УДК 519.176+004.421

О решении задачи маршрутизации транспорта с помощью подвижного генетического алгоритма

Д. О. Сидоренко, А. Ю. Городилов
Пермский государственный национальный исследовательский университет; г. Пермь, Россия
aleksey.gorodilov@gmail.com; AuthorID 537865
dima.sidorenk@yandex.ru

Описан подход к решению задачи маршрутизации транспорта на основе подвижного генетического алгоритма. Подвижные генетические алгоритмы отличаются от классических более гибкой схемой кодирования решений, что актуально для задач со сложной структурой решения. В статье приведена математическая постановка задачи. Авторами предложено два варианта кодирования особей, а также алгоритм пересчета вероятностей, формирующих хромосому в подвижном генетическом алгоритме. Выполнено сравнение предложенного подхода с другими существующими подходами решения задачи маршрутизации транспорта. Проведенные исследования позволяют утверждать, что для решения поставленной задачи применение подвижных генетических алгоритмов возможно. Получаемые результаты корректны, однако при больших объемах данных алгоритм работает слишком медленно, получаемое решение оказывается значительно хуже решения, получаемого классическим генетическим алгоритмом. В статье рассматриваются возможные варианты решения возникших проблем. Данная статья является расширенной версией работы, представленной на конференции "Математика и междисциплинарные исследования 2021" [1].

Ключевые слова: *подвижный генетический алгоритм; задача маршрутизации транспорта*

Поступила в редакцию 25.10.2021, принята к опубликованию 12.11.2021

On solving the vehicle routing problem using a fluid genetic algorithm

D. O. Sidorenko, A. Yu. Gorodilov
Perm State University; Perm, Russia
aleksey.gorodilov@gmail.com; AuthorID: 537865
dima.sidorenk@yandex.ru

Abstract. The article describes an approach to solving the vehicle routing problem based on a fluid genetic algorithm. Fluid genetic algorithms differ from classical ones by a more flexible coding scheme for solutions, which is relevant for problems with a complex solution structure. The article presents a mathematical formulation of the problem. The authors proposed two variants of encoding individuals, as well as an algorithm for recalculating the probabilities that form a chromosome in a fluid genetic algorithm. The proposed approach is compared with other existing approaches. The conducted research suggests that the use of fluid genetic algorithms to solve the vehicle routing problem is possible. The obtained results are correct, however, the algorithm works too slowly on a big data, and the resulting solution turns out to be significantly worse than the solution obtained by the classical genetic algorithm. The article discusses possible solutions to the problems encountered. This paper is an expanded version of the work presented at the conference "Mathematics and Interdisciplinary Research 2021" [1].

Keywords: *fluid genetic algorithm; vehicle routing problem*

Received 25.10.2021, accepted 12.11.2021

DOI: 10.17072/1993-0550-2021-4-43-48

Введение

На российском рынке логистических услуг большая часть от всего объема рынка логистики приходится на транспортную логистику, поскольку на территории России имеется множество автомобильных (841 тыс. км), железных (86 тыс. км) дорог и воздушных путей (800 тыс. км). В связи с этим управление транспортными потоками является важнейшим элементом логистики, при этом по некоторым оценкам применение современных подходов позволит снизить общие экономические издержки в среднем на 15–35 %, а транспортные расходы – примерно на 25 % [2]. При планировании поставок возникает задача построения оптимального маршрута для набора транспортных средств, которые должны посетить множество заданных объектов.

Например, для снабжения сети магазинов со склада с помощью имеющихся автомобилей необходимо построить для них маршруты передвижения, оптимизировав суммарное преодолеваемое расстояние (для минимизации расходов на транспортировку) и/или время доставки товаров.

Такая задача хорошо известна в теории графов и называется *задачей маршрутизации транспорта.* Задача является NP-трудной, поэтому на практике для ее решения применяют различные приближенные и эвристические алгоритмы, в том числе генетические алгоритмы (ГА) [3].

Исследования, касающиеся генетических алгоритмов, ведутся уже сравнительно давно, но и в настоящее время развиваются достаточно активно. В частности, в последние годы появилось такое новое направление, как подвижные генетические алгоритмы [4–6]. Учитывая специфику задачи маршрутизации транспорта и сложную структуру, описывающую решение задачи, интересным представляется исследование возможности построения подвижного генетического алгоритма для решения этой задачи.

Таким образом, цель данной статьи – исследовать возможность решения задачи маршрутизации транспорта с помощью подвижного генетического алгоритма, проанализировать скорость работы такого алгоритма, качество получаемого решения, а также сравнить его с другими существующими подходами.

