ДИСКРЕТНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ
Апрель–июнь 2020. Т. 27, № 2. С. 43–64

УДК 519.8+518.25 DOI 10.33048/daio.2020.27.673

ГИБРИДНЫЙ АЛГОРИТМ ЛОКАЛЬНОГО ПОИСКА ДЛЯ ЗАДАЧИ МАРШРУТИЗАЦИИ ТРАНСПОРТНЫХ СРЕДСТВ С МНОГОКРАТНЫМ ПОСЕЩЕНИЕМ КЛИЕНТОВ

И. Н. Кулаченко[1,a], *П. А. Кононова*[2,b]

[1] Новосибирский государственный университет,
ул. Пирогова, 90 Новосибирск, Россия

[2] Институт математики им. С. Л. Соболева СО РАН,
пр. Акад. Коптюга, 4, 630090 Новосибирск, Россия

E-mail: [a] soge.ink@gmail.com, [b] pkononova@math.nsc.ru

Аннотация. Рассматривается задача построения маршрутов для транспортных средств на конечном временном интервале. Компания имеет множество транспортных средств ограниченной грузоподъёмности, находящихся в разных депо. Требуется построить такие маршруты посещения всех клиентов, чтобы суммарное пройденное расстояние было минимальным, а рабочее время водителей укладывалось в рабочую смену. Задана частота посещений каждого клиента, которые должны проходить через равные промежутки времени. Обслуживание клиента должно производиться одним и тем же транспортным средством.

Построена модель частично-целочисленного линейного программирования. Разработан метод локального поиска с чередующимися окрестностями, усиленный методом поиска с запретами. Для расширения пространства решений ограничение на длительность рабочей смены и грузоподъёмность заносятся в целевую функцию со штрафами, изменяемыми в ходе поиска. Процедуры интенсификации и диверсификации применяются для повышения эффективности метода. Приводятся результаты расчётов на реальных исходных данных. Табл. 6, ил. 1, библиогр. 28.

Ключевые слова: метод штрафов, метаэвристика, окрестность Кернигана — Лина, транспортное средство ограниченной грузоподъёмности.

Исследование выполнено при финансовой поддержке Российского фонда фундаментальных исследований и Новосибирской области (проект № 19–47–540005).

Введение

Задачи маршрутизации транспортных средств (Vehicle Routing Problem, VRP) представляют собой один из широко известных классов задач комбинаторной оптимизации. В классическом варианте VRP имеется неограниченный парк идентичных транспортных средств (ТС) и конечное множество клиентов. Требуется построить такой набор маршрутов, чтобы все клиенты были обслужены, а суммарное пройденное расстояние было минимальным. Каждый маршрут начинается и заканчивается в депо. Изучение этой задачи началось более 60 лет назад, когда впервые была предложена математическая постановка и алгоритм решения задачи развозки бензина на заправки [1]. Известно, что задача VRP NP-трудна, поскольку задача коммивояжёра является её частным случаем. Предложено множество обобщений этой модели. Одним из естественных уточнений является ограниченная грузоподъёмность ТС, так называемая Capacitated VRP, или CVRP. Если ТС находятся не в одном депо, а в нескольких, то задачу называют Multiple Depot VRP, или MDVRP [2].

Другое важное обобщение задачи появляется с введением конечного планового периода для обслуживания клиентов [3], когда задана частота обслуживания каждого клиента или даже дни его обслуживания. В новой постановке Periodic VRP, или PVRP, нужно построить набор маршрутов транспортных средств для каждого дня планового периода. Дальнейшее развитие этой модели связано с желанием логистической компании получить дополнительные конкурентные преимущества. С этой целью вводятся два новых требования к маршрутам транспортных средств. Во-первых, клиент должен обслуживаться одним и тем же транспортным средством на протяжении всего планового периода, что упрощает общение поставщика и клиента. Во-вторых, обслуживание должно проводиться примерно в одно и то же время, чтобы клиент мог к нему подготовиться заранее. Такую модель принято называть Consistent VRP, или ConVRP [4].

В настоящей работе рассматривается новая задача типа ConVRP, которая родилась в результате сотрудничества с одной из российских логистических компаний. Компания владеет набором ТС различной грузоподъёмности. Каждое ТС базируется в своём депо. Известно множество клиентов. Каждый клиент имеет свою частоту посещения и обслуживается одним и тем же ТС. Частота посещения задаётся на весь плановый период, при этом интервалы между посещениями должны быть одинаковыми (раз в неделю по понедельникам, раз в две недели, и т. п.). Время посещения может быть любым в рамках рабочей смены, т. е. порядок посещения клиентов внутри рабочего дня может быть произвольным. Требуется построить такое расписание обслуживания клиентов и набор

маршрутов ТС для каждого дня планового периода, чтобы суммарное пройденное расстояние было минимальным и выполнялись условия на частоту посещения клиентов, грузоподъёмность ТС и длину рабочей смены. В [4–6] рассматривались близкие по смыслу задачи. В [5] игнорировались ограничения на согласованность посещений. В [4, 6] клиенты должны посещаться только одним ТС примерно в одно и то же время, но дни посещения фиксированы.

Поскольку рассматриваемая задача NP-трудна, для её решения применяются метаэвристики [7], в частности, алгоритм локального поиска с чередующимися окрестностями (Variable Neighborhood Search, VNS) [8, 9]. Разработаны 9 окрестностей, включая четыре большие окрестности Кернигана — Лина [10, 11], учитывающих частоту посещения клиентов и интервалы между их посещениями. Ограничения на длительность рабочей смены и грузоподъёмность ТС заносятся в целевую функцию со штрафом. Такой подход позволяет расширить область поиска и выходить за границы допустимой области. Величина штрафа меняется в ходе поиска в зависимости от величины нарушений соответствующих ограничений. Процедуры интенсификации и диверсификации применяются для повышения эффективности алгоритма. Алгоритм тестировался на реальных исходных данных и показал высокую эффективность. Для сравнения приводятся результаты расчётов базовой версии VNS и коммерческого решателя Gurobi, который применяется к построенной модели частично-целочисленного линейного программирования. Предварительная версия статьи докладывалась на международных конференциях MOTOR 2019 [12] и MIM 2019 [13].

Статья организована следующим образом. В разд. 1 вводятся обозначения и приводится математическая модель. Она даёт возможность использовать коммерческие решатели для получения точных и приближённых решений. В разд. 2 приведено описание используемых окрестностей для гибридного алгоритма VNS. В разд. 3 представлена схема локального поиска с чередующимися окрестностями с интенсификацией и диверсификацией поиска. В разд. 4 приводятся результаты численных экспериментов. В последнем разделе приводятся заключения и направления дальнейших исследований.

