Математика. Прикладная математика

УДК 519.854.2

Максимова Надежда Николаевна
Амурский государственный университет
г. Благовещенск, Россия
E-mail: knnamursu@mail.ru
Maksimova Nadezhda Nikolaevna
Amur State University
Blagoveshchensk, Russia
E-mail: knnamursu@mail.ru
Колтунов Николай Сергеевич
Амурский государственный университет
г. Благовещенск, Россия
E-mail: knnamursu@mail.ru
Koltunov Nikolay Sergeevich
Amur State University
Blagoveshchensk, Russia
E-mail: knnamursu@mail.ru

ПОИСК ОПТИМАЛЬНОГО КОЛЬЦЕВОГО МАРШРУТА НА КАРТЕ Г. БЛАГОВЕЩЕНСКА С ИСПОЛЬЗОВАНИЕМ АЛГОРИТМА ИМИТАЦИИ ОТЖИГА

SEARCH FOR AN OPTIMAL RING ROUTE ON THE BLAGOVESHCHENSK MAP USING THE ANNEALING SIMULATION ALGORITHM

Аннотация. В статье рассматривается применение метода имитационного отжига для исследования задачи построения кольцевого маршрута по реальным данным на примере карты г. Благовещенска.

Abstract. The article discusses the application of the simulated annealing method for the study of the constructing a ring route problem of large dimension according to real data on the example of a map of Blagoveshchensk.

Ключевые слова: природные вычисления, задача коммивояжера, математическая модель задачи коммивояжера, алгоритм имитационного отжига, ППП Matlab, OSRM, OSM.

Key words: natural computing, traveling salesman problem, algorithm, mathematical model of the traveling salesman problem, simulation annealing method, Matlab, OSRM, OSM.

DOI: 10/22250/jasu.1

Введение

Предметом исследования в данной статье является одна из самых известных и важных задач транспортной логистики, классическая задача дискретной оптимизации – задача коммивояжера (задача построения кольцевого маршрута, англ. «Travelling salesman problem», TSP) [1]. Суть задачи – построить путь наименьшей длины, проходящий через заранее заданные пункты (города) по одному разу и заканчивающийся в начальной точке. Для решения задач малой размерности (при малом коли-

честве пунктов) можно применить аналитические методы (например, венгерский алгоритм или метод ветвей и границ). Данные алгоритмы позволяют найти точное решение, но при увеличении количества точек число шагов метода, а следовательно и временны́е затраты, будут значительно возрастать.

В настоящее время для решения комбинаторных задач применяются методы из нового научного направления «Природные вычисления» (Natural Computing). К таким алгоритмам относятся генетические алгоритмы, эволюционное программирование, нейросетевые вычисления, ДНК-вычисления, клеточные автоматы, алгоритмы роя (муравьиные алгоритмы, пчелиные алгоритмы) и др. [2]. В основе подобных методов лежит симуляция биологических процессов, в частности механизмов, связанных с процессом эволюции [3].

Природные алгоритмы успешно применяются при решении комбинаторных задач – задачи выбора маршрута, задачи коммивояжера, задачи о восьми ферзях, задачи о ранце, задачи раскроя, задачи о назначениях, задачи составления расписания [4].

В работе для решения поставленной задачи применяется один из таких алгоритмов – метод имитационного отжига. Для задачи с произвольной матрицей расстояний (со случайным выбором координат городов) алгоритм тестируется с целью определения оптимальных режимов и параметров, затем метод применяется для решения задачи построения кольцевого маршрута на примере реальных данных по адресам г. Благовещенска.

Описание алгоритма имитации отжига

Впервые описанный в [5] алгоритм имитации отжига представляет собой метаэвристический алгоритм, который находит глобальный минимум целевой функции.

Алгоритм сводится к имитации физического процесса, происходящего при кристаллизации вещества (например, при отжиге металлов). Предполагается, что атомы уже выстроились в кристаллическую решетку, но допускаются переходы отдельных атомов из одной ячейки в другую. Весь процесс протекает при постепенно уменьшающейся температуре. Переход атома из одной ячейки в другую возможен с некоторой вероятностью, и, чем меньше температура, тем меньше будет вероятность такого перехода. Устойчивая кристаллическая решетка соответствует минимуму энергии атомов. Значение энергии атомов принимается за целевую функцию. Атом либо переходит в состояние с меньшим значением этой функции, либо остается на месте [6-8].

