ЛИТЕРАТУРА

1. *Истигечева Е.В., Григорьева Т.Е., Панов С.А.* Моделирование бизнес-процессов на примере модели «Сбыт» // Таврический научный обозреватель. – 2015. – № 3, ч. 2. – С.55-59.
2. *Истигечева Е.В., Григорьева Т.Е., Панов С.А.* Моделирование бизнес-процессов на примере модели «Хранение» // Научная дискуссия: вопросы технических наук. Сб. статей по материалам XL международной заочной научно-практической конференции. – М., 2015. – № 11 (29). – С. 18-23.
3. *Григорьева Т.Е., Панов С.А.* Сети Петри с транзактами для моделирования бизнес-процессов // Материалы Всероссийской научно-технической конференции студентов, аспирантов и молодых ученых. Научная сессия ТУСУР-2015. – Томск, 2015. – Ч. 5. – С. 304-306.
4. *Дмитриев В.М., Шутенков А.В., Зайченко Т.Н., Ганджа Т.В.* МАРС – среда моделирования технических устройств и систем. – Томск: В-Спектр, 2011.
5. *Дмитриев В.М.,* Ганджа Т.В. Принцип формирования многоуровневых компьютерных моделей SCADA-систем для управления сложными технологическими объектами // Информатика и системы управления. – 2013. – № 2 (36). – С. 24-35.
6. *Григорьева Т.Е.* Дискретно-событийное моделирование в СМ «МАРС» для курса «Системы массового обслуживания» // Доклады Томского государственного университета систем управления и радиоэлектроники. – 2014. – 1(31).– С.152-155.

Статья представлена к публикации членом редколлегии А.А. Шелупановым.

E-mail:
Дмитриев Вячеслав Михайлович – dmitriewvm@gmail.com;
Григорьева Татьяна Евгеньевна – tanya_grig_1991@mail.ru;
Панов Сергей Аркадьевич – spytech3000@gmail.com;
Истигечева Елена Валентиновна – ievne@mail.ru;
Дмитриев Игорь Вячеславович – igor.dmitriev@omu.ru.

УДК 519.854.2

© 2016 г. **О.Э. Долгова**,
В.В. Пересветов, канд. физ.-мат. наук
(Вычислительный центр ДВО РАН, Хабаровск)

ЛУЧЕВОЙ ПОИСК И МУРАВЬИНЫЙ АЛГОРИТМ В РЕШЕНИИ ЗАДАЧИ МАРШРУТИЗАЦИИ ТРАНСПОРТА

Рассматривается задача маршрутизации транспортных средств с ограничениями на грузоподъемность и на длину пройденного пути в пределах одного маршрута. Для ее решения предложен гибридный подход, сочетающий лучевой поиск, алгоритм муравьиных колоний и локальный поиск.
Ключевые слова: задача маршрутизации транспорта с ограничением на грузоподъемность, алгоритм муравьиных колоний, лучевой поиск, локальный поиск.

Введение

В работе рассматривается статическая задача маршрутизации транспорта (транспортных средств) с ограничением на грузоподъемность – *Capacitated Vehicle Routing Problem* (CVRP) – с дополнительным ограничением на длину пройденного пути в пределах маршрута. Эта задача комбинаторной оптимизации является одним из частных случаев общей постановки известной задачи коммивояжера и принадлежит к классу сложности *NP* [1]. Для задач маршрутизации в общем случае не существует методов нахождения их точных решений и проверки оптимальности приближенных за полиномиальное время.

В последнее время ведутся работы в области создания и исследования неклассических алгоритмов глобальной оптимизации, с помощью которых удается найти оптимальные (или близкие к оптимальным в случае задач большой размерности) решения широкого класса задач за приемлемое для практического использования время. При этом часто применяется гибридный подход — сочетаются два и более метода. Для решения рассматриваемых здесь задач CVRP предложены метаэвристические алгоритмы [2 – 9]. Среди них муравьиные алгоритмы (Ant Colony Optimization, ACO) являются одними из наиболее эффективных [2, 5, 8, 9]. Гибридный подход использования метода лучевого поиска (Beam Search) [10] и ACO, известный в англоязычной литературе как Beam-ACO optimization, впервые был предложен в работе [11] для решения задачи календарного планирования. В дальнейшем этот метод применялся для решения задачи коммивояжера с временными окнами обслуживания [12, 13], задачи составления расписаний [14] и др.