1. Постановка задачи

Для начала формализуем решаемую задачу в терминах теории графов.

Пусть

$V = \{v_0, v_1, \ldots, v_N\}$ – множество вершин, где v_0 – место начального расположения всех транспортных средств (депо или склад), $v_1, \ldots, v_N$ – цели, которые необходимо посетить;

$C = \{c_{ij}\}$, $i, j \in [0, N]$ – квадратная матрица расстояний между вершинами, имеющая размерность $N+1$, где c_{ij} – расстояние между вершинами v_i и v_j;

M – количество имеющихся транспортных средств;

$R = \{R_i\}$, $i \in [1, M]$ – множество маршрутов транспортных средств, где $R_i = (j_{i,0}, j_{i,1}, \ldots, j_{i,k})$ – маршрут i-го транспортного средства, представляющий собой последовательность номеров посещенных вершин, в которой первый и последний элементы равны нулю ($j_{i,0} = j_{i,k} = 0$), что означает, что маршрут начинается и заканчивается в точке начального расположения;

$C(R_i) = c_{j_{i,0}, j_{i,1}} + c_{j_{i,1}, j_{i,2}} + \ldots + c_{j_{i,k-1}, j_{i,k}}$ – длина маршрута, равная сумме расстояний между каждой парой соседних вершин в маршруте R_i.

Входными данными в задаче является матрица расстояний C и количество транспортных средств M. Решением задачи является такое множество маршрутов R, что все цели оказываются посещенными, а суммарная длина всех маршрутов является минимальной.

Заметим, что посещение одной цели несколькими транспортными средствами ведет только к увеличению общей длины маршрутов, поэтому будем считать, что каждая вершина входит ровно в один маршрут. Таким образом, на искомое множество R накладывается ограничение: для любой вершины v_i, $i \in [1, N]$, существует единственное j такое, что $i \in R_j$. Оптимизируемая величина определяется следующим образом:

$$F = \sum_{j=1}^{M} C(R_j) \to \min . \qquad (1)$$

Другим вариантом оптимизации в сформулированной задаче является минимизация не суммарной длины маршрутов, а мак-

симальной из длин, что будет соответствовать оптимизации времени на доставку всех товаров. В этом случае оптимизируемая величина определяется так:

$$F' = \max_{j\in[1,M]} C(R_j) \to \min . \qquad (2)$$

2. Подвижный генетический алгоритм

В классическом генетическом алгоритме, описанном в [3], каждое решение кодируется одной хромосомой, представляющей собой перестановку на множестве чисел от 1 до *N*. Такая перестановка определяет последовательность посещения вершин, но не содержит разделения на маршруты. Решить задачу оптимального разбиения последовательности на маршруты можно, например, методом динамического программирования за время $O(N \cdot M)$. Таким образом, хромосома в таком алгоритме не кодирует явно одно допустимое решение, а является, по сути, промежуточным представлением, позволяющим впоследствии построить решение за полиномиальное время.

В разработанном подвижном ГА применен аналогичный подход с использованием промежуточного представления, однако хромосома хранит не саму перестановку, а предрасположенность к превращению в ту или иную перестановку. Рассмотрим два варианта кодирования.

2.1. Первый вариант кодирования

В первом варианте перестановка представляется в виде кода факториального представления ее порядкового номера. Для получения такого кода каждый элемент перестановки необходимо заменить его порядковым номером в упорядоченном списке чисел, составленных из элементов перестановки, чьи позиции не меньше текущего.

Например, кодом перестановки (4, 5, 1, 3, 2) является код (4, 4, 1, 2, 1). У данного кодирования есть два преимущества: простота кодирования и раскодирования, а также возможность использования стандартного одноточечного оператора кроссинговера. Элементы кода перестановки обладают следующим свойством:

$$MaxVal_i \le N - i + 1,\ 1 \le i \le N \qquad (3)$$

где i – позиция в коде перестановки, а $MaxVal_i$ – максимальное значение элемента на позиции i.

Хромосома в генетическом алгоритме будет представлять собой матрицу $P = \{p_{ij}\}$, $i, j \in [1, N]$, где p_{ij} – вероятность того, что ген на позиции i будет равен j. Свойство (3) позволяет хранить хромосому в виде треугольной матрицы. Кроме того, для вероятностей, очевидно, должно выполняться условие, что сумма всех вероятностей для каждой позиции равна 1, то есть

$$\forall i: \sum_{j=1}^{N-i+1} p_{ij} = 1. \qquad (4)$$

При вычислении приспособленности вначале матрица вероятностей трансформируется в код перестановки (i-й элемент кода принимает значение j с вероятностью p_{ij}). Затем по коду строится сама перестановка. Наконец, приспособленность особи вычисляется на основе формул (1) или (2). Поскольку в генетических алгоритмах большей приспособленности должны соответствовать более хорошие решения, приспособленностью особи будем считать значение, обратное к F (или F').