Из описанного выше можно сформулировать следующее:

1) совокупность позиций всех атомов – это состояние системы, где каждому состоянию соответствует определенный уровень энергии;

2) цель метода – привести систему в состояние с наименьшей энергией.

Рассмотрим алгоритм для решения задачи коммивояжера. Пусть имеется n городов с заданными координатами (x_i, y_i). По известным координатам можно вычислить расстояние между каждой парой городов. Множество S представляет собой векторы, координатами которых являются всевозможные перестановки натуральных чисел от 1 до n. Добавляя к каждому такому вектору $(n+1)$-й элемент, равный первому элементу, будем получать всевозможные кольцевые маршруты. Каждому вектору $s \in S$ будет соответствовать «энергия», равная длине $f(s)$ соответствующего кольцевого маршрута.

Тогда алгоритм может быть описан 5 этапами, которые представлены далее.

Этап I. Выбирается стартовая точка $s^0 \in S$ и параметр T_0.

В качестве начального вектора s^0 будем брать случайный вектор из S. Параметр T_0 будет соответствовать начальному значению температуры (при реализации алгоритма положили начальное значение температуры равным $T_0 = 1$).

Этап II. На (k+1)-й итерации выбирается некоторое пробное решение – вектор $q^{k+1} \in S$ – в соответствии с заданным правилом $D(\cdot, s^k)$.

Новое решение q^{k+1} генерируется из текущего s^k путем перестановки двух случайных городов.

Этап III. Случайно задается значение $p_k \in [0,1]$ согласно закону равномерного распределения. Следующее приближение к решению выбирается по правилу:

$$s^{k+1} = \begin{cases} q^{k+1}, & \textit{если} \quad p_k \le A(s^k, q^{k+1}, T_k), \\ s^k, & \textit{иначе.} \end{cases} \tag{1}$$

Основной особенностью алгоритма является его способность на каждом шаге выбирать такое решение, которое увеличивает значение целевой функции, что позволяет ему избегать локальных минимумов [9].

Одним из вариантов функции, лежащей в основе критерия, – принятие решения о том, брать ли новое значение в качестве решения, может ли быть выбрана функция Метрополиса [4]:

$$A(s,q,T) = \min\left[1, \exp\left(-\frac{f(q)-f(s)}{T}\right)\right]. \tag{2}$$

Функция Метрополиса всегда принимает положительное решение о принятии очередного q^{k+1} в качестве нового приближения в случае, если оно уменьшает значение f, т.е. $f(q^{k+1}) \le f(s^k)$. В то же время с определенной вероятностью может быть принято решение, увеличивающее значение целевой функции. Это свойство позволяет алгоритму не остаться в локальном минимуме. Вероятность принятия положительного решения в случае увеличения значения целевой функции зависит от параметра T, представляющего «температуру» системы. Легко заметить, что в случае, когда $\{T_k\} \to 0$ $A(s, q, T) \to 0$ при $k \to \infty$. Таким образом, вероятность принятия нового приближения, увеличивающего значение f, уменьшается с ростом количества итераций.

Другим возможным вариантом функции может стать так называемая функция Баркера [10]:

$$A(s,q,T) = \left[1 + \exp\left(\frac{f(q)-f(s)}{T}\right)\right]^{-1} \tag{3}$$

В работе реализована функция принятия новых приближений в виде функции Метрополиса [2], поскольку вычислительные опыты показали, что функция Баркера не сильно отличается результатом.

Этап IV. Пересчитывается параметр (температура) $T_{k+1} = U(T_0, k)$.

Определять температуру нужно в каждой точке индивидуально, функция определения температуры должна быть монотонно убывающей.

Схема Больцмана [5], имеющая вид

$$T_{k+1} = T_0 / \ln(1+k), \tag{4}$$

считается для данного алгоритма исторически первой схемой охлаждения. Основной недостаток схемы Больцмана – медленное убывание температуры, что приводит к медленной сходимости алгоритма.

Альтернативной схемой охлаждения является схема Коши [8]:

$$T_{k+1} = T_0 / (1+k). \tag{5}$$

В литературе встречается также экспоненциальный закон понижения температуры, определяемый формулой [7]:

$$T_{k+1} = T_0 \cdot e^{-c(k+1)}, \tag{6}$$

где $c > 0$ – некоторая константа, и геометрическая функция [7]

$$T_{k+1} = \alpha \cdot T_k, \tag{7}$$

где $\alpha \in (0,1)$.