Муравьиные алгоритмы совместно с комбинированным локальным поиском и их параллельные варианты с использованием одновременно нескольких независимых колоний разрабатывались нами для решения следующих задач комбинаторной оптимизации: составление расписаний для одного прибора с минимизацией суммарного взвешенного запаздывания [15], CVRP [16], маршрутизация транспорта с временными окнами [17].

В настоящей работе предложена гибридная схема (далее – GB-ACO) алгоритмов решения CVRP на основе подхода Beam-ACO [11, 13]: применяется метод лучевого поиска с механизмом ACO построения допустимых решений и оценки нижних границ. В GB-ACO входят разработанные для этой задачи алгоритмы комбинированного локального поиска. Представлены результаты вычислительных экспериментов с тестовыми задачами из наборов Christofides, Mingozzi, Toth [18] и Taillard [19].

Формулировка задачи

Пусть $V = \{v_0,...,v_N\}$ – множество вершин, представляющее географическое расположение склада и клиентов. Для каждого клиента $v_i \in V' = V \backslash \{v_0\} = \{v_1,...,v_N\}$ заданы координаты (x_i, y_i), спрос товара $q(i) > 0$, время обслуживания $s(i)$. Для склада также заданы соответствующие параметры: (x_0, y_0), $q(0) = 0$, $s(0) = 0$. Расстояние между объектами v_i и v_j, $0 \le i \le N$, $0 \le j \le N$:

$d(i,j)=\sqrt{\left(x_i-x_j\right)^2+\left(y_i-y_j\right)^2}$, $d(i,j)=d(j,i)$, $d(i,i)=\infty$.

Для доставки товара имеется некоторое количество идентичных транспортных средств с грузоподъемностью $Q \geq \max\{q(i): i=1,...,N\}$. Определена максимальная длина пройденного пути L_m при обслуживании клиентов в пределах маршрута. Это дополнительное ограничение не входит в исходную формулировку CVRP, но необходимо для решения некоторых задач из тестового набора Christofides, Mingozzi, Toth [18] и часто используется [2, 4 – 6, 16] и др. Каждый маршрут начинается и заканчивается на складе. Каждый клиент обслуживается одним транспортным средством и единожды.

Целью решения рассматриваемой задачи является нахождение множества $R=\{r_1,...,r_K\}$ независимых маршрутов движения K транспортных средств минимальной суммарной длины.

Пусть $r_k(i)=w_i$, $i=0,1,...,N_k,N_{k+1}$ — последовательность обслуживания N_k клиентов $\{w_1,...,w_{N_k}\}\subseteq V'$ в маршруте k, $w_0=w_{N_k+1}=v_0$. Длина маршрута k находится следующим образом:

$$D(r_k)=\sum_{i=0}^{N_k} d(r_k(i),r_k(i+1)). \quad (1)$$

Составляя маршруты, определим задачу минимизации общей длины маршрутов (целевой функции)

$$f(R)=\sum_{r_k\in R} D(r_k)=\sum_{r_k\in R}\sum_{i=0}^{N_k} d(r_k(i),r_k(i+1))\to \min, \quad (2)$$

не нарушая ограничений на грузоподъемность

$$q(r_k)=\sum_{i=1}^{N_k} q(r_k(i))\leq Q,\ \forall r_k\in R \quad (3)$$

и на максимальное пройденное расстояние в пределах маршрута с учетом обслуживания

$$\sum_{i=0}^{N_k} d(r_k(i),r_k(i+1))+\sum_{i=1}^{N_k} s(r_k(i))\leq L_m, \forall r_k\in R. \quad (4)$$

Алгоритм решения

В настоящей работе для решения задачи CVRP реализована схема построения гибридного метода лучевого поиска и муравьиных колоний, которая аналогична предложенной в [13] для решения задачи коммивояжера. В разработанном гибридном алгоритме вероятностный лучевой поиск применяется для нахождения допустимых решений с использованием следов феромона – множества T с элементами τ_{ij}, $\tau_{ij}=\tau_{ji}$, $i\neq j$, $0\leq i,j\leq N$. Из найденных допустимых решений выбирается лучшее на данной итерации и делается попытка его усовершенствовать с помощью методов локального поиска. Если было улучшение, то обновляется лучшее решение, найденное с начала работы алгоритма, вычисляется показатель сходимости и принимается решение о перезапуске. В общем случае

метод работает до выполнения всех заданных итераций или достижения максимального времени работы программы.