При выполнении генетических операторов они применяются не только к самой хромосоме (матрице вероятностей), но и к соответствующему коду перестановки. После выполнения генетического оператора вероятности для каждой позиции $i<N$ пересчитываются по следующему алгоритму.

Пусть в коде перестановки на позиции i после выполнения оператора находится значение j. Тогда элемент матрицы p_{ij} необходимо увеличить, а все остальные ненулевые элементы i-й строки уменьшить. Вначале определяется предельная величина изменения

$$D_i = \max(p_{ij} + \eta_{ind}, 1 - \eta_{DR}) - p_{ij}, \qquad (5)$$

где η_{ind} – индивидуальная скорость обучения, η_{DR} – коэффициент разнообразия (гиперпараметры подвижного ГА).

Далее определяется предельная величина уменьшения каждого элемента:

$$d_i = \frac{D_i}{cnt_i - 1}, \qquad (6)$$

где cnt_i – это количество ненулевых элементов в i-й строке.

Затем все остальные (находящиеся не в столбце j) ненулевые элементы i-й строки матрицы уменьшаются на величину

$$\begin{matrix} \delta_{ik} = p_{ik} - \min(p_{ik} - d_i, \eta_{DR}), \\ k = 1..N - i + 1, k \neq j \end{matrix}. \quad (7)$$

Наконец, элемент p_{ij} увеличивается на величину

$$\Delta_i = \sum_{k=1..N-i+1, k \neq j} \delta_{ik} . \quad (8)$$

Такой алгоритм пересчета гарантирует сохранение свойства (4) для матрицы вероятностей.

Рассмотрим пример. Пусть N=4, $\eta_{ind} = 0.12, \eta_{DR} = 0.08$, хромосома непосредственно после применения оператора имеет вид $P = \begin{pmatrix} 0.1 & 0.3 & 0.4 & 0.2 \\ 0.5 & 0.2 & 0.3 & \\ 0.4 & 0.6 & & \\ 1 & & & \end{pmatrix}$, а соответствующий код – (3, 1, 2, 1). Тогда для позиции i=1 имеем: j=3, D_1=0.12, d_1=0.04, $\delta_{11} = 0.02, \delta_{12} = 0.04, \delta_{14} = 0.04, \Delta_1 = 0.1$ и после корректировки первая строка матрицы P примет вид (0.08, 0.26, 0.5, 0.16). Для остальных позиций алгоритм аналогичен.

2.2. Второй вариант кодирования

Во втором варианте код перестановки не используется. Хромосома так же представляет собой матрицу вероятностей $P = \{p_{ij}\}$, $i, j \in [1, N]$, но p_{ij} здесь означает вероятность того, что i-й элемент самой перестановки равен значению j. При таком подходе возникает ряд проблем. Например, возникает необходимости следить за корректностью перестановки. Данная проблема решается следующим образом. При формировании перестановки на основе хромосомы будем хранить список уже использованных номеров.

Тогда значение в текущей позиции не может быть равно ни одному значению из этого списка.

Пусть $A(i)$ – список доступных значений для позиции i.

Пересчитаем вероятность каждого из доступных значений по формуле

$$p_{ij}^* = \frac{p_{ij}}{S_i}, \quad (9)$$

где $S_i = \sum_{j \in A(i)} p_{ij}$, и выберем одно из доступных значений в соответствии с этими вероятностями.

Таким образом, будет получена корректная перестановка. Далее все особи отсортируем в порядке увеличения приспособленности.

Для лучших особей пересчитаем их хромосомы по формулам (5)–(8) аналогично тому, как это описано выше для первого варианта при применении генетических операторов.

Для худших особей их хромосомы также будут пересчитаны аналогично, но с точностью до знака (будем уменьшать вероятность появления соответствующих значений).

Результаты

Оба описанных выше варианта были реализованы в виде компьютерной программы на языке C++. Исходный код программы находится в открытом доступе[1].

Для сравнения также была использована существующая реализация классического генетического алгоритма[2]. Для небольших размерностей входных данных (N < 12) также выполнялось сравнение с точным переборным алгоритмом.

