Обзор методов решения задачи о приёмке и доставке с временными ограничениями. Часть I: точный подход

Н.В. Царьков, Д.Ю. Голембиовский

***Аннотация* — В статье рассматривается точный подход на основе метода ветвей и отсечений (branch and cut) к решению задачи о приёмке и доставке с временными ограничениями (PDPTW), представляющая собой важный класс задач маршрутизации в транспортной логистике. Подробно анализируются четыре математические формулировки задачи: 3-индексная, 2-индексная с ограничениями парности и порядка, компактная 2-индексная и асимметричная, каждая из которых имеет свои особенности с точки зрения вычислительной сложности и симметрии решений. Численные эксперименты, основанные на модифицированном наборе данных Li & Lim, демонстрируют, что компактная 2-индексная формулировка обладает наименьшей вычислительной сложностью и наилучшей скоростью нахождения точного решения, особенно при использовании эффективных ILP-солверов. Работа ориентирована на исследователей и специалистов, заинтересованных в применении точных методов для решения PDPTW.**

***Ключевые слова* — VRP, PDPTW, ILP, линейное программирование, маршрутизация, транспортные задачи.**

I. Введение

Транспорт и логистика играют ключевую роль в экономике: ежедневно перевозятся тонны грузов и тысячи людей, а в крупных городах активно развиваются сервисы доставки. Компании, так или иначе связанные с перевозками, стремятся снизить расходы на топливо, обслуживание техники и заработную плату персоналу, для чего они прибегают как к модернизации инфраструктуры, так и к использованию методов исследования операций *(Operations Research),* которые помогают определить наилучший способ использования имеющихся ресурсов.

Одними из наиболее популярных в научной литературе транспортных задач являются задачи маршрутизации транспортных средств *(Vehicle Routing Problem, VRP)*. В таких задачах задается набор пунктов, которые необходимо посетить, известна стоимость проезда между различными парами пунктов и депо и требуется построить оптимальные маршруты для разных машин, так, чтобы каждый пункт посетила только одна машина, и чтобы суммарные расходы по маршрутам были минимальными. Классическим примером VRP является задача коммивояжёра *(Travelling Salesman Problem)*, а её расширениями являются задачи с временными ограничениями *(VPR with time windows)* или с ограничением на вместимость транспортных средств *(Capacitated VRP)*. Задачи VRP относятся к классу NP-сложных задач [1], а поэтому при их решении необходимо учитывать, что вычислительная сложность резко возрастает при увеличении размерности задачи.

Задачи о приёмке и доставке *(Pickup and Delivery Problems, PDP)* представляют собой подкласс задач маршрутизации, в которых необходимо распределить по машинам набор запросов, каждый из которых заключается в доставке груза из пункта приёмки в пункт доставки Цель такая же: построить маршруты для машин, при этом каждый запрос должен быть выполнен только одной машиной, при этом суммарные расходы должны быть минимальными.

В данной работе рассматривается точный подход к решению задачи о приемке и доставке с временными ограничениями *(Pickup and Delivery Problem with Time Windows, PDPTW)*. Основной акцент сделан на анализе различных математических формулировок задачи и использовании метода ветвей и отсечений, для него также описываются техники сокращения области поиска. В отличие от эвристических методов, рассматриваемых в других работах, цель данной статьи — показать, как выбор формулировки и предобработка задачи может позволить ускорить поиск точного решения. Работа сопровождается численными экспериментами на стандартных бенчмарках.

Статья организована следующим образом: в 2-м разделе предлагается начальный обзор литературы. Во 3-м разделе дается описание и сравнение четырех формулировок задачи о приемке и доставке с временными ограничениями. В 4-м разделе статьи приводится численное исследование.

II. Текущее состояние проблемы

Задача, рассматриваемая в данной статье, имеет некоторые общие черты с транспортными задачами, ранее изученными в литературе и названными выше.

В [1], [2] рассматривается семейство задач VRP и предлагается основные точные и эвристические

Статья получена 5 мая 2025.
Царьков Никита Владимирович, ВМК, МГУ имени М.В. Ломоносова (e-mail: tsarkov90@gmail.com).

Голембиовский Дмитрий Юрьевич, д.т.н., ВМК, МГУ имени М.В. Ломоносова (email: golemb@cs.msu.ru).

алгоритмы, разработанные для решения VRP. Точные алгоритмы включают в себя методы ветвей и границ и его модификации, динамического программирования и алгоритмы разбиения множеств выполнимых решений.

Задача о приемке и доставке с временными ограничениями описывается в научных статьях [5], [6], [8] и находит множество практических применений, наиболее заметных в курьерской доставке товаров с маркетплейсов: клиенты заказывают товары из разных магазинов, а маркетплейс затем должен собрать все товары перед тем, как доставить их на дом клиенту. Разумеется, компания может объединить заказы разных клиентов в одной поездке.

В статьях [5], [9] представлен алгоритм адаптивного поиска ALNS с несколькими видами эвристик удаления и вставки запросов (пар «пункт приемки – пункт доставки») для задачи о приёмке и доставке.

В статье [11] предлагается метод ветвей и отсечений для решения задачи Dial-a-Ride, тесно связанной с PDPTW, с акцентом на временные окна и ограничения вместимости. В работах [12], [13] авторы представляют комбинированный метод ветвей, отсечений и ценообразования *(branch-and-cut-and-price)* для решения PDPTW и описывают различные ограничения для усиления линейной релаксации, эффективность применения которых была продемонстрирована на различных наборах тестовых данных.