Метод лучевого поиска является вариацией метода ветвей и границ. Основная идея его состоит в разбиении множества решений, представляемых вершинами, на попарно непересекающиеся множества. Каждое подмножество в этом разбиении представляется потомком исходной вершины.

Упорядоченное множество с номерами посещенных объектов, которое может быть расширено за счет добавления необслуженных клиентов, называется частичным (неполным) решением. Пусть B_t – множество активных вершин (луч), t определяет шаг разбиения множества решений, $|B_t| \le k_{bw}$, где $k_{bw} \in Z^+$ – ширина луча. В процессе разбиения, не нарушая ограничений (3), (4), может порождаться не более k_{ext} потомков для каждой вершины из B_t, которые представляют собой непосещенных клиентов или склад. Каждое множество полученных решений запоминается в B_{ext}, если решение неполное, или в B_c, когда все клиенты обслужены. При формировании нового множества B_{t+1} для следующего шага $t+1$ выбирается до k_{bw} решений из набора B_{ext}. В оценке частичных решений используется нижняя граница — минимальное значение $f(R)$ для любого полного решения, которое может быть получено из частичного.

Далее изложен алгоритм 1 вероятностного лучевого поиска – Probabilistic Beam Search (PBS) [13], который используется в алгоритме GB-ACO для поиска возможных решений.

Алгоритм 1. PBS.

$t := 0$;
$B_t := \{v_0\}$;
$B_c := \{\ \}$;
WHILE (B_t не пусто) **DO**
  $B_{ext} := \{\ \}$;
  FOR $A = 1,...,|B_t|$ **DO**
    выбрать вершину ветвления A из B_t;
    породить потомков $i^A = \overline{1, n_A}, n_A \le k_{ext}$ вершины A по правилу перехода ACO и нижние границы;
    IF ($i^A = \overline{1, n_A}$ это частичное решение) **THEN**
      сохранить i^A в наборе B_{ext};
    ELSE
      сохранить i^A в наборе B_c;
  $t := t + 1$;
  $B_t := \{\ \}$;
  выбрать до $\min\{k_{bw}, |B_{ext}|\}$ лучших продолжений из набора B_{ext} и сформировать новый луч B_t;
RETURN $\arg\min_f \{R | R \in B_c\}$.

В алгоритме 1 на шаге t для каждой вершины A определен список табу с номерами уже посещенных клиентов, кандидаты для ветвления выбираются согласно этому списку. Если никакой из непосещенных клиентов не может быть включен в текущий маршрут из-за нарушений (3), (4), то начинается новый маршрут со склада. Если все клиенты включены в маршруты, то ветвление прекращается и решение считается полным.

Пусть Ω – склад и множество непосещенных клиентов, включение которых в текущий маршрут не нарушает (3), (4). В алгоритме 1 в частичном решении с вероятностью $p_0 \in [0,1]$ после v_i-го клиента (или склада) выбирается клиент v_j с учетом $\eta_{ij} = 1/d(i,j)$ по формуле:

$$v_j = \arg\max_{h\in\Omega}\{\tau_{ih}^{\alpha}\cdot\eta_{ih}^{\beta}\}. \qquad (5)$$

В остальных случаях (с вероятностью $1-p_0$) переход из вершины v_i в вершину v_j выполняется по правилу рулетки согласно P_{ij}:

$$P_{ij} = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\cdot\eta_{ij}^{\beta}}{\sum\limits_{h\in\Omega}\tau_{ih}^{\alpha}\cdot\eta_{ih}^{\beta}}, v_j \in \Omega \\ 0, \qquad\qquad v_j \notin \Omega \end{cases}, \qquad (6)$$

где α и β – параметры, определяющие влияние τ_{ij} и η_{ij} соответственно.