В реализации алгоритма для понижения температуры используется схема Коши [8].

Этап V. Проверяется условие останова итерационного процесса. Если оно не выполняется, устанавливается $k = k + 1$ и происходит возврат к *этапу II*.

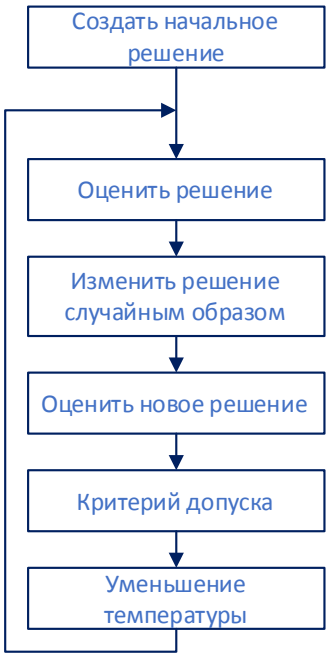

Рис. 1. Блок-схема алгоритма.

В качестве критерия остановки итераций используется достижение заданного нижнего порога температуры T_{min}, которая рассчитывается по формуле:

$$T_{min} = T_{max} / k_{max}.$$

Работу алгоритма имитации отжига можно представить в виде упрощенной блок-схемы, представленной на рис. 1.

Поиск оптимального кольцевого маршрута на карте г. Благовещенска

Имитационный отжиг был протестирован в «реальных условиях», т.е. к точкам были привязаны реальные адреса г. Благовещенска. Адреса содержат название улицы, номер здания и соответствующая каждому адресу широта и долгота.

На языке Python 3.7 в программной среде Microsoft Visual Studio была разработана программа. Результатом работы данной программы стала матрица расстояний между выбранными точками на карте Благовещенска. Расстояния от одного адреса до другого и соответствующая каждому адресу широта и долгота были взяты из ресурса 2GIS [11].

Для иллюстрации работы алгоритма были выбраны следующие адреса: Институтская, 15; Дьяченко, 4; Комсомольская, 26; Ленина, 100; Красноармейская, 161; Красноармейская, 139/1; Центральная, 12; Строителей, 107к1; Трудовая, 182; Мухина, 114; Кантемирова, 6/2; Кольцевая, 40; Театральная, 276; Калинина, 130/2; Свободная, 33; Горького, 153; Студенческая, 43/3; Амурская, 151; Ломоносова, 154; Горького, 202; Ленина, 177; Перспективная, 53; Богдана Хмельницкого, 42/1; Зейская, 235; Амурская, 186/1; Релочный пер., 3; Амурская, 99; Чайковского, 90; Соколовская, 39; Чайковского, 175; Дьяченко, 9/1; Кантемирова, 15; Кантемирова, 8/2; 50 лет Октября, 208; Тополиная, 51; Студенческая, 34/5; Амурская, 261; Игнатьевское шоссе, 14/7; Политехническая, 52; Зеленая, 4; Заводская, 2; Лазо, 45(53); Речной пер., 11; Василенко, 20/5; Пушкина, 183/1; Октябрьская, 236; Свободная, 31; Ленина, 283/3; 50 лет Октября, 201/1; Студенческая, 28/1.

Полученные данные сохраняются в отдельных файлах, которые впоследствии используются в виде матрицы расстояний в программном коде Matlab. Так как было три варианта точек (10, 30, 50), соответственно матриц будет тоже три (10×10, 30×30, 50×50).

С помощью ранее реализованного имитационного отжига в Matlab осуществляется поиск оптимального пути по имеющейся матрице расстояний. После расчетов в данной программе выводится массив с порядком следования по адресам (для удобства все адреса пронумерованы).

По полученным порядкам следования по адресам и соответствующим им координатам (широта и долгота) формируются специальные ссылки, ведущие на сайт ресурса (https://map.project-osrm.org/), который работает на основе системы OSRM, где строятся соответствующие маршруты.