1. Математическая модель

Рассмотрим полный ориентированный граф $G = (V, A)$ с множеством вершин V и множеством рёбер A. Для каждого ребра $(i, j) \in A$ известна длина d_{ij} и время на передвижение t_{ij}. Матрицы (d_{ij}) и (t_{ij}) симметричны и удовлетворяют неравенству треугольника. Множество $V = M \cup I$ является объединением множеств депо M и мест расположения клиентов I. Число клиентов обозначим через $n = |I|$. В каждом депо $m \in M$

имеется определённое количество транспортных средств различной грузоподъёмности. Конечное множество K задаёт всё множество ТС. Для каждого $k \in K$ известна грузоподъёмность v_k и депо $m(k)$, в котором находится ТС. Ежедневно каждое ТС выезжает из депо в момент времени 0 и должно вернуться в него до времени T. Предполагается, что в один день нельзя несколько раз возвращаться в депо. Для каждого клиента $i \in I$ заданы спрос q_i и частота посещения μ_i. Клиент должен быть посещён μ_i раз в течение планового периода D, каждый раз одним и тем же ТС. Временной интервал между последовательными посещениями клиента i должен быть одинаковым и равным $\tau_i = \lfloor |D|/\mu_i \rfloor$. Через s_i обозначим время обслуживания, которое положительно для всех клиентов и равно нулю для депо.

Введём переменные задачи:

$$x_{ijkd} = \begin{cases} 1, & \text{если маршрут ТС } k \text{ содержит ребро } (i,j) \text{ в день } d, \\ 0 & \text{иначе;} \end{cases}$$

$$y_{ikd} = \begin{cases} 1, & \text{если маршрут ТС } k \text{ в день } d \text{ содержит вершину } i \in V, \\ 0 & \text{иначе;} \end{cases}$$

$$w_{id} = \begin{cases} 1, & \text{если клиент } i \text{ обслуживается в день } d, \\ 0 & \text{иначе.} \end{cases}$$

Вспомогательные переменные $u_{ikd} \geqslant 0$ будут использованы для исключения подциклов. Математическая модель задачи частично-целочисленного линейного программирования может быть представлена следующим образом:

$$\min \sum_{d \in D} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ijkd} \tag{1}$$

при ограничениях

$$\sum_{i \in I} q_i y_{ikd} \leqslant v_k, \quad k \in K,\, d \in D, \tag{2}$$

$$y_{mkd} = \begin{cases} 1, & \text{если } m = m(k), \\ 0, & \text{если } m \neq m(k), \end{cases} \quad m \in M,\, k \in K,\, d \in D, \tag{3}$$

$$\sum_{k \in K} y_{ikd} = w_{id}, \quad i \in I,\, d \in D, \tag{4}$$

$$\sum_{d \in D} w_{id} = \mu_i, \quad i \in I, \tag{5}$$

$$\sum_{t=0}^{\tau_i - 1} w_{i(d+t)} = 1, \quad i \in I,\, d \in \{0, \ldots, (\mu_i - 1)\tau_i\}, \tag{6}$$

$$w_{i\alpha} + w_{i\beta} - 2 \leqslant y_{ik\alpha} - y_{ik\beta}, \quad i \in I,\, k \in K,\, \alpha, \beta \in D,\, \alpha \neq \beta, \tag{7}$$

$$\sum_{i \in V} x_{ijkd} = \sum_{i \in V} x_{jikd} = y_{jkd}, \quad j \in V,\, k \in K,\, d \in D, \tag{8}$$

$$u_{ikd} - u_{jkd} + n x_{ijkd} \leqslant n - 1, \quad i, j \in I,\, k \in K,\, d \in D, \tag{9}$$

$$\sum_{i \in V} \sum_{j \in V} x_{ijkd}(t_{ij} + s_j) \leqslant T, \quad k \in K,\, d \in D, \tag{10}$$

$$u_{ikd} \geqslant 0, \quad i \in I,\, k \in K,\, d \in D, \tag{11}$$

$$w_{id}, x_{ijkd}, y_{ikd} \in \{0, 1\}, \quad i, j \in V,\, k \in K,\, d \in D. \tag{12}$$

Целевая функция (1) задаёт суммарное пройденное расстояние для всех ТС на всём плановом периоде. Неравенства (2) контролируют грузоподъёмность ТС. Равенства (3) устанавливают распределение ТС по депо. Условия (4) и (5) гарантируют требуемую частоту посещения клиентов. Неравенства (6) задают необходимые временные интервалы между последовательными посещениями. Требование посещать клиента одним и тем же ТС записано в (7). Равенства (8) гарантируют каждому клиенту предшественника и преемника в маршруте ТС, а также выезд ТС и возвращение в депо. Неравенство (9) обеспечивает отсутствие подциклов. Длительность рабочей смены ограничена условием (10). Последние два условия задают область изменения переменных.

Заметим, что число переменных можно сократить заменой трёхиндексной матрицы переменных u_{ikd} на двухиндексную матрицу u_{id}.

Теорема 1. *Замена переменных u_{ikd} новыми переменными u_{id} и введение новых ограничений*

$$n w_{id} \geqslant u_{id} \geqslant 0, \quad i \in I,\, d \in D,$$

вместо условий (11) *не меняет множество допустимых маршрутов.*

Доказательство. Вспомогательные переменные u_{ikd} использованы в ограничениях (9) для запрета циклов, не проходящих через депо. Без ограничения общности можно считать, что они задают порядок обхода клиентов в каждом маршруте в каждый день планового периода. Индекс k задаёт номер маршрута; для каждого маршрута своя нумерация. Удаление этого индекса означает единую нумерацию посещения клиентов в каждый день планового периода: сначала нумеруются клиенты первого маршрута, затем второго и т.д. Условие $n w_{id} \geqslant u_{id}$ удаляет клиентов из этой нумерации, если они не посещаются в день d. Таким образом, множество маршрутов остаётся тем же самым. Теорема 1 доказана.

Задача (1)–(12) может не иметь допустимых решений в силу ограничений на парк ТС, их грузоподъёмность и длину рабочей смены. Внесём

ограничения (2) и (10) в целевую функцию со штрафами $\gamma_{kd} \geqslant 0$, $\lambda_{kd} \geqslant 0$, $k \in K$, $d \in D$. Получим релаксированную задачу

$$L(x, \gamma, \lambda) = \min \sum_{d \in D} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ijkd} + \sum_{d \in D} \sum_{k \in K} (\gamma_{kd} \kappa_{kd} + \lambda_{kd} \varepsilon_{kd}) \quad (13)$$

при условиях (3)–(9), (11), (12) и дополнительных ограничениях для новых переменных $\kappa_{kd}, \varepsilon_{kd} \geqslant 0$, которые для каждой пары (k, d) обозначают превышение грузоподъёмности и сверхурочные часы соответственно:

$$\kappa_{kd} \geqslant \sum_{i \in I} q_i y_{ikd} - v_k, \quad k \in K,\ d \in D, \quad (14)$$

$$\varepsilon_{kd} \geqslant \sum_{i \in V} \sum_{j \in V} x_{ijkd}(t_{ij} + s_j) - T, \quad k \in K,\ d \in D. \quad (15)$$

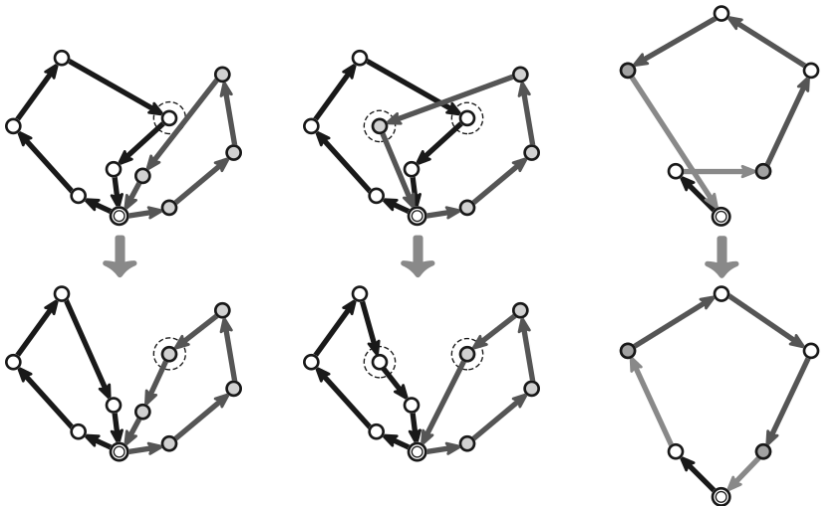
Релаксированная задача (3)–(9), (11)–(15) имеет допустимые решения даже в случае одного ТС в произвольном депо.