Отбор лучших частичных решений происходит в соответствии с оценками нижних границ целевой функции любого полного решения в данном подмножестве. Алгоритм вычисления нижних границ – ключевой компонент лучевого поиска. При этом часто имеется выбор между границами, которые относительно точны, но требуют относительно большого времени вычисления, и границами, которые не столь точны, но быстро вычисляются. В этой работе реализована стратегия, предложенная в [13]. Пусть R^p – частичное решение. Построим до N^s полных решений, добавляя к R^p один за другим непосещенных клиентов до тех пор, пока все клиенты не будут обслужены. Следующий клиент для включения в частичное решение находится по формулам (5), (6). Нижней границей для R^p будет минимальное значение $f(R)$, найденное среди значений N^s полных решений. Введем обозначения: R^{bf} – лучшее решение, найденное с начала работы алгоритма; R^{rb} – лучшее решение, найденное с момента последнего повторного запуска; R^{ib} – лучшее решение на данной итерации; $0 \le cf \le 1$ – показатель сходимости, который вычисляется следующим образом [12]:

$$cf = \frac{2\cdot\sum_{\tau_{ij}\in T}\max(\tau^{\max}-\tau_{ij},\tau_{ij}-\tau^{\min})}{|T|\cdot(\tau^{\max}-\tau^{\min})} - 1, \qquad (7)$$

где $\tau^{\max} = 0.999$ и $\tau^{\min} = 0.001$ – верхняя и нижняя границы $\tau^{\min} \le \tau_{ij} \le \tau^{\max}$. При инициализации и повторном запуске алгоритма $cf = 0$, $\tau_{ij} = 0.5\ \forall \tau_{ij} \in T$. Далее изложена схема построения алгоритма решения рассматриваемой задачи.

Алгоритм 2. GB-ACO.

INPUT: α, β, p_0, k_{bw}, k_{ext}, N^s;

1: $R^{bf} := null$, $R^{rb} := null$, $cf := 0$, $bsUpdate :=$ **false**;
2: $\tau_{ij} := 0.5 \ \forall \tau_{ij}$;
3: **WHILE** (не достигнут критерий останова) **DO**
4: $R^{ib} := \mathrm{PBS}(k_{bw}, k_{ext}, N^s)$ по алгоритму 1;
5: локальный поиск по алгоритму 3;
6: **IF** ($R^{ib} < R^{rb}$) **THEN**
7: $R^{rb} := R^{ib}$;
8: **IF** ($R^{ib} < R^{bf}$) **THEN**
9: $R^{bf} := R^{ib}$;
10: вычислить показатель сходимости cf;
11: **IF** ($bsUpdate =$ **true**) **AND** ($cf > 0.99$) **THEN**
12: $\tau_{ij} := 0.5 \ \forall \tau_{ij}$;
13: $R^{rb} := null, bsUpdate :=$ **false**;
14: **ELSE**
15: **IF** ($cf > 0.99$) **THEN**
16: $bsUpdate :=$ **true**;
17: обновление $\tau_{ij} \in T$;
18: **RETURN** R^{bf}.

В алгоритм 2 включен алгоритм 1, позволяющий найти решение на данной итерации R^{ib}, которое может быть улучшено с помощью локального поиска. Если было улучшение $f(R)$ с начала работы алгоритма, то обновляются R^{rb}, R^{bf} и вычисляется показатель сходимости cf по формуле (7). Далее в зависимости от полученного значения cf и логической переменной $bsUpdate$ принимается решение о перезапуске алгоритма.

При обновлении $\tau_{ij} \in T$ используются три решения: R^{ib}, R^{rb} и R^{bf}. Влияние каждого из них зависит от величины показателя сходимости cf. Значения τ_{ij} корректируются в соответствии с формулой [13]:

$$\tau_{ij} = \tau_{ij} + \rho \cdot \left(\xi_{ij} - \tau_{ij}\right), \tag{8}$$

где $\xi_{ij} = \kappa^{ib} \cdot R^{ib}_{ij} + \kappa^{rb} \cdot R^{rb}_{ij} + \kappa^{bf} \cdot R^{bf}_{ij}$, $\rho \in (0,1]$ – коэффициент испарения феромона, $R^{*}_{ij} = 1$, если после клиента i обслуживается клиент j, иначе $R^{*}_{ij} = 0$. Параметры κ^{ib}, κ^{rb}, κ^{bf} задают веса решений R^{ib}, R^{rb}, R^{bf}, $\kappa^{ib} + \kappa^{rb} + \kappa^{bf} = 1$.

В табл. 1 приведены значения κ^{ib}, κ^{rb}, κ^{bf} в зависимости от cf и логической переменной $bsUpdate$. После обновления: если $\tau_{ij} > \tau^{\max}$, то $\tau_{ij} = \tau^{\max}$; если $\tau_{ij} < \tau^{\min}$, то $\tau_{ij} = \tau^{\min}$.