Open Source Routing Machine (OSRM) – это реализация на языке C++ высокопроизводительного механизма маршрутизации для кратчайших путей в дорожных сетях [13]. OSRM сочетает в себе сложные алгоритмы маршрутизации с открытыми и бесплатными данными дорожной сети проекта Open Street Map (OSM). Вычисление кратчайшего пути в сети континентального размера может занять, как правило, до нескольких секунд. Программное обеспечение, используя реализацию иерархий сжатия, может вычислять и выводить кратчайший путь между любым источником и пунктом назначения в течение нескольких миллисекунд, в результате вычисление чистого маршрута занимает го-

раздо меньше времени. Бóльшая часть усилий затрачивается на аннотирование маршрута и передачу геометрии по сети.

Выбор данного Интернет-ресурса был обоснован тем, что в нем можно строить маршруты с неограниченным количеством точек (у карт от Google и Yandex стоит ограничение на 10 точек). И повторимся: данный ресурс используется только для построения маршрута на карте, так как порядок адресов оптимального маршрута уже рассчитан методом имитационного отжига.

В табл. 1 отображены основные показатели всех проделанных расчетов и опытов, при поставленной реальной задаче. В таблице приняты обозначения: Iteration – количество затраченных итераций по поиску оптимального маршрута; Matlab – дистанция (км) маршрута, рассчитанная в программной среде Matlab; OSM – дистанция (км) маршрута, рассчитанная в открытых данных дорожной сети (на основе реальных картографических данных).

Таблица 1

Данные по маршрутам. Метод имитационного отжига

	Варианты проходов по итерациям					
Кол-во адресов	**№ 1**		**№ 2**		**№ 3**	
10	Iteration: Matlab: OSM:	10 51.3 51.1	Iteration: Matlab: OSM:	50 36.4 36.1	Iteration: Matlab OSM:	100 32.8 34.3
30	Iteration: Matlab: OSM:	50 110.8 112.7	Iteration: Matlab: OSM:	100 81.5 80.6	Iteration: Matlab: OSM:	500 64.0 65.8
50	Iteration: Matlab: OSM:	100 182.2 187.9	Iteration: Matlab: OSM:	500 121.9 122.8	Iteration: Matlab: OSM:	700 110.5 108.2

Ниже представлены рисунки с оптимальными маршрутами для каждого из списка адресов (рис. 2-4).

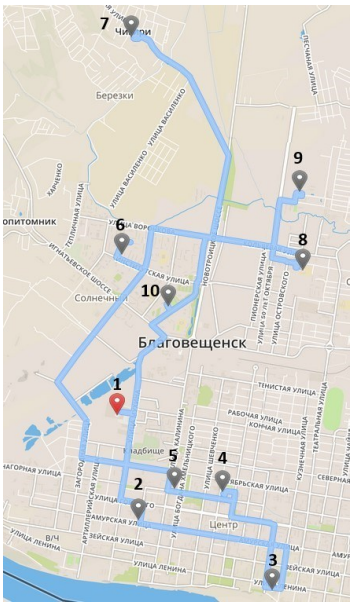

Рис. 2. Оптимальный маршрут при 10 точках.

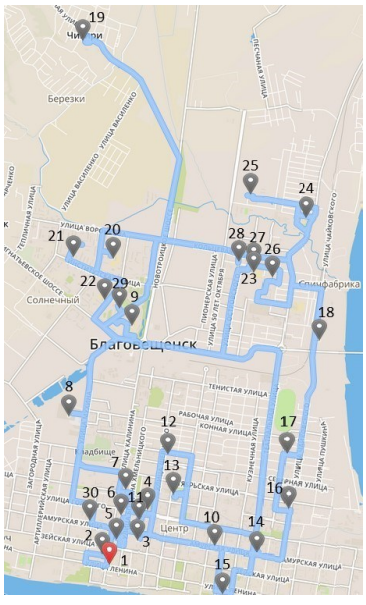

Рис. 3. Оптимальный маршрут при 30 точках.

