реальном времени, не имеют гибкости при построении собственного маршрута, при планировании прибытия к пункту назначения в нужное время, а также при покупке билетов. Первые две нерешенные задачи являются более актуальными, так как с ростом численности населения будет расширяться городская инфраструктура в результате чего будут появятся новые маршруты. В таком случае очень часто будут встречаться пересадки с маршрута на другой маршрут внутри города, что усложнит планирование собственного маршрута и потребует гибкости от существующих приложений. Кроме того, большинство приложений на данный момент предоставляют только мониторинг общественного транспорта и выбора собственного маршрута передвижения из предложенного списка.

На сегодняшний день планирование маршрутов передвижения гражданами, использующими общественный транспорт, становится актуальной задачей. Многие автобусы достигают остановки не по графику, что осложняет расчет времени пользователей. По городу автобусы доходят до остановки через 20 – 25 мин, а за городом это цифра увеличивается с 30 минут и до часу, что существенно увеличивает время ожидания общественного автотранспорта и повышает вероятность опоздать на маршрут.

Для решения вышеуказанной проблемы в данной работе предложен алгоритм по планированию прибытия к пункту назначения в нужное время.

Для проектирования было использовано draw.io, с помощью которого были разработаны IDEF3-диаграммы. В ходе проектирования алгоритма передвижения пользователей общественного автотранспорта, был построен алгоритм составления маршрутов передвижения по наименьшему пути.

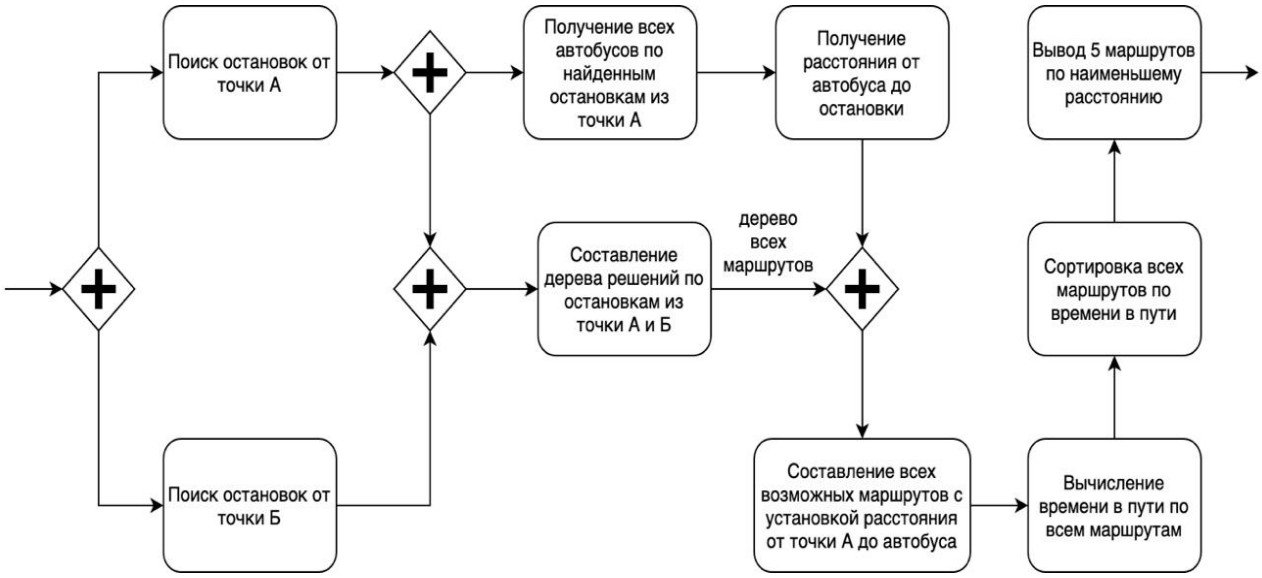

Рисунок 1 – Алгоритм составления маршрутов по наименьшему пути

Начальный процесс запускает поиск маршрутов по 2 точкам «А» и «Б». Причем, они могут располагаться на любом участке карты и на разном расстоянии. Сам поиск будет осуществляться с определенным радиусом, где центром и есть установленная точка. Далее будут получены остановки, которые попали в зону поиска, где каждая из них будет записана в свой массив. Из первого массива с остановками получаем все общественные автотранспорты и находим их расстояние до остановок. Кроме этого, мы используем данные массивы для составления дерева решений. Получив дерево всех маршрутов, для них устанавливается расстояние и вычисляется время в пути. Далее происходит сортировка маршрутов по времени в пути и вывод пяти маршрутов, которые были найдены по наименьшему расстоянию.