Таблица 1

bsUpdate	*false*				*true*
cf	$[0,0.4)$	$[0.4,0.6)$	$[0.6,0.8)$	$[0.8,1.0]$	–
κ^{ib}	1	2/3	1/3	0	0
κ^{rb}	0	1/3	2/3	1	0
κ^{bf}	0	0	0	0	1

Алгоритм локального поиска начинается с выбора некоторого начально допустимого решения и системы окрестностей, которая будет просматриваться для улучшения решения. Когда удается найти улучшенное решение, оно запоминается, затем поиск продолжается в окрестности этого нового решения, пока локальный оптимум не будет найден. Для рассматриваемой задачи теоретических обоснований по выбору «хорошей» системы окрестностей и методу поиска по ней нет.

В этой работе был реализован один метод в пределах одного маршрута – **3-exchange** и четыре метода для двух различных маршрутов – **2-exchange**, **2-exchange***, **Relocate**, **Exchange**, описанных в [20]. В этих алгоритмах поиск по окрестности выполняется до первого улучшения. Окрестности просматриваются в соответствии с естественным лексикографическим порядком, индуцированным нумерацией.

Алгоритм 3. Комбинированная схема локального поиска.

INPUT: R^{ib};
1: **FOR** $iteration_{ls} = 1,...,MaxIteration_{ls}$ **DO**
2: выбрать случайно два маршрута r_1, r_2 из решения R^{ib};
3: выбрать случайным образом число x из $\{1,2\}$;
4: **IF** ($x = 1$) **THEN**
5: **2-exchange**(r_1, r_2);
6: **IF** ($x = 2$) **THEN**
7: **2-exchange***(r_1, r_2);
8: **Relocate**(r_1, r_2);
9: **Exchange**(r_1, r_2);
10: $\forall r \in R^{ib}$ **3-exchange**(r);
11: **RETURN** R^{ib}.

Результаты вычислительных экспериментов

Во всех расчетах использовались следующие значения параметров GB-ACO: $\alpha = 1.0$, $\beta = 1.0$, $\rho = 0.45$, $p_0 = 0.95$; $k_{bw} = 2 \cdot \mathrm{int}((N-1)/50) + 2$, $k_{ext} = 2$, $N^s = 1$ (если $N \le 50$) или $N^s = 2$ (если $N > 50$); $MaxIteration_{ls} = C_{lsIter} \cdot N^2$, $C_{lsIter} = 0.06$. Всюду время решения дано в секундах. Число запусков программы для каждой тестовой задачи – 50. Все численные результаты решения задач методом GB-ACO получены с использованием микропроцессора Intel E5-2630 2.3 ГГц, операционная система GNU/Linux 2.6.32, язык программирования Fortran 95.

Далее приведены результаты решения CVRP двух тестовых наборов. Тестовый набор Christofides, Mingozzi, Toth [18] включает 14 задач различной сложности в зависимости от числа клиентов N, грузоподъемности Q, случайного (задачи C1-C10) или кластерного (задачи C11-C14) расположения клиентов, ограничений на длину пройденного пути в пределах маршрута L_m (табл. 2). Известные оптимальные значения — best-so-far (BSF) – впервые были опубликованы для задач C1-C4, C6-C9 и C11-C14 – в [21]; C5 – в [22]; C10 – в [19].

Таблица 2

Тест	N	Q	L_m	BSF	**GB-ACO**	IACO	ACO	ABC	OCGA
C1	50	160	∞	524.61	0/0	0/0	0.00/0.48	0/0.309	0/–
C2	75	140	∞	835.26	0/0.012	0/1.627	1.27/2.12	0.18/0.923	0/–
C3	100	200	∞	826.14	0/0.147	0.47/2.20	1.47/3.43	0.61/1.02	0/–
C4	150	200	∞	1028.42	0.206/0.690	0/1.37	4.94/6.64	0.32/3.25	0/–
C5	199	200	∞	1291.29	1.34/2.95	1.08/2.36	8.17/9.71	2.24/3.09	0.647/–
C6	50	160	200	555.43	0/0	0/0.848	0.00/1.32	0/0.479	0/–
C7	75	140	160	909.68	0/0	0/1.035	2.21/4.04	0/0.758	0/–
C8	100	200	230	865.94	0/0	0/0.644	1.07/2.94	0/1.176	0/–
C9	150	200	200	1162.55	0.011/0.197	0/2.78	5.46/8.65	0.66/2.59	0.0713/–
C10	199	200	200	1395.85	0.945/1.79	0/1.22	9.25/11.25	1.39/2.73	0.744/–
C11	120	200	∞	1042.11	0/0	0/0.577	0.77/1.29	0.75/1.28	0/–
C12	100	200	∞	819.56	0/0	0/0.500	2.05/3.55	0/0.221	0/–
C13	120	200	720	1541.14	0.112/0.284	0.31/0.721	1.90/2.51	0.31/1.11	0.0720/–
C14	100	200	1040	866.37	0/0	0/0.0785	0.15/0.30	0/0.257	0/-