Тестирование проводилось на 7 тестах, со значениями гиперпараметров $\eta_{ind} = 0.05$, $\eta_{DR} = 0.005$.

Данные значения были подобраны экспериментальным путем. Время работы классического ГА на каждом тесте было ограничено 10 секундами. Для подвижного ГА было установлено минимальное количество итераций для завершения – 1000; в случае достижения этого количества время работы ограничивалось также 10 секундами.

Число особей в популяции после отбора было равно 60. Результаты исследования первого варианта реализации подвижного ГА приведены в таблице.

[1] https://github.com/DimaSidorenko/VPR.Fluid_ Genetic_algorithm

[2] https://github.com/Alexandr-TS/CVRP

Сравнение первого варианта реализации подвижного ГА с другими алгоритмами

N	Точный алгоритм	Классический ГА		Подвижный ГА (первый вариант кодирования)	
	Суммарная длина (*F*)	Суммарная длина (*F*)	Количество итераций	Суммарная длина (*F*)	Количество итераций
5	2395	2395	4508046	2395	1164
10	2354	2354	3627050	2354	1000
20	-	1416	2835683	4782	1000
40	-	2136	1528882	13753	1000
60	-	2039	872248	19488	1000
80	-	2152	546916	27304	1000
100	-	2125	353726	36993	1000

Как видно из таблицы, при $N \leq 10$ оба ГА находят точные оптимальные решения. Однако при $N > 20$ подвижный ГА работает значительно хуже классического.

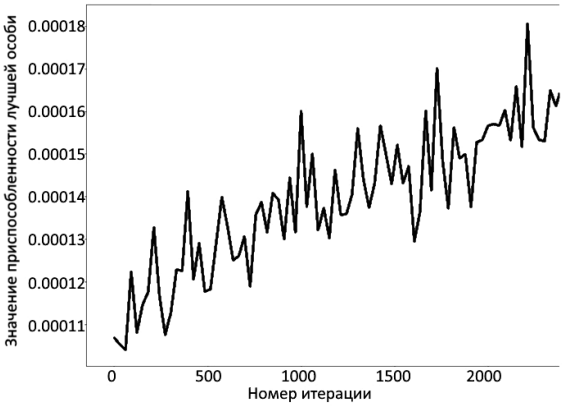
Дальнейший анализ результатов показал, что в подвижном ГА не наблюдается стабильного роста средней приспособленности популяции, иными словами, алгоритм не сходится даже к точке локального экстремума. Одна из причин может заключаться в том, что очень похожие коды, кодируемые хромосомами, порождают совершенно разные перестановки.

Например, коды (4, 2, 1, 1) и (1, 2, 1, 1) отличаются всего одним значением, но при этом кодируют очень разные перестановки (4, 2, 1, 3) и (1, 3, 2, 4) соответственно. Одна из этих перестановок может иметь высокую приспособленность, за счет этого она будет чаще участвовать в операциях скрещивания, вероятность появления соответствующего кода будет увеличиваться. Однако при этом повышается и вероятность появления другого кода, порождающего совсем другую перестановку с возможно очень низкой приспособленностью. В результате средняя приспособленность популяции может не измениться или даже стать меньше.

Еще один фактор, на который можно обратить внимание – подвижный ГА работает значительно медленнее классического.

Как видно из таблицы, при $N \geq 10$ за 10 секунд не успевают завершиться даже 1000 итераций, в то время как количество итераций классического ГА оказывается на несколько порядков больше.

Результаты работы второго варианта реализации подвижного ГА оказались лишь на 10 % лучше первого варианта, хотя из анализа графика (см. рисунок) следует, что средняя приспособленность популяции достаточно стабильно растет.



Зависимость приспособленности лучшей особи в популяции от номера итерации для второго варианта кодирования

По всей видимости, в данном случае решающим становится ограничение в 1000 итераций, которых попросту не хватает для достижения экстремума.

Заключение

Проведенные исследования позволяют утверждать, что применение подвижных генетических алгоритмов для решения задачи маршрутизации транспорта, возможно, и дает корректные результаты. При количестве целей $N \leq 10$ подвижный ГА позволяет получить точное решение.

Однако при больших значениях N алгоритм работает слишком медленно, получаемые результаты оказываются значительно хуже результатов классического ГА. Выбор способа кодирования существенно влияет на результаты работы.

Таким образом, дальнейшие исследования могут быть направлены на оптимизацию подвижного ГА, рассмотрение других способов кодирования и пересчета вероятностей, а также применение дополнительных оптимизационных эвристик.