2. Окрестности

Локальный поиск является одним из перспективных направлений в области решения задач комбинаторной оптимизации [14]. Идеи локального поиска хорошо зарекомендовали себя при решении многих NP-трудных задач, в том числе VRP [15]. Для этих задач рассмотрено много разнообразных окрестностей [16–21]. Ниже представлены девять окрестностей для рассматриваемой задачи, которые учитывают частоту посещения клиентов и требования на интервалы между посещениями.

Представим решение задачи (3)–(9), (11)–(15) как совокупность последовательностей посещаемых клиентов для каждого ТС в каждый день. Последовательность посещаемых клиентов задаёт расписание ТС на каждый день. Решение задачи допустимо, если выполняются все ограничения, в частности, клиенты посещаются заданное число раз через равные промежутки времени одним и тем же ТС. Обозначим через σ допустимое решение задачи. Пару (k, d) назовём *перегруженной*, если ограничения (2) или (10) нарушаются для этой пары, и *неперегруженной* в противном случае. Будем стараться перестановками клиентов так перестроить решение, чтобы все пары стали неперегруженными, а значение целевой функции (1) было бы минимальным.

Определим следующие типы окрестностей для решения σ. Окрестность «Сдвиг» $N_{\text{move}}(\sigma)$ состоит из допустимых решений задачи, отличающихся от σ перемещением одного клиента к другому ТС в любой день или тому же ТС, но в другой день (рис. 1(а)). Если клиент должен быть посещён несколько раз, то перемещаются все его посещения. Кроме того, перемещения производятся только из перегруженных



(а) Сдвиг клиента (б) Замена двух клиентов (в) Устранение пересечений

Рис. 1. Типы окрестностей

пар в неперегруженные. При вставке клиента в маршрут выбирается наилучшая позиция без перестройки очерёдности посещения клиентов. Мощность этой окрестности равна $O(|I||D||K|)$. Это большое множество, поэтому также введём рандомизированный вариант этой окрестности $N^q_{\text{move}}(\sigma)$, $0 < q < 1$, который является случайной частью окрестности $N_{\text{move}}(\sigma)$. С этой целью каждый элемент окрестности $N_{\text{move}}(\sigma)$ включается в окрестность $N^q_{\text{move}}(\sigma)$ с вероятностью q независимо от других элементов.

Окрестность $\widetilde{N}^q_{\text{move}}(\sigma)$ имеет ту же структуру, но меньше ограничений на возможные пары для перемещения клиентов. Рассматриваются все варианты, кроме перемещений из неперегруженных пар в перегруженные.

Окрестность «Замена» $N_{\text{swap}}(\sigma)$ состоит из допустимых решений задачи, отличающихся от σ заменой двух клиентов с одинаковой частотой посещения для одного или разных ТС (рис. 1(б)). При этом рассматриваются только те клиенты, которые расположены достаточно близко, т. е. расстояние между ними не превышает заданного порога R, где R является параметром окрестности. Мощность данной окрестности равна $O(|I|^2)$, поэтому для неё аналогичным образом вводится рандомизированный вариант $N^q_{\text{swap}}(\sigma)$.

Окрестность $\widetilde{N}^q_{\text{swap}}(\sigma)$ имеет ту же структуру, но также включает решения с заменой клиентов с разной частотой посещения. Как следствие,

становятся возможными такие замены, как один клиент с четырьмя посещениями на двух клиентов с двумя посещениями или один клиент с двумя посещениями на двух клиентов с одним посещением и др.

Кроме вышеприведённых окрестностей будут использоваться большие окрестности типа Кернигана — Лина. Основная идея этих окрестностей близка к идее алгоритма поиска с запретами по малой окрестности, скажем, $N(\sigma)$. Окрестность $KL(\sigma)$ состоит из l решений, получаемых из σ по следующему правилу [10, 22].

(1) Найти наилучшее допустимое решение σ' в окрестности $N(\sigma)$.

(2) Положить $\sigma := \sigma'$, даже если решение σ' хуже по целевой функции, чем решение σ.

(3) Повторять шаги 1 и 2 l раз. Если перестановка уже использовалась на предыдущих шагах 1 и 2, то она не может быть использована снова.

Последовательность $\sigma_1, \ldots, \sigma_l$ задаёт l соседей решения σ. Будем говорить, что решение σ_b является локальным минимумом относительно KL-окрестности, если σ_b является лучшим решением из $\sigma_1, \ldots, \sigma_l$.

Используя четыре базовые окрестности $N^q_{\text{move}}(\sigma)$, $\widetilde{N}^q_{\text{move}}(\sigma)$, $N^q_{\text{swap}}(\sigma)$ и $\widetilde{N}^q_{\text{swap}}(\sigma)$ вместо окрестности $N(\sigma)$, получим четыре окрестности Кернигана — Лина $KL_{\text{move}}(\sigma)$, $\widetilde{KL}_{\text{move}}(\sigma)$, $KL_{\text{swap}}(\sigma)$ и $\widetilde{KL}_{\text{swap}}(\sigma)$ соответственно.

Как уже упоминалось выше, позиция нового клиента в расписании выбирается без изменения предыдущего порядка. Для улучшения конечного расписания применяется локальный спуск по окрестности *2-opt* для каждой пары (k, d). Идея построения этой окрестности состоит в удалении двух несмежных рёбер в цикле и добавлении двух других рёбер так, чтобы снова получился цикл. Её цель состоит в устранении пересечений рёбер (рис. 1(в)). Другими словами, на данном этапе задача разбивается на $|K||D|$ подзадач одного коммивояжёра, а затем для каждой из этих задач независимо выполняется локальный спуск по окрестности *2-opt* [23].

3. Локальный поиск

Поиск с чередующимися окрестностями является эффективным методом локального поиска, предложенным более 20 лет назад П. Хансеном и Н. Младеновичем [8]. Идея метода состоит в систематическом переходе от одной окрестности к другой. Этот подход хорошо зарекомендовал себя при решении многих прикладных задач [9], включая задачи маршрутизации [24, 25] и размещения [26–28]. В основе данного подхода лежит очевидное утверждение, что локальный минимум для одной окрестности может не быть локальным минимумом для другой. Используя разнородные окрестности, можно легко переходить от одного локального оптимума

к другому и в результате получать решения достаточно высокого качества. Ниже приводится описание этого подхода к решению релаксированной задачи.