В табл. 2 показаны через разделитель / минимальные $\delta_{\min}$ и средние δ_{avg} значения погрешности отклонения найденных значений целевой функции $f(R)$ от известных BSF: $\delta = 100\% \cdot (f(R) - \text{BSF})/\text{BSF}$. Критерием останова для GB-ACO было достижение максимального времени решения: $T_{\max} = 1000$ для задач C1-C3, C6-C8, C11, C12, C14 и $T_{\max} = 4000$ – для задач C4, C5, C9, C10, C13.

В табл. 2 результаты проведенных вычислительных экспериментов методом GB-ACO сравниваются с результатами, полученными с использованием метаэвристических алгоритмов IACO [2], ACO [5], ABC [4], OCGA [6]. Алгоритм IACO [2] – улучшенный алгоритм муравьиной колонии с новой стратегией обновления следов феромона и оператором мутации, дополненный локальным поиском (**2-opt** обмен в пределах одного маршрута). В табл. 2 показаны результаты [2], полученные авторами с помощью процессора Intel Pentium 1 ГГц. ACO [5] – алгоритм муравьиной колонии с различными эвристиками для улучшения последовательности обслуживания клиентов в пределах маршрута. В табл. 2 представлены результаты решения для варианта построения ACO и схемы локального поиска **swap**+**3-opt**, который авторы считают удовлетворительным. Число запусков – 10, критерий останова – число итераций, превышающее 50000. Процессор Intel Core2 Duo 3 ГГц. ABC [4] – усовершенствованный алгоритм искусственной пчелиной колонии. Результаты (табл. 2) были получены в [4] без использования локального поиска и других методов. Число запусков – 20, критерий останова – превышение числа итераций $2000 \cdot N$ (зависит от размерности задачи). Процессор – Intel Pentium

1.73 ГГц. OCGA [6] – усовершенствованный генетический алгоритм. Процессор – Intel Pentium 4 2.0 ГГц. Число запусков – 50, в каждом запуске расчет прекращался, если за последние 100 поколений не было улучшения значения целевой функции.

При решении методом GB-ACO некоторых задач удается получать известные BSF за относительно небольшое время во всех запусках программы. Для таких задач проведены дополнительные вычислительные эксперименты без ограничений $T_{\max}$. В табл. 3 представлены минимальные $t_{\min}$, максимальные $t_{\max}$ и средние t_{avg} значения времени решения задач методом GB-ACO по 50 запускам, где также показаны средние t_{avg}, полученные другими описанными методами: IACO [2], ACO [5], ABC [4], OCGA [6].

Таблица 3

Тест	**GB-ACO**			IACO	ACO	ABC	OCGA
	$t_{\min}$	$t_{\max}$	t_{avg}	t_{avg}	t_{avg}	t_{avg}	t_{avg}
C1	0.03	0.95	0.34	2	412.7	39	48.33
C6	0.05	5.4	1.7	24	41.34	42	75.09
C7	2.3	694	147	20	34.35	141	257.41
C8	1.7	103	31	57	163.1	152.4	440.77
C11	5.4	871	212	61	788.0	208.2	–
C12	0.14	1.8	0.73	31	242.5	174	267.13
C14	0.12	10	2.7	43	143.7	201	394.38

Метод GB-ACO позволил значительно улучшить опубликованные нами в [16] результаты решения задач размерности до 120 из тестового набора Christofides, Mingozzi, Toth с помощью с гибридного подхода, в котором максиминный алгоритм муравьиных систем дополнен локальным поиском.

Для дополнительного исследования GB-ACO в решении задач кластерного типа проведены вычислительные эксперименты с тестовым набором Taillard [19]. Из него решались по четыре задачи для размерностей $N = 75,100,150$ (табл. 4). У всех этих задач кластерное расположение клиентов с различным числом таких кластеров и их плотностью, нет ограничений на длину пройденного пути в пределах одного маршрута ($L_m = \infty$). Величина спроса товара q имеет экспоненциальное распределение.