Алгоритм составления дерева решений может запускаться каждый раз, когда происходит изменение координат одной из двух точек «А» или «Б». При этом, предыдущие результаты могут кэшироваться для производительности. К примеру, пользователь может запускать поиск из точки «А» каждый раз в одном месте, а именно дома. В результате поиска, остановки можно будет забирать из базы данных напрямую, чтобы не нагружать систему. Потому что, сам алгоритм поиска ближайших остановок (см. рисунок 2) достаточно ресурсоемкий.

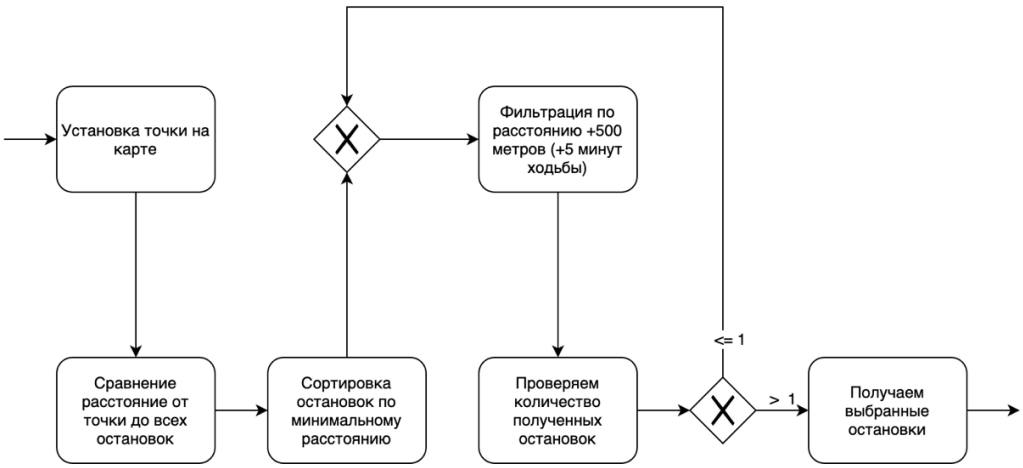

Рисунок 2 – Алгоритм поиска ближайших остановок для установленной точки

Данный алгоритм начинает свой процесс с установки точки на карте. После этого, от нее сравнивается расстояние до всех остановок, которые были получены из базы данных. Все остановки сортируются по дистанции между выбранной точкой и остановкой. После этого, фильтруются по расстоянию от 500 метров и проверяется количество полученных остановок. Если остановок будет меньше двух, то радиус поиска увеличится в 2 раза относительно установленной точки «А», что составит 1000 метров. Алгоритм поиска ближайших остановок будет выполняться рекурсивно, увеличивая радиус поиска до тех пор, пока не будет найдено более 1 остановки.

Так же будут записываться результаты выполнения для алгоритма «составление дерева решений по остановкам из точки А и Б» (см. рисунок 3).