3.1. Общая схема алгоритма. Приведённая ниже гибридная схема алгоритма является расширенным вариантом базовой схемы VNS [8]. Предложенный алгоритм использует идею метода штрафов для получения допустимых маршрутов и усилен окрестностями Кернигана — Лина, в определении которых неявно используется поиск с запретами по рандомизированным окрестностям различной структуры. Степень рандомизации окрестностей может быть различна и $q_i \neq q_j$, $1 \leqslant i \neq j \leqslant 4$.

Алгоритм 1 (гибридный алгоритм VNS)

Вход: Задать критерий остановки и правила интенсификации.

1: Построить начальное решение x. Выбрать параметры λ_{kd}, γ_{kd}, $q_1, \ldots, q_4$, l, R и системы окрестностей
2: **while** не достигнут критерий остановки **do**
3: Встряска. Случайно сгенерировать решение x' в окрестности x
4: Задать $x := x'$
5: Локальный поиск
6: Применить локальный поиск по базовым окрестностям
7: Применить локальный спуск по KL-окрестностям
8: Применить локальный спуск по окрестности *2-opt* для каждой пары (k, d)
9: Обозначить через x'' лучшее по целевой функции решение из найденных в течение шагов 6–8
10: **if** $L(x'', \gamma, \lambda) < L(x, \gamma, \lambda)$ **then**
11: Задать $x := x''$
12: **end if**
13: **if** не происходило изменения x в течение p итераций **then**
14: Интенсификация и обновление штрафов
15: **if** произошло уже h процедур интенсификации для x **then**
16: Перейти на шаг 2
17: **end if**
18: **end if**
19: Перейти на шаг 5
20: **end while**
21: Применить локальный спуск по окрестностям «Замена» с $q = 1$
22: Применить локальный спуск по окрестности *2-opt* для каждой пары (k, d)

Выход: Предъявить лучшие найденные решения

Эти величины выбираются таким образом, чтобы мощности всех рандомизированных окрестностей были примерно одинаковыми и в среднем равны 200. Такой приём позволяет ускорить поиск, уменьшая время выполнения одной итерации, и добавляет необходимую диверсификацию и разнообразие в процесс поиска.

Алгоритм начинает свою работу с построения начального решения релаксированной задачи и определения штрафов. Начальное решение может быть произвольным, но в данной работе используется жадная эвристика с равномерным распределением клиентов по всем парам (k, d), $k \in K$, $d \in D$. Сначала распределяются клиенты с высокой частотой посещения. Стремимся минимизировать максимальное число клиентов в день у одного ТС [5]. Всем штрафам присваиваются одинаковые начальные значения $\lambda_{kd} = \lambda$, $\gamma_{kd} = \gamma$.

На каждой итерации локального поиска выбирается окрестность, после чего происходит переход от текущего решения к лучшему найденному в окрестности решению. Для окрестностей Кернигана — Лина генерируется l решений и из них выбирается лучшее.

Критерием остановки (шаг 2) может являться общее число итераций, которое зависит от числа клиентов и частоты их посещения. Мы используем $O\big(n_1^2\big)$ итераций в наших экспериментах, где $n_1 = \sum\limits_{i \in I} \mu_i$. Также критерием остановки может являться достижение нулевых суммарных штрафов (допустимое решение исходной задачи), максимального числа процедур встряски или общее время работы алгоритма.

Локальный поиск (шаг 5) применяется по окрестностям «Сдвиг» и «Замена» и затем по окрестностям Кернигана — Лина. В последнем случае совершаются только локальные улучшения и при достижении локального оптимума процесс прекращается. Для четырёх базовых окрестностей процесс поиска прекращается после совершения заранее установленного числа итераций, а переход в лучшее найденное в окрестности решение производится независимо от того, хуже оно или лучше, с последующей сменой окрестности. Далее (шаг 8) происходит локальный поиск по окрестности *2-opt* для каждой пары (k, d). Если решение на данном этапе оказывается допустимым для исходной задачи, т. е. $\varepsilon_{kd} = 0$, $\kappa_{kd} = 0$ для всех пар (k, d), то далее при выполнении процедуры интенсификации (шаг 14) штрафы уменьшаются. Иначе во время данной процедуры штрафы увеличиваются. В общем случае штрафы изменяются по следующему правилу:

$$\lambda_{kd} := \lambda_{kd}\delta(\varepsilon_{kd}), \quad \gamma_{kd} := \gamma_{kd}\theta(\kappa_{kd}), \quad k \in K,\, d \in D,$$

где $\delta(\varepsilon_{kd}) = 0{,}97$, если $\varepsilon_{kd} = 0$, и $\delta(\varepsilon_{kd}) = 1{,}03$, если $\varepsilon_{kd} > 0$, а $\theta(\kappa_{kd}) = 0{,}97$, если $\kappa_{kd} = 0$, и $\theta(\kappa_{kd}) = 1{,}04$, если $\kappa_{kd} > 0$.

3.2. Интенсификация поиска. В процедуре интенсификации происходит возвращение в локальный рекорд, обозначенный через x в псевдокоде выше, изменение значений штрафов $\lambda_{kd}, \gamma_{kd}$ и увеличение параметров рандомизации окрестностей $q_1, \ldots, q_4$, для того чтобы исследовать наиболее многообещающие области более тщательно. Если находится решение лучшее, чем текущий локальный рекорд, то он обновляется, а значения параметров рандомизации окрестностей возвращаются к первоначальным значениям. При этом процедура интенсификации совершается только при условии, что локальный рекорд не обновлялся заданное число итераций.

При обновлении локального рекорда (шаги 10–11) для сравнения решений используется целевая функция (13) с текущими значениями штрафов.

3.3. Диверсификация поиска. Процедура встряски (шаг 3), так же как и рандомизация поиска, введена для диверсификации поиска. Её идея заключается в переходе в случайное решение из некоторой окрестности. Для реализации этой процедуры использовались случайные операции сдвигов и замен клиентов всех типов. Встряска совершается, только если уже произошло предустановленное число возвращений в один локальный рекорд. После встряски параметры $q_1, \ldots, q_4$ возвращаются к своим начальным значениям, а локальный рекорд становится равным текущему решению с целью исследования новой области решений (шаг 4).

3.4. Постоптимизация и выдача результатов. На каждой итерации работы алгоритма независимо от локальных рекордов сохраняются наилучшие найденные решения. Они включают в себя решение с наименьшим значением суммарной невязки $\sum\limits_{kd}(\varepsilon_{kd}+\kappa_{kd})$, решение с наименьшим суммарным расстоянием и промежуточные несравнимые решения.

В конце алгоритма для каждого сохранённого решения совершаются локальные спуски по окрестностям N_{swap} и $\widetilde{N}_{\text{swap}}$ ($q_3 = q_4 = 1$) и *2-opt* для каждой пары (k, d) (шаги 21 и 22). При этом значения штрафов устанавливаются достаточно высокими для возвращения в допустимую область исходной задачи: $\lambda_{kd} = 40$, $\gamma_{kd} = 50$.

4. Численные эксперименты

Описанный выше алгоритм реализован на языке C++ с использованием компилятора MSVC++ 14.16 со стандартными release параметрами. Эксперименты по оценке эффективности алгоритма проводились на компьютере с процессором AMD Ryzen 5 2600 3,4 ГГц и 16 ГБ ОЗУ, работающем под операционной системой Microsoft Windows 10 (64-bit).