Значения BSF были опубликованы для задач tai75a, tai75c, tai75d, tai150a, tai150c, tai150d в [21]; tai75b в [23]; tai100a, tai100c, tai150b в [24]; tai100b в [22]; tai100d в [25].

В табл. 4 показаны через разделитель / минимальные $\delta_{\min}$ и средние δ_{avg} значения погрешности отклонения найденных значений целевой функции $f(R)$ от известных BSF: $\delta = 100\% \cdot (f(R) - \text{BSF})/\text{BSF}$.

Критерием останова для GB-ACO являлось достижение максимального времени решения задач: $T_{\max} = 1000$ для tai75a-d, tai100a, tai100c, tai100d и $T_{\max} = 4000$ для tai100b, tai150a-d.

Таблица 4

Тест	*N*	*Q*	BSF	**GB-ACO**	JCell2o1i	OCGA
tai75a	75	1445	1618.36	0/0	0/0.0884	0/–
tai75b	75	1679	1344.62	0/0	0/0.0357	0.0007/–
tai75c	75	1122	1291.01	0/0	0/0.294	0/–
tai75d	75	1699	1365.42	0/0	0/0.0081	0/–
tai100a	100	1409	2041.34	0/0.118	0.32/1.42	0.456/–
tai100b	100	1842	1939.90	0.037/0.045	0.02/0.191	0/–
tai100c	100	2043	1406.20	0/0	0.39/0.940	0.156/–
tai100d	100	1297	1580.46	0/0.064	0.24/1.21	0.0481/–
tai150a	150	1544	3055.23	0/0.254	0.04/2.23	0/–
tai150b	150	1918	2656.47	2.86/3.05	2.87/4.73	3.71/–
tai150c	150	2021	2341.84	0.868/0.962	0.95/2.90	0.470/–
tai150d	150	1874	2645.39	0.013/0.343	0.35/1.63	0.565/–

В табл. 4 результаты проведенных экспериментов методом GB-ACO сравниваются с результатами, полученными методами, основанными на генетических алгоритмах: JCell2o1i [23], OCGA [6]. В JCell2o1i [23] реализован генетический алгоритм с локальным поиском **2-opt** и **1-Interchange**. Число запусков – 100, частота процессора 2.8 ГГц. OCGA [6] был описан выше.

Обобщая результаты вычислительных экспериментов с тестовым набором Christofides, Mingozzi, Toth и набором Taillard, можно сделать вывод, что метод GB-ACO эффективен в решении задач кластерного типа и задач небольшой размерности случайного типа. Для большинства тестовых задач были найдены оптимальные маршруты за приемлемое для практики время.

Заключение

Для решения задач маршрутизации транспорта разработан и реализован метаэвристический гибридный метод, в котором лучевой поиск и алгоритм муравьиной колонии дополнен комбинированной схемой локального поиска. Использование муравьиного алгоритма для вероятностного построения полных решений с последующей оценкой нижних границ оказалось результативным.

ЛИТЕРАТУРА

1. *Lenstra J.K., Kan A.H.G. Rinnooy*. Complexity of vehicle routing and scheduling problems // Networks. – 1981. – Vol. 11, № 2. – P. 221-227.
2. *Yu Bin, Yang Zhong-Zhen, Yao Baozhen*. An improved ant colony optimization for vehicle routing problem // European Journal of Operational Research. – 2009. – Vol. 196, № 1. – P. 171-176.
3. *Ипатов А.В.* Модифицированный метод имитации отжига в задаче маршрутизации транспорта // Труды Института математики и механики УрО РАН. – 2011. – Т. 17, № 4. – С. 121-125.
4. *Szeto W.Y., Wu Yongzhong, Sin C. Ho.* An artificial bee colony algorithm for the capacitated vehicle routing problem // European Journal of Operational Research. – 2011. – Vol. 215, № 1. – P. 126-135.
5. *Tan W.F., Lee L.S., Majid Z.A., Seow H.V.* Ant colony optimization for capacitated vehicle routing problem // Journal of Computer Science. – 2012. – Vol. 8, № 6. – P. 846-852.
6. *Nazif H., Lee L.S.* Optimised crossover genetic algorithm for capacitated vehicle routing problem // Applied Mathematical Modelling. – 2012. – Vol. 36, № 5. – P. 2110-2117.