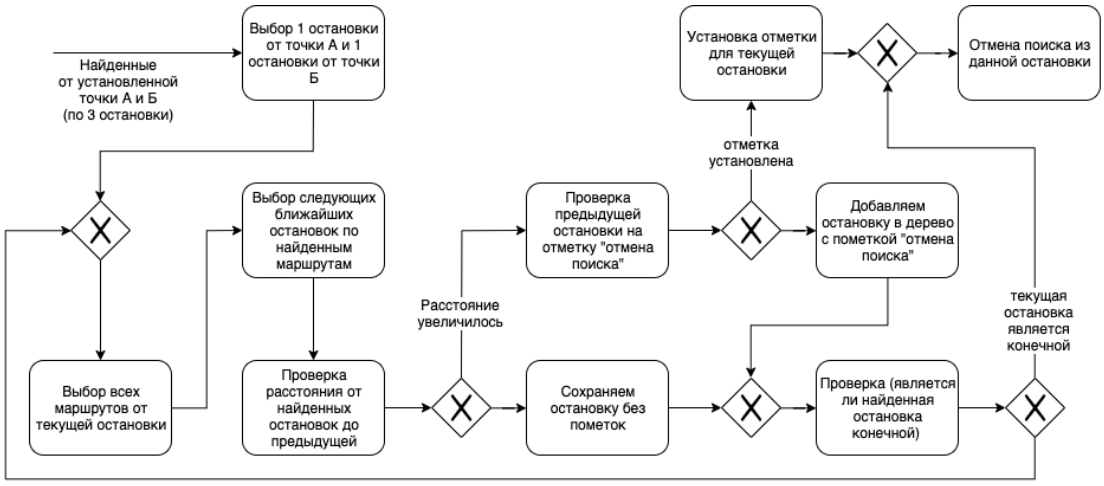

Рисунок 3 – Алгоритм составления дерева решений по остановкам из точки А и Б.

На первом этапе происходит инициализация дерева решений. Оно заполняется остановками (объектами Point) и проверяется по 2 точкам «А» и «Б». В случае, если найдены маршруты от точки «А» до точки «Б» напрямую, то алгоритм возвращает найденные маршруты. В случае если, маршруты не были найдены, то процесс переходит в следующий этап, а именно в поиск ближайших остановок по найденным маршрутам. Данный этап

предусматривает очередность выбора из двух массивов (points, B) объектов, с дальнейшим их сравнением и поиском ближайшей остановки. Найденные остановки добавляются в выбранный объект Point. Далее сам объект передается рекурсивно вниз, до тех пор, пока все его найденные объекты не будут находится вблизи конечных остановок.

Как пример результата данного алгоритма представлен на рисунке 4.

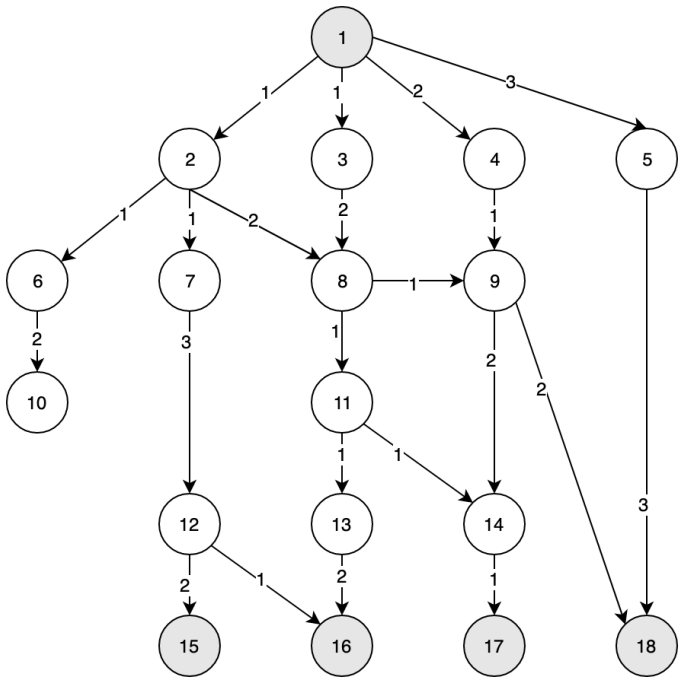

Рисунок 4 – Результат выполнения алгоритма составления дерева решений по двум точкам

В результате выполнения процесса построения дерева решений был получен граф. Само дерево решений будет сохранено в базу данных, а именно в таблицу по двум точкам для дальнейшей оптимизации. В следующий раз, когда пользователь выберет именно эти две точки, то сначала будет проверена таблица на допустимый маршрут. В случае, когда маршрут будет найден в данной таблице, все данные из ячейки value выберутся в формате json, которые будут представлять дерево решений.

Список литературы:

1. Node.js событийно-ориентированный язык программирования [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Node.js

2. About Node.js [Электронный ресурс] URL: https://nodejs.org/en/about

3. Amaksr. 2015. GPS-монитор под андроид «KidsTrack [Электронный ресурс] URL: https://habrahabr.ru/post/257443/

4. Null 2010. Онлайн-мониторинг транспорта своими руками [Электронный ресурс] URL: https://habrahabr.ru/post/99508/