Список литературы

1. *Сидоренко Д.О., Городилов А.Ю.* Подвижный генетический алгоритм для решения задачи маршрутизации транспорта // материалы Всерос. науч.-практ. конф. молодых ученых с междунар. участием "Математика и междисциплинарные исследования". 2021. С. 177–180.
2. *Гончарова Ю.А., Валеев Р.С., Валеева А.Ф.* Задачи маршрутизации при транспортировке: обзор моделей, методов и алгоритмов // Логистика и управление цепями поставок. 2019. № 4. С. 74–88.
3. *Prins C.* A simple and effective evolutionary algorithm for the vehicle routing problem // Computers & Operations Research. 2004. №31. P. 1985–2002.
4. *Jafari-Marandi R., Smith B. K.* Fluid Genetic Algorithm (FGA) // Journal of Computational Design and Engineering. 2017. Vol. 4. P. 158–167.
5. *Hong, Haoyuan & Panahi, Mahdi & Shirzadi, Ataollah & Ma, Tianwu & Liu, Junzhi & Zhu, A-Xing & Chen, Wei & Kougias, Ioannis & Kazakis, Nerantzis. (2018).* Flood susceptibility assessment in Hengfeng area coupling adaptive neuro-fuzzy inference system with genetic algorithm and differential evolution. Science of The Total Environment. 621. 1124–1141. 10.1016/j.scitotenv.2017.10.114.
6. *Кротких А.А., Максимов П.В.* Постановка обобщенного жидкостного генетического алгоритма и оценка его применимости в рамках решения задачи топологической оптимизации // Математика и междисциплинарные исследования – 2020: материалы Всерос. науч.-практ. конф. молодых ученых с междунар. участием (г. Пермь, 12–14 октября 2020 г.) / гл. ред. А.П. Шкарапута. Пермский государственный национальный исследовательский университет. Пермь, 2020.

References

1. *Sidorenko D.O., Gorodilov A.YU.* Podvizhnyj geneticheskij algoritm dlya resheniya zadachi marshrutizacii transporta // materialy Vserossijskoj nauch.-prakt. konf. molodyh uchenyh s mezhdunar. Uchastiem "Matematika i mezhdisciplinarnye issledovaniya". 2021. S. 177–180.
2. *Goncharova YU.A., Valeev R.S., Valeeva A.F.* Zadachi marshrutizacii pri transportirovke: obzor modelej, metodov i algoritmov // Logistika i upravlenie cepyami postavok. 2019. №4. S. 74–88.
3. *Prins C. A* simple and effective evolutionary algorithm for the vehicle routing problem // Computers & Operations Research. 2004. №31. P. 1985–2002.
4. *Jafari-Marandi R., Smith B. K.* Fluid Genetic Algorithm (FGA) // Journal of Computational Design and Engineering. 2017. Vol. 4. P. 158–167.
5. *Hong, Haoyuan & Panahi, Mahdi & Shirzadi, Ataollah & Ma, Tianwu & olLiu, Junzhi & Zhu, A-Xing & Chen, Wei & Kougias, Ioannis & Kazakis, Nerantzis*. (2018). Flood susceptibility assessment in Hengfeng area coupling adaptive neuro-fuzzy inference system with genetic algorithm and differential evolution. Science of The Total Environment. 621. 1124-1141. 10.1016/j.scitotenv.2017.10.114.
6. *Krotkih A.A., Maksimov P.V.* Postanovka obobshchyonnogo zhidkostnogo geneticheskogo algoritma i ocenka ego primenimosti v ramkah resheniya zadachi topologicheskoj optimizacii // Matematika i mezhdisciplinarnye issledovaniya – 2020: materialy Vserossijskoj nauch.-prakt. konf. molodyh uchenyh s mezhdunarodnym uchastiem (g. Perm', 12–14 oktyabrya 2020 g.) / gl. red. A.P. SHkaraputa. Permskij gosudarstvennyj nacional'nyj issledovatel'skij universitet. Perm', 2020.

Просьба ссылаться на эту статью:
Сидоренко Д.О., Городилов А.Ю. О решении задачи маршрутизации транспорта с помощью подвижного генетического алгоритма // Вестник ПГУ. Математика. Механика. Информатика. 2021. № 4(55). С. 43–48. DOI: 10.17072/1993-0550-2021-4-43-48.

Please cite this article as:
Sidorenko D.O., Gorodilov A.YU. On solving the vehicle routing problem using a fluid genetic algorithm. Bulletin of Perm University // Mathematics. Mechanics. Computer Science. 2021. № 4(55). P. 43–48. DOI: 10.17072/1993-0550-2021-4-43-48.