Реализованная программа является многопоточной, что ускорило её выполнение.

Эксперименты с пакетом Gurobi (версия 8.1.1) для модели из разд. 1 и C++ интерфейсом производились на компьютере с процессором Intel Xeon X5675 3,07 ГГц и 32 ГБ ОЗУ, работающем под операционной системой Microsoft Windows Server 2008 R2 (64-bit).

4.1. Исходные данные. Данные для тестирования алгоритма были предоставлены одной российской логистической компанией и содержат геоданные 892 клиентов Оренбургской области. Среди них одна треть клиентов требуют одного посещения в течение четырёх недель, чуть более половины клиентов надо посещать два раза, а остальные клиенты должны посещаться четыре раза. Имеется три депо, расположенных на расстоянии около 250 км друг от друга. Для получения тестовых примеров меньшей размерности из вышеописанного большого примера выбиралась случайная часть. Для этого варьировались определённые параметры, среди которых число, необходимое для локализации депо (радиус), вероятности отбора клиента, разные для разных депо, и число, задающее случайный сдвиг ТС относительно их начального положения. Для отбора клиентов и далее в алгоритме использовался генератор псевдослучайных чисел Mersenne Twister (32-bit). В результате было сгенерировано 10 различных примеров с числом клиентов 620–670. Этот диапазон позволяет получить разнообразные большие примеры с одинаковым количеством ТС во всех депо. Кроме того, этот диапазон близок к реальному числу обслуживаемых компанией клиентов в одном из регионов. Для данных примеров было назначено по два ТС в каждое депо.

Также для генерации меньших примеров был использован пример с 670 клиентами из Оренбургской области. Чуть менее трёх четвертей из них имеют частоту посещения 1, а число клиентов частоты 2 и 4 примерно равны. Для этого примера имеются те же три депо, что и ранее. Использовав тот же приём, что и для примера с 892 клиентами, были сгенерированы примеры с 300–350, 130–150, 41–55 и 18–34 клиентами. Для примеров с 300–350 клиентами было назначено по одному ТС в каждое из трёх депо, а для остальных — одно ТС в некоторое депо.

Для задач размерности 300–350 были сгенерированы два набора примеров: для которых решения без штрафов находятся довольно легко и для которых это сделать сложно, но возможно. Эти наборы были получены за счёт изменения грузоподъёмности ТС. В лёгких примерах она составляла [400,500], а в более сложных [320,340]. Такие размеры позволяют находить решения без превышения грузоподъёмности.

Данные клиентов включают название, GPS координаты (широта и долгота), частоту посещения, время обслуживания и величину запроса. Рабочая смена составляет 8 часов, включая 40 минут на перерыв.

Время перерыва не зафиксировано в расписании ТС, так что они могут поставить его в любое свободное от обслуживания клиентов время своего рабочего дня. Рассматриваемая нами проблема не включает в себя временные окна. Тем самым для внесения в модель перерывов продолжительность рабочей смены может быть сокращена до $T = 7$ ч 20 мин.

В данной задаче ТС могут покидать депо начиная с 8:30, но должны прибыть к первому клиенту не ранее 9:00. Последний клиент должен быть обслужен до 17:00, но ТС должно вернуться в депо не позднее 18:00. Для того чтобы включить эти дополнительные требования в модель, которая считает, что выезд из депо происходит у всех одновременно в 9:00, мы скорректировали матрицу (t_{ij}) для дуг, связывающих депо и клиентов, по следующему правилу:

$$t_{ij} = \begin{cases} t_{ij}, & \text{если } i, j \in I, \\ \max\{0, t_{ij} - 30'\}, & \text{если } i \in M,\ j \in I, \\ \max\{0, t_{ij} - 60'\}, & \text{если } i \in I,\ j \in M. \end{cases}$$

То, что матрица (t_{ij}) теперь несимметрична, не мешает применению окрестности *2-opt*, так как стоимость дуг, связанных с депо, пересчитывается, а остальные дуги симметричны.

Плановый период составляет 20 дней. Скорость каждого ТС равна 54 км/ч. Для окрестностей Кернигана — Лина параметр l был установлен равным 25. Порог R для окрестностей «Замена» равен 20 км. Количество итераций, в течение которых выполняется поиск по базовым окрестностям (шаг 6), установлен равным 1300 итерациям. Количество итераций для возвращения в локальный рекорд равно $p = 3000$. Процедура встряски осуществлялась после $h = 3$ возвращений в рекорд. Доля рассматриваемой окрестности q во время процедуры интенсификации увеличивалась вдвое, но не превышала 0,6.

4.2. Особенности реализации алгоритма. Первый эксперимент направлен на выбор схемы изменения штрафов. Сравнение проводилось на примерах с 320–350 клиентами. Каждый пример решался 10 раз по 5 минут на каждый расчёт.

Исследовалось восемь вариантов схемы изменения штрафов. Результаты представлены в табл. 1, где приведены средние относительные погрешности по сравнению с базовой стратегией 5. Верхняя строчка соответствует лёгким примерам, нижняя — сложным. Для стратегий 1–3 значения штрафов изменяются каждый раз после завершения шагов 6–8 локального поиска. Для стратегий 5–8 изменение происходит только во время процедуры интенсификации поиска (шаг 14). Стратегия 4 соответствует фиксированным значениям штрафов, которые не изменяются в течение всей работы алгоритма. Для вариантов 1–2 и 6–7 решение

Таблица 1

Сравнение стратегий изменения штрафов

1	2	3	4	5	6	7	8
+1,14	+1,41	+1,36	+0,46	+0,00	+0,94	+0,69	+0,66
−0,53	+3,24	+3,23	+2,36	+0,00	−2,67	−0,91	−1,06

о том, уменьшать или увеличивать штрафы, принимается на основе значений ε и κ так, как описано в п. 3.1. При этом для стратегий 1 и 6 значения ε и κ смотрятся у решения, которое было достигнуто к моменту совершения процедуры смены штрафов. Для стратегий 2 и 7 данные значения рассматриваются у решения с наименьшими нарушениями, найденного с момента предыдущей процедуры обновления штрафов. Для вариантов 3 и 8 решение об изменении принимается на основе того, каких решений было найдено больше с момента предыдущей процедуры обновления штрафов, с нарушениями или без них. Для стратегии 5 значения штрафов увеличиваются до нахождения решения без нарушений, после чего они уменьшаются до некоторого предельного значения.

Можно сказать, что для простых примеров разница между стратегиями совсем не велика. Для более сложных предпочтительна стратегия, изменяющая штрафы во время интенсификации поиска. В дальнейших численных экспериментах будет использоваться стратегия 5 и сложные примеры с 300–350 клиентами.

Следующий эксперимент направлен на оценку вклада разных окрестностей в итоговую эффективность алгоритма. Применение окрестности *2-opt* позволяет получать решения в среднем на 0,96% лучшие, чем без её использования. Изменение доли рассматриваемой окрестности приводит к падению целевой функции в среднем на 1,18%. Удаление окрестностей $\widetilde{N}^q_{\mathrm{move}}(\sigma)$ или $N^q_{\mathrm{swap}}(\sigma)$ ведёт к сильному ухудшению результатов, более 8%. Исключение окрестности $\widetilde{N}^q_{\mathrm{swap}}(\sigma)$ ведёт к ухудшению результатов примерно на 3%. Варианты алгоритма без процедуры встряски (диверсификация вносится за счёт рандомизации окрестности и перехода в худшие по целевой функции решения) и при совершении встряски после каждого возвращения в локальный рекорд являются менее эффективными, чем выбранная.