7. *Zhou Yongquan, Xie Jian, Zheng Hongqing*. A hybrid bat algorithm with path relinking for capacitated vehicle routing problem // Mathematical Problems in Engineering. – 2013.
8. *Сластников С.А.* Применение метаэвристических алгоритмов для задачи маршрутизации транспорта // Экономика и математические методы. – 2014. – Т. 50, № 1. – С. 117-126.
9. *Кажаров А.А., Курейчик В.М.* Муравьиные алгоритмы для решения транспортных задач // Известия РАН. Теория и системы управления. – 2010. – № 1. – С. 32-45.
10. *Ow P.S., Morton T.E.* Filtered beam search in scheduling // International Journal of Production Research. – 1988. – Vol. 26, № 1. – P. 297-307.
11. *Blum C.* Beam-ACO – hybridizing ant colony optimization with beam search: an application to open shop scheduling // Computers and Operations Research. – 2005. – Vol. 32, № 6. – P. 1565-1591.
12. Beam-ACO based on stochastic sampling for makespan optimization concerning the TSP with time windows / *M. Lopez-Ibanez, C. Blum, D. Thiruvady* et al. // Lecture Notes in Computer Science. – 2009. – Vol. 5482. – P. 97-108.
13. *Lopez-Ibanez M., Blum C.* Beam-ACO for the travelling salesman problem with time windows // Computers and Operations Research. – 2010. – Vol. 37, № 9. – P. 1570-1583.
14. Hybridizing Beam-ACO with constraint programming for single machine job scheduling / *D. Thiruvady, C. Blum, M. Meyer, A. Ernst* // Lecture Notes in Computer Science. – 2009. – Vol. 5818. – P. 30-44.
15. *Долгова О.Э, Пересветов В.В.* Составление расписаний с минимизацией суммарного запаздывания на одном приборе методом параллельных муравьиных колоний // Вестник ТОГУ. – 2012. – Т. 25, № 2. – С. 45-52.
16. *Долгова О.Э, Пересветов В.В.* Задача маршрутизации транспортных средств с заданной грузоподъемностью // Информационные технологии XXI века: Материалы междунар. науч. конф. – Хабаровск, 2013. – С. 330-335.
17. *Долгова О.Э, Пересветов В.В.* Задача маршрутизации транспортных средств с временными окнами // Информационные технологии и высокопроизводительные вычисления: Материалы Всероссийской науч.-практ. конф. – Хабаровск, 2013. – С. 119-124.
18. *Christofides N., Mingozzi A., Toth P*. The vehicle routing problem // Combinatorial optimization / Ed. by A. Mingozzi, P. Toth, N. Christofides, C. Sandy. – Wiley, Chichester, 1979. – P. 315-338.
19. *Rochat Y., Taillard E.D.* Probabilistic diversification and intensification in local search for vehicle routing // Journal of heuristics. – 1995. – Vol. 1. – P. 147-167.
20. *Savelsbergh M.W.P.* The vehicle routing problem with time windows: Minimizing route duration // ORSA Journal on Computing. – 1992. – Vol. 4. – P. 146-154.
21. *Taillard R.E.* Parallel iterative search methods for vehicle routing problems // Networks. – 1993. – Vol. 23, № 8. – P. 661-673.
22. *Mester D., Braysy O.* Active guided evolution strategies for large-scale vehicle routing problems with time windows // Computers and Operations Research. – 2005. – Vol. 32, № 6. – P. 1593-1614.
23. *Alba E., Dorronosoro B*. Computing nine new best-so-far solutions for capacitated VRP with a cellular genetic algorithm // Information Processing Letters. – 2006. – Vol. 98, № 6. – P. 225-230.
24. *Gambardella L., Taillard E., Agazzi G.* MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows // New Ideas in Optimization, McGraw-Hill / Ed. by D. Corne, M. Dorigo, F. Glover. – 1999. – P. 63-76.
25. *Nagata Y., Braysy O.* Edge assembly-based memetic algorithm for the capacitated vehicle routing problem // Networks. – 2009. – Vol. 54, № 4. – P. 205-215.

Статья представлена к публикации членом редколлегии С.И. Смагиным.

E-mail:
Долгова Ольга Эдуардовна – o.dolgova@live.ru;
Пересветов Владимир Викторович – vvperesv@yandex.ru.