ЭВРИСТИЧЕСКИЕ АЛГОРИТМЫ ПОИСКА КРАТЧАЙШЕГО ПУТИ В СЕТИ ОБЩЕСТВЕННОГО ТРАНСПОРТА

А. Ю. Лобанов

В геоинформационных сервисах существует задача поиска наиболее быстрого пути от одной точки в городе до другой с помощью маршрутов общественного транспорта. Такой путь состоит из сегментов поездок на транспорте, соединенных между пересадками, которые могут включать в себя пеший переход между остановками.

Классическим подходом к решению задачи поиска наиболее быстрого пути в сетях общественного транспорта является использование графов. По имеющейся модели транспортной сети строится взвешенный ориентированный граф, а затем применяется какой-либо известный алгоритм для поиска пути в полученном орграфе. Для поиска кратчайшего пути в таком графе можно использовать алгоритм Дейкстры [1]. Также для поиска таких путей часто используется модификация алгоритма A*, называемая ALT (A* + Landmarks + triangle inequation) [2].

Однако на практике в последнее время все чаще начинают применяться алгоритмы, которые используют тот факт, что сеть общественного транспорта сама по себе проектируется эффективным образом. Примерами таких алгоритмов являются алгоритм RAPTOR [3] и алгоритм на основе маршрутов для поиска путей в сети общественного транспорта [4].

В данной работе рассматривается ряд эвристических алгоритмов поиска кратчайшего пути в сети общественного транспорта, которые позволяют быстро искать пути с небольшим числом пересадок, удовлетворяющие следующим шаблонам:

- «Без пересадок» – данный шаблон описывает пути, которые содержат не более одного сегмента проезда на маршруте общественного транспорта;
- «С одной пересадкой» – данный шаблон описывает пути, которые содержат два сегмента проезда на объещественном транспорте;
- «Автобус + Метро + Автобус» – данный шаблон описывает пути, в которых первый и последний сегменты соответствуют использованию наземного транспорта, а все прочие сегменты между ними соответствуют использованию метро.

Чтобы оценить применимость описанных выше шаблонов путей был проведен вычислительный эксперимент. Для экспериментов использовались данные транспортной сети Нью-Йорка. Выбор этих данных обоснован тем, что Нью-Йорк имеет одну из крупнейших в мире сетей общественного транспорта, и описывающие её данные публично доступны в

формате GTFS. Кроме этого, данные для Нью-Йорка часто используются в исследованиях алгоритмов, что позволяет сравнивать результаты с другими публикациями. В рамках данного исследования пешеходные участки пути строились по прямой без учета расположения улиц и имели ограничение по длине не более 1,5 км.

В рамках эксперимента рассмотрим все возможные пары остановок. Для каждой такой пары остановок проверим существование путей, соответствующих приведенным выше шаблонам. При этом если был найден путь для шаблона «Без пересадок», то пути для шаблонов «С одной пересадкой» и «Автобус + Метро + Автобус» не рассматривались. Если же был найден путь для шаблона «С одной пересадкой», то не рассматривался шаблон «Автобус + Метро + Автобус». Аналогичным образом рассматривалось существование кратчайшего пути, удовлетворяющего данным шаблонам. Доли пар остановок, для которых пути будут удовлетворять данным шаблонам, приведены в табл. 1.

Таблица 1

Доля пар остановок, для которых путь будет удовлетворять шаблону

Шаблон пути	Существует путь	Существует кратчайший путь
Без пересадок	0,470	0,140
С одной пересадкой	0,432	0,300
Автобус + Метро + Автобус	0,052	0,215
Остальные	0,046	0,345

Результаты эксперимента показывают, что в 95,4 % всех случаев будет существовать путь, удовлетворяющий данным шаблонам, а в 65,5 % случаев будет существовать кратчайший путь, удовлетворяющий данным шаблонам.

Поиск путей, соответствующих приведенным выше шаблонам, можно производить эффективно с помощью подхода «Meet-in-the-middle». Для нахождения пути «Без пересадок» рассмотрим все остановки, достижимые пешком из начальной точки. Далее построим список маршрутов транспорта, на которые можно сесть на одной из данных остановок. Аналогичным образом рассмотрим все остановки, достижимые пешком из конечной точки, и построим список маршрутов транспорта, с помощью которых можно высадиться на одной из этих остановок. Далее нужно найти маршруты транспорта, присутствующие в обоих списках. С помощью хэш-таблицы это может быть сделано за линейное время от суммарной длины списков. Если в некотором таком маршруте остановка, достижимая из начальной точки, встречается не позже, чем остановка, достижимая из конечной точки, то этот маршрут транспорта дает путь, соответствующий шаблону «Без пересадок».