4.3. Сравнение с другими подходами. Используя модель частично-целочисленного линейного программирования, можно получить оптимальное решение задачи методом ветвей и границ (решатель Gurobi). К сожалению, такой подход работает только на примерах малой размерности. В экспериментах для 18–34 и 40–55 клиентов проведено сравнение результатов Gurobi и разработанного гибридного алгоритма VNS.

Таблица 2

Сравнение с Gurobi, $|D| = 8$**,** $n = 18$–34

#	Gurobi	VNS			gap (%)
		мин.	сред.	макс.	
1	1854,82	1850,64	1850,64	1850,64	**−0,23**
2	1481,40	1470,03	1471,44	1472,38	**−0,68**
3	1136,65	1131,60	1132,10	1132,55	**−0,40**
4	309,43	309,43	309,43	309,43	**−0,00**
5	1340,47	1337,39	1337,39	1337,39	**−0,23**
6	358,52	354,42	354,58	355,04	**−1,11**
7	601,15	579,86	580,15	582,19	**−3,62**
8	1038,51	1038,19	1038,19	1038,19	**−0,03**
9	261,33	256,61	256,61	256,61	**−1,84**
10	1116,53	1111,51	1111,51	1111,51	**−0,45**

Для Gurobi установлено ограничение по времени в 1 час, а для алгоритма VNS — 40 секунд при решении примеров на 18–34 клиентов и 60 секунд для примеров на 40–55 клиентов. Результаты экспериментов представлены в табл. 2 и 3. Столбец «gap» показывает разницу между решением Gurobi и средним результатом для алгоритма VNS.

Даже для примеров столь малой размерности Gurobi не удалось доказать оптимальность ни одного из найденных решений. В девяти примерах из десяти алгоритм VNS нашёл решения на 0,86% лучше, чем Gurobi. В одном из примеров найденные решения совпали. Для примеров с 40–55 клиентами разница между методами составила 2,82%. Результат подчёркнут, если после каждого запуска найдено одно и то же решение.

Таблица 3

Сравнение с Gurobi, $|D| = 12$**,** $n = 40$–55

#	Gurobi	VNS			gap (%)
		мин.	сред.	макс.	
1	2742,88	2658,79	2660,31	2663,15	**−3,10**
2	2080,50	2047,59	2052,26	2055,67	**−1,38**
3	1260,69	1224,44	1224,91	1225,24	**−2,92**
4	442,51	425,29	427,14	430,10	**−3,60**
5	1508,92	1485,41	1486,82	1488,56	**−1,49**
6	1194,99	1130,43	1133,51	1138,79	**−5,42**
7	1418,01	1376,86	1377,58	1379,70	**−2,94**
8	2056,56	2036,50	2038,38	2045,84	**−0,89**
9	476,54	453,79	453,79	453,79	**−5,01**
10	1873,43	1844,30	1846,63	1848,83	**−1,45**

При решении примеров размерности 130–150 за час работы Gurobi удавалось найти решение в несколько раз хуже решения VNS, поэтому для больших размерностей сравнение проводилось с базовой схемой VNS. Она отличается от уже представленной схемы тем, что в локальном поиске совершаются только улучшающие переходы, отсутствуют процедуры интенсификации, а изменения штрафов и окрестности для процедуры встряски проводятся каждый раз, когда очередной локальный оптимум оказался хуже уже имеющегося решения. При этом для неё использовались те же окрестности, включая окрестности Кернигана — Лина.

Таблица 4

Сравнение с базовой схемой, $|D| = 20$**,** $n = 130$–150

#	Базовый VNS		Гибридный VNS		gap (%)
	мин.	сред.	мин.	сред.	
1	3184	3194,2	3201	3208,6	+0,45
2	3383	3385,4	3384	3390,7	+0,16
3	2730	2738,6	2738	2756,8	+0,66
4	1617	1651	1656	1663,5	+0,76
5	2977	3010,3	3047	3061,9	+1,71
6	2102	2110,5	2110	2113,5	+0,14
7	2816	2817,3	2818	2819,1	+0,06
8	2948	2964,2	2960	2983,2	+0,63
9	2877	2879	2880	2883,2	+0,15
10	2987	3063,1	3071	3091,5	+0,93

Результаты экспериментов представлены в табл. 4–6. Для примеров с 130–150 клиентами время работы обоих алгоритмов было ограничено 2 минутами. Минимальное и среднее значения за 30 испытаний алгоритма представлены в табл. 4. При такой размерности алгоритмы показали примерно одинаковые результаты. Базовый метод VNS находил решения в среднем на 0,57% лучше, чем разработанная гибридная схема.

Примеры с 320–350 клиентами были несколько труднее, и уже не всегда получалось найти решение без переработки. Для сравнения результатов использовалась целевая функция (13) со значениями штрафов $\lambda_{kd} = 40$. Значения суммарного сверхурочного времени ε, приведённые в таблицах ниже, указаны в минутах. Различие между решениями, найденными методами, составляет около 2%. Аналогичное различие сохранялось и при других размерностях (см. табл. 5 и 6). С ростом числа клиентов преимущество гибридной схемы растёт.

Заключение

Представлена новая модель оптимизации маршрутов транспортных средств с многократным посещением клиентов. Для её решения

Таблица 5

Сравнение с базовой схемой, $|D| = 20$**,** $n = 320$–350

#	Базовый VNS				Гибридный VNS				gap (%)
	мин.		сред.		мин.		сред.		
	ε	расст.	ε	расст.	ε	расст.	ε	расст.	
1	0	10513	0,4	10672,4	0	10463	0	10482,8	**−1,92**
2	0	10537	12,9	10789	1	10466	0,5	10663,5	**−5,50**
3	22	10218	17,8	10494,8	0	10832	0	10947,3	**−2,32**
4	2	9967	3,7	10202,9	4	9931	3,3	10048,8	**−1,64**
5	0	8791	0,2	8953,7	0	8856	0,4	9060,1	+1,28
6	0	8887	1,1	9254,5	0	8844	0	8890	**−4,39**
7	0	9768	0,2	10074	0	9832	0	9863,9	**−2,16**
8	0	9219	0,2	9426,9	0	9179	1,6	9256,3	**−1,21**
9	0	11081	16,1	11160	0	10691	1,6	11091,3	**−5,50**
10	0	10216	3	10167	0	10214	0,7	10226,6	**−0,31**

разработан гибридный алгоритм локального поиска с чередующимися окрестностям, использующий метод штрафов для расширения области поиска. Алгоритм способен решать примеры большой размерности за сравнительно небольшое время на персональном компьютере. Для повышения эффективности алгоритма применяются вероятностные окрестности, а также процедуры интенсификации и диверсификации поиска. Алгоритм реализован на языке C++ и тестировался на реальных примерах российской логистической компании.