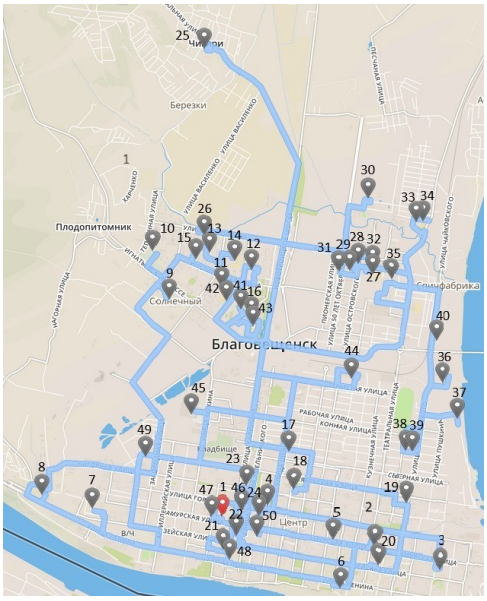

Рис. 4. Оптимальный маршрут при 50 точках.

Далее были выбраны лучшие результаты по каждому варианту, их данные внесены в табл. 2. Также в эту таблицу были добавлены лучшие результаты по муравьиному алгоритму, полученные в ранее опубликованных работах [14].

Таблица 2

Маршрутные данные двух алгоритмов (методов)

	Варианты проходов по итерациям			
Кол-во адресов	**Муравьиный алгоритм**		**Имитационный отжиг**	
10	Iteration: Matlab: OSM:	20 32.5 км 32.1 км	Iteration: Matlab: OSM:	100 32.8 36.1
30	Iteration: Matlab: OSM:	40 46.4 км 45.1 км	Iteration: Matlab: OSM:	500 64.0 80.6
50	Iteration: Matlab: OSM:	60 70.2 72.6	Iteration: Matlab: OSM:	700 110.5 108.2

Заключение

Сравнивая полученные показатели оптимальных маршрутов по двум алгоритмам, можно сделать вывод, что алгоритм имитации отжига в сравнении с муравьиным алгоритмом с поставленной задачей справляется хуже, особенно при значительном количестве городов. Большое количество итераций данного метода не гарантирует получения оптимального маршрута, так как имитационный отжиг имеет вероятностную природу. Случай может распорядиться так, что поиск оптимального решения «застрянет» в локальном минимуме**,** не дойдя до глобального, чего нельзя сказать о муравьином алгоритме. Главный его плюс заключается в поиске глобального оптимума. При бесконечном числе итераций вероятность нахождения глобального лучшего стремится к единице. Вопрос только в затрате времени на поиск оптимального маршрута.

1. Мудров, В.И. Задача о коммивояжере. – М.: Знание, 1969. – 62 с.

2. Dorigo, M. Optimization, learning and natural algorithms. – Ph. D. Thesis, Politecno di Milano, Italy – 1992.

3. Карпенко, А.П. Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой: учебное пособие. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2014. – 446 с.

4. Штовба, С.Д. Муравьиные алгоритмы // ExponentaPro. Математика в приложениях. – 2003. – № 4 – С. 70-75.

5. Kirkpatrick, S. Optimization by Simulated Annealing / S. Kirkpatrick, C.D. Gelatt Jr, M.P. Vecchi // Science. – 1982. – Vol. 220. – P. 671-680.

6. Джонс, М.Т. Программирование искусственного интеллекта в приложениях / пер. с англ. – М.: ДМК Пресс, 2011. – 312 с.

7. Лопатин, А. С. Метод отжига // Стохастическая оптимизация в информатике. – СПб.: Изд-во СПбГУ, 2005. – Вып. 1. – С. 133-149.

8. Szu, H.H. Fast Simulated Annealing / H.H. Szu, R.L. Hartley // Physical Letters A. 122, 1987 – P. 157-162.

9. Устинов, С.М. Вычислительная математика / С.М. Устинов, В.А. Зимницкий. – СПб.: БХВ-Петербург, 2009. – 336 с.

10. Gendreau, M. Handbook of Metaheuristics (International Series in Operations Research and Management Science) / M. Gendreau, J.Y. Potvin. – Springer, 2010. – 675 p.

11. 2gis.ru: 2ГИС. Карта города Благовещенска. – URL: https://2gis.ru/blagoveshensk (дата обращения: 15.01.2019).

12. Map.project-osrm.org: OSRM. Карта города Благовещенска. – URL: https://map.project-osrm.org/ – (дата обращения: 15.01. 2020).

13. Колтунов, Н.С. Построение оптимального кольцевого маршрута на карте г. Благовещенска // Материалы XX региональной научно-практической конференции «Молодежь XXI века: шаг в будущее». – Благовещенск, 2019. – Т. 3. – С. 164-165.