Для поиска путей с одной пересадкой нужно получить список остановок, достижимых из начальной и конечной точек с помощью одной поездки на транспорте. Для этого можно использовать списки маршрутов транспорта, полученные в рамках поиска пути для шаблона «Без пересадок». Для каждого маршрута из списка маршрутов, достижимых из начальной точки нужно выписать остановки, следующие после остановки посадки. Аналогично для списка маршрутов, достижимых из конечной точки, нужно выписать остановки, идущие до остановки высадки. Далее в этих списках нужно найти пары остановок, расстояние между которыми не превышает допустимой длины пешего участка пути. Каждая такая пара остановок будет давать путь, соответствующий шаблону. Для поиска такой пары точек использовался пространственный индекс в виде сетки с шагом равным среднему расстоянию между остановками транспорта. Такой индекс позволяет осуществлять поиск точки в заданном радиусе за константное время. В результате трудоемкость поиска путей, соответствующих шаблону «С одной пересадкой», будет линейной от суммарного количества остановок в рассматриваемых маршрутах.

Для эффективного построения путей, соответствующих шаблону «Автобус + Метро + Автобус», использовался тот факт, что количество станций метро в транспортной сети существенно меньше, чем общее количество остановок. На момент написания этой статьи, самый крупный по числу станций метрополитен в мире – метрополитен Нью-Йорка – содержит 468 станций. Таким образом, кратчайшие пути между всеми парами станций метрополитена можно просчитать предварительно с помощью алгоритма Дейкстры [1].

Для поиска путей, соответствующих шаблону «Автобус + Метро + Автобус», поместим станции метро в пространственный индекс. Рассматривая описанные в алгоритме для шаблона «С одной пересадкой» списки остановок, с помощью пространственного индекса будем находить станции метро, которые достижимы пешком от данных остановок. Рассмотрев все такие станции метро, можно получить кратчайший путь, соответствующий шаблону «Автобус + Метро + Автобус», используя предварительно просчитанные пути для каждой пары станций метро.

Описанные в данной работе эвристические алгоритмы могут быть использованы для поиска пути сами по себе, но при этом для 34,5 % пар точек путь будет не найден, либо найденный путь не будет кратчайшим. Однако даже если найденный путь не будет кратчайшим, то он все равно может оказаться удобным для практического использования, так как содержит небольшое число пересадок. Кроме этого, величина стоимости пути, полученная с помощью данных эвристических алгоритмов, может

быть использована как верхняя оценка стоимости кратчайшего пути для алгоритма ALT [2].

В табл. 2 приведены результаты сравнения производительности программных реализаций алгоритмов. В таблицу включены результаты, полученные для данных эвристических алгоритмов как самих по себе, так и в качестве вспомогательных совместно с алгоритмом ALT. Результаты времени работы алгоритма для шаблона «С одной пересадкой» не приведены отдельно, так как алгоритм для шаблона «Автобус + Метро + Автобус» включает в себя пути и для шаблона «С одной пересадкой». Тестирование проводилось на компьютере, снабженном процессором Intel Core i5-3570K, работающем на частоте 4,3GHz. В качестве тестовых запросов для поиска пути использовался список из 1000000 пар остановок транспорта, выбранных случайным образом.

Таблица 2

Сравнение производительности алгоритмов

Название алгоритма	Суммарное время, с	Квантили на один запрос, мс			
		0,5	0,75	0,9	1
Алгоритм Дейкстры	45970	47	69	80	96
Алгоритм ALT	9736	9	12	17	42
Беспересадочный	66	1	1	1	2
1 пересадка + метро	3658	2	4	8	37
ALT + Беспересадочный	6895	6	10	15	35
ALT + 1 пересадка + метро	5318	4	6	11	42

Приведенные в табл. 2 результаты показывают, что описанные в данной работе эвристические алгоритмы показывают более высокую производительность, чем алгоритм Дейкстры или алгоритм ALT, а также позволяют увеличить производительность алгоритма ALT.

Литература

1. *Schulz F., Wagner D., Weihe K.* Dijkstra's Algorithm On-Line: An Empirical Case Study from Public Railroad Transport // Journal of Experimental Algorithmics. – 2000. – Vol. 5, № 12.
2. *Delling D., Wagner D.* Landmark-Based Routing in Dynamic Graphs. // In Proc. of the 6th Workshop on Experimental Algorithms (WEA'07). – Springer, 2007. – P. 52–65.
3. *Delling D., Pajor Th., Werneck R.F.* Round-Based Public Transit Routing // In Proc. of the 14th Meeting on Algorithm Engineering and Experiments (ALENEX'12) – 2012. – P.130–140.
4. *Лобанов А.Ю.* Алгоритм на основе маршрутов для поиска путей в сети общественного транспорта // Информатизация образования. – 2013. – №1(70). – С. 46–59.