Одним из направлений дальнейших исследований является совершенствование математической модели, например, путём введения временны́х

Таблица 6

Сравнение с базовой схемой, $|D| = 20$**,** $n = 620$–670

#	Базовый VNS				Гибридный VNS				gap (%)
	мин.		сред.		мин.		сред.		
	ε	расст.	ε	расст.	ε	расст.	ε	расст.	
1	0	16002	0	16332,1	0	15422	0	15891,9	**−2,70**
2	0	16370	0,5	16695,6	0	15920	0	16203,7	**−3,06**
3	4	16506	21,6	16950,1	4	16534	8,2	16768,7	**−4,02**
4	12	16570	1,2	17499,7	0	17083	0	17196,2	**−2,00**
5	1	15455	0,9	16186,2	0	15317	2	15657,5	**−2,99**
6	0	14509	2	14696,4	4	14164	0,9	14602,4	**−0,93**
7	0	15745	0,2	16310,1	0	15420	0,1	15728,8	**−3,59**
8	0	16588	6,8	17143,2	0	16474	0,2	16846,7	**−3,22**
9	27	16222	37,5	16349,5	0	17249	11	17342,8	**−0,37**
10	1	17140	2,1	17658,8	1	17300	4,1	17644	+0,37

окон для посещения клиентов. Логистическая компания может получить дополнительное конкурентное преимущество, если клиенты будут посещаться в удобное для них время. Кроме того, представляет несомненный интерес учёт загруженности дорог при построении маршрутов. Скорость передвижения может сильно зависеть от времени суток, и это обстоятельство должно учитываться при решении задачи.

ЛИТЕРАТУРА

1. **Dantzig G. B., Ramser J. H.** The truck dispatching problem // Manage. Sci. 1959. Vol. 6, No. 1. P. 80–91.
2. **Salhi S., Imran A., Wassan N. A.** The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation // Comput. Oper. Res. 2014. Vol. 52. P. 315–325.
3. **Christofides N., Beasley J. E.** The period routing problem // Networks. 1984. Vol. 14, No 2. P. 237–256.
4. **Kovacs A. A., Parragh S. N., Hartl R. F.** A template-based adaptive large neighborhood search for the consistent vehicle routing problem // Networks. 2014. Vol. 63, No. 1. P. 60–81.
5. **Gaudioso M., Paletta G.** A Heuristic for the Periodic Vehicle Routing problem // Transport. Sci. 1992. Vol. 26, No. 2. P. 86–92.
6. **Groër C., Golden B., Wasil E.** The consistent vehicle routing problem // Manufacturing & Service Operations Manage. 2009. Vol. 11, No. 4. P. 630–643.
7. **Talbi E.-G.** Metaheuristics: From design to implementation. Hoboken, NJ: Wiley, 2009.
8. **Mladenović N., Hansen P.** Variable neighborhood search // Comput. Oper. Res. 1997. Vol. 24. P. 1097–1100.
9. **Hansen P., Mladenović N.** Developments of variable neighborhood search // Essays and surveys in metaheuristics. New York: Springer, 2002. P. 415–439.
10. **Kernighan B. W., Lin S.** An efficient heuristic procedure for partitioning graphs // Bell Syst. Tech. J. 1970. Vol. 49, No. 2. P. 291–307.
11. **Кононова П. А., Кочетов Ю. А.** Локальный поиск с чередующимися окрестностями для задачи Джонсона с пассивным буфером // Дискрет. анализ и исслед. операций. 2012. Т. 19, № 5. С. 63–82.
12. **Kulachenko I. N., Kononova P. A.** The VNS approach for a consistent capacitated vehicle routing problem under the shift length constraints // Commun. Comput. Inform. Sci. 2019. Vol. 1090. P. 51–67.
13. **Kulachenko I. N., Kononova P. A., Kochetov Yu. A., Kurochkin A. A.** The variable neighborhood search for a consistent vehicle routing problem under the shift length constraints // IFAC-PapersOnLine. 2019. Vol. 52, No. 13. P. 2314–2319.
14. **Aarts E. H. L., Lenstra J. K.** Local search in combinatorial optimization Chichester: Wiley, 1997.
15. The vehicle routing problem: Latest advances and new challenges. New York: Springer, 2008.

16. **Grover L. K.** Local search and the local structure of NP-complete problems // Oper. Res. Lett. 1992. Vol. 12, No. 4. P. 235–243.
17. **Alekseeva E. V., Kochetov Yu. A., Plyasunov A. A.** Complexity of local search for the p-median problem // Eur. J. Oper. Res. 2008. Vol. 191, No. 3. P. 736–752.
18. **Gutin G. Z., Yeo A.** Small diameter neighbourhood graphs for the traveling salesman problem: at most four moves from tour to tour // Comput. Oper. Res. 1994. Vol. 26, No. 4. P. 321–327.
19. **Ahuja R. K., Ergun Ö., Orlin J. B., Punnen A. P.** A survey of very large-scale neighborhood search techniques // Discrete Appl. Math. 2002. Vol. 123, No. 1–3. P. 75–102.
20. **Кочетов Ю. А.** Вычислительные возможности локального поиска в комбинаторной оптимизации // Журн. вычисл. математики и мат. физики. 2008. Т. 48, № 5. P. 788–807.
21. **Toth P., Vigo D.** Vehicle routing: Problems, methods, and applications. Philadelphia, PA: SIAM, 2014.
22. **Kochetov Yu. A., Kononova P. A., Paschenko M. G.** Formulation space search approach for the teacher/class timetabling problem // Yugoslav J. Oper. Res. 2008. Vol. 18, No. 1. P. 1–11.
23. **Hemmelmayr V. C., Doerner K. F., Hartl R. F.** A variable neighborhood search heuristic for periodic routing problems // Eur. J. Oper. Res. 2009. Vol. 195, No. 3. P. 791–802.
24. **Кочетов Ю. А., Хмелёв А. В.** Гибридный алгоритм локального поиска для задачи маршрутизации разнородного ограниченного автопарка // Дискрет. анализ и исслед. операций. 2015. Т. 22, № 5. С. 5–29.
25. **Khmelev A. V., Kochetov Yu. A.** A hybrid VND method for the split delivery vehicle routing problem // Electron. Notes Discrete Math. 2015. Vol. 47. P. 5–12.
26. **Davydov I. A., Kochetov Yu. A., Carrizosa E.** A local search heuristic for the $(r|p)$-centroid problem in the plane // Comput. Oper. Res. 2014. Vol. 52. P. 334–340.
27. **Diakova Z. S., Kochetov Yu. A.** A double VNS heuristic for the facility location and pricing problem // Electron. Notes Discrete Math. 2012. Vol. 39. P. 29–34.
28. **Davydov I. A., Kochetov Yu. A., Carrizosa E.** VNS heuristic for the $(r|p)$-centroid problem on the plane // Electron. Notes Discrete Math. 2012. Vol. 39. P. 5–12.

Кулаченко Игорь Николаевич
Кононова Полина Александровна

Статья поступила
31 октября 2019 г.
После доработки —
27 декабря 2019 г.
Принята к публикации
19 февраля 2020 г.

DISKRETNYI ANALIZ I ISSLEDOVANIE OPERATSII
/DISCRETE ANALYSIS AND OPERATIONS RESEARCH/
April–June 2020. Vol. 27, No. 2. P. 43–64

UDC 519.8+518.25 DOI 10.33048/daio.2020.27.673

A HYBRID LOCAL SEARCH ALGORITHM FOR CONSISTENT PERIODIC VEHICLE ROUTING PROBLEM

I. N. Kulachenko[1,a] and *P. A. Kononova*[2,b]

[1] Novosibirsk State University,
2 Pirogov Street, 630090 Novosibirsk, Russia

[2] Sobolev Institute of Mathematics SB RAS,
4 Acad. Koptyug Avenue, 630090 Novosibirsk, Russia

E-mail: [a] soge.ink@gmail.com, [b] pkononova@math.nsc.ru

Abstract. Under consideration is some new real-world application of vehicle routing planning in a finite time horizon. Let a company have a set of capacitated vehicles in depots and serve a set of customers. There is a frequency for each customer which describes how often the customer should be visited. Time intervals between two consecutive visits are fixed, but the visiting schedule is flexible. To obtain competitive advantages, the company tries to increase the service quality. To this end, each customer should be visited by one driver only. The goal is to minimize the total length of the vehicle paths over the planning horizon under the frequency constraints and driver shift length constraints.

We present a mixed-integer linear programming model for this new consistent capacitated vehicle routing problem. To find near optimal solutions, we design the variable neighborhood search metaheuristic with several neighborhood structures. The driver shift length and capacitated constraints are penalized and included into the objective function. Some numerical results for the real-test cases are discussed. Tab. 6, illustr. 1, bibliogr. 28.

Keywords: penalty method, metaheuristic, Kernighan–Lin neighborhood, vehicle of limited capacity.

This research is supported by the Russian Foundation for Basic Research and Novosibirsk region (Project 19–47–540005).

English version: Journal of Applied and Industrial Mathematics **14** (2), 339–351 (2020), DOI 10.1134/S199047892002012X.

REFERENCES

1. **G. B. Dantzig** and **J. H. Ramser,** The truck dispatching problem, *Manage. Sci.* **6** (1), 80–91 (1959).
2. **S. Salhi, A. Imran,** and **N. A. Wassan,** The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation, *Comput. Oper. Res.* **52**, 315 – 325 (2014).
3. **N. Christofides** and **J. E. Beasley,** The period routing problem, *Networks* **14** (2), 237–256 (1984).
4. **A. A. Kovacs, S. N. Parragh,** and **R. F. Hartl,** A template-based adaptive large neighborhood search for the consistent vehicle routing problem, *Networks* **63** (1), 60–81 (2014).
5. **M. Gaudioso** and **G. Paletta,** A heuristic for the periodic vehicle routing problem, *Transp. Sci.* **26** (2), 86–92 (1992).
6. **C. Groër, B. Golden,** and **E. Wasil,** The consistent vehicle routing problem, *Manuf. Serv. Oper. Manage.* **11** (4), 630–643 (2009).
7. **E.-G. Talbi,** *Metaheuristics: From design to implementation* (Wiley, Hoboken, NJ, 2009).
8. **N. Mladenović** and **P. Hansen,** Variable neighborhood search, *Comput. Oper. Res.* **24**, 1097–1100 (1997).
9. **P. Hansen** and **N. Mladenović,** Developments of variable neighborhood search, in *Essays and Surveys in Metaheuristics* (Springer, New York, 2002), pp. 415–439.
10. **B. W. Kernighan** and **S. Lin,** An efficient heuristic procedure for partitioning graphs, *Bell Syst. Tech. J.* **49** (2), 291–307 (1970).
11. **P. A. Kononova** and **Yu. A. Kochetov,** The variable neighborhood search for the two machine flow shop problem with a passive prefetch, *Diskretn. Anal. Issled. Oper.* **19** (5), 63–82 (2012) [Russian] [*J. Appl. Ind. Math.* **7** (1), 54–67 (2013)].
12. **I. N. Kulachenko** and **P. A. Kononova,** The VNS approach for a consistent capacitated vehicle routing problem under the shift length constraints, *Commun. Comput. Inf. Sci.* **1090**, 51–67 (2019).
13. **I. N. Kulachenko, P. A. Kononova, Yu. A. Kochetov,** and **A. A. Kurochkin,** The variable neighborhood search for a consistent vehicle routing problem under the shift length constraints, *IFAC-PapersOnLine* **52** (13), 2314–2319 (2019).
14. **E. H. L. Aarts** and **J. K. Lenstra,** *Local Search in Combinatorial Optimization* (Wiley, Chichester, 1997).
15. *The Vehicle Routing Problem: Latest Advances and New Challenges* (Springer, New York, 2008).
16. **L. K. Grover,** Local search and the local structure of NP-complete problems, *Oper. Res. Lett.* **12** (4), 235–243 (1992).
17. **E. V. Alekseeva, Yu. A. Kochetov,** and **A. A. Plyasunov,** Complexity of local search for the p-median problem, *Eur. J. Oper. Res.* **191** (3), 736–752 (2008).

18. G. Z. Gutin and **A. Yeo,** Small diameter neighbourhood graphs for the traveling salesman problem: At most four moves from tour to tour, *Comput. Oper. Res.* **26** (4), 321–327 (1999).

19. R. K. Ahuja, Ö. Ergun, J. B. Orlin, and **A. P. Punnen,** A survey of very large-scale neighborhood search techniques, *Discrete Appl. Math.* **123** (1–3), 75–102 (2002).

20. Yu. A. Kochetov, Computational bounds for local search in combinatorial optimization, *Zh. Vychisl. Mat. Mat. Fiz.* **48** (5), 788–807 (2008) [Russian] [*Comput. Math. Math. Phys.* **48** (5), 747–763 (2008)].

21. P. Toth and **D. Vigo,** *Vehicle Routing: Problems, Methods, and Applications* (SIAM, Philadelphia, PA, 2014).

22. Yu. A. Kochetov, P. A. Kononova, and **M. G. Paschenko,** Formulation space search approach for the teacher/class timetabling problem, *Yugosl. J. Oper. Res.* **18** (1), 1–11 (2008).

23. V. C. Hemmelmayr, K. F. Doerner, and **R. F. Hartl,** A variable neighborhood search heuristic for periodic routing problems, *Eur. J. Oper. Res.* **195** (3), 791–802 (2009).

24. Yu. A. Kochetov and **A. V. Khmelev,** A hybrid algorithm of local search for the heterogeneous fixed fleet vehicle routing problem, *Diskretn. Anal. Issled. Oper.* **22** (5), 5–29 (2015) [Russian] [*J. Appl. Ind. Math.* **9** (4), 503–518 (2015)].

25. A. V. Khmelev and **Yu. A. Kochetov,** A hybrid VND method for the split delivery vehicle routing problem, *Electron. Notes Discrete Math.* **47**, 5–12 (2015).

26. I. A. Davydov, Yu. A. Kochetov, and **E. Carrizosa,** A local search heuristic for the $(r|p)$-centroid problem in the plane, *Comput. Oper. Res.* **52**, 334–340 (2014).

27. Z. S. Diakova and **Yu. A. Kochetov,** A double VNS heuristic for the facility location and pricing problem, *Electron. Notes Discrete Math.* **39**, 29–34 (2012).

28. I. A. Davydov, Yu. A. Kochetov, and **E. Carrizosa,** VNS heuristic for the $(r|p)$-centroid problem on the plane, *Electron. Notes Discrete Math.* **39**, 5–12 (2012).

Igor N. Kulachenko
Polina A. Kononova

Received October 31, 2019
Revised December 27, 2019
Accepted February 19, 2020