В статье [21] авторы предлагают точный алгоритм для PDPTW, основанный на формулировке задачи в виде задачи о разбиении множества *(set partitioning)* и применении метода ветвей и отсечений для нахождения оптимального решения.

В статье [10] авторы представили ассиметричную формулировку, призванную бороться с симметрией решений – ситуацией для задачи с идентичными транспортными средствами, когда несколько решений имеют одну и ту же целевую функцию, т.е. множество маршрутов для транспортных средств совпадает, но различие в решениях заключаются в другом назначении маршрутов транспортным средствам (такая ситуация приводит к тому, что дерево ветвей и отсечений дублируется).

III. Задача о приёмке и доставке с временными ограничениями (Pickup and Delivery Problem with Time Windows)

Как было сказано ранее, в этой задаче требуется выполнять запросы на доставку – доставлять грузы из одного места в другое (со склада в пункт выдачи или непосредственно к клиенту), при этом необходимо учитывать время работы (или желаемое время доставки), а также ограничение на вместимость транспортных средств. Введем некоторые термины и обозначения:

Пусть K – парк идентичных транспортных средств и Q_{max} – их вместимость (в некоторых единицах, например, кубических метрах). $P = \{1, \dots, n\}$ и $D = \{n+1, \dots, 2n\}$ – наборы пунктов, откуда нужно забрать и куда нужно доставить груз соответственно ($|P| = |D| = n$). Определим $N = P \cup D$ как множество клиентских пунктов, а пункты с индексами 0 и $2n+1$ как депо, откуда машины начинают и где заканчивают свой маршрут.

Множество всех пунктов, используемых в задаче, определим как $V = N \cup \{0,\ 2n+1\}$. Каждый пункт $i \in V$ имеет своё время на обслуживание $s_i \geq 0$ (время на разгрузку или погрузку) и интервал обслуживания $TW_i = [a_i, b_i]$ (время работы, желаемое время приемки и доставки), т.е. машина должна прибыть в пункт i не позднее времени b_i, а если она прибудет в пункт i раньше времени a_i, то будет должна ждать до времени a_i.

Пусть $R = \{r_1, \dots, r_n\}$ – набор запросов (на доставку), где каждый запрос $r \in R$ представлен пунктом $p_r \in P$, откуда нужно забрать груз и пунктом $d_r \in D$, куда необходимо доставить этот груз. Обозначим за $r(i)$ – запрос, связанный с клиентским пунктом $i \in N$. Помимо этого, для каждого запроса $r \in R$ определен размер груза $q_{p_r} \geq 0$, который необходимо доставить: $q_{p_r} = -q_{d_r}$

Рассмотрим ориентированный граф $G = (V, A)$, где A – это множество дуг $A = V \times V$, за исключением тех, которые ведут к невыполнимым решениям: мы исключаем дугу (i, j) из пункта i в пункт j, если $b_j < a_i + s_i + t_{i,j}$, где $t_{i,j}$– время на преодоление расстояния $\rho_{i,j}$ между пунктами i и j. Кроме того, $c_{i,j}$ – расходы на поездку между пунктами i и j.

Задача PDPTW заключается в построении таких маршрутов для транспортных средств, чтобы все запросы на доставку были выполнены с минимальными расходами с учётом всех ограничений.

На рис. 1 представлен пример задачи о приемке и доставке: на нем зелёными точками изображены пункты приёмки, зелеными точками – пункты доставки, а стрелками между ними – запросы: пары пунктов приёмки

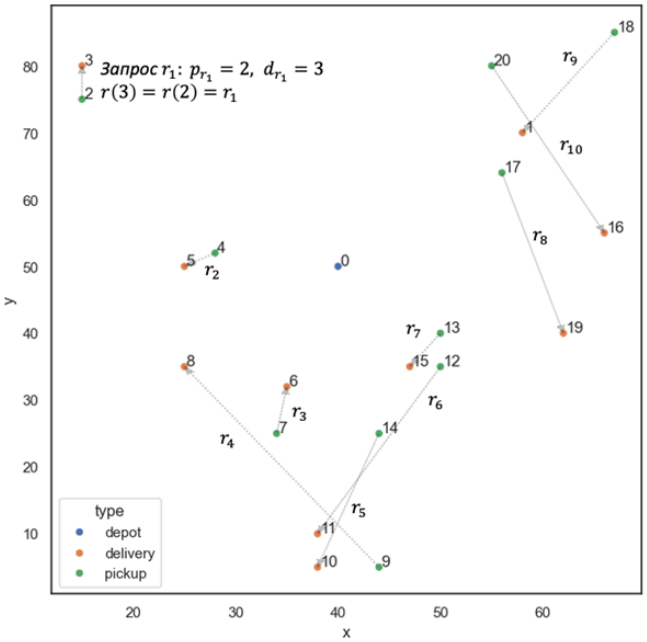

Рис. 1. Пример задачи о приёмке и доставке. На карте точками обозначены пункты (синим цветом – депо, зеленым – пункты приёмки, оранжевым – пункты доставки), пунктирными стрелками обозначены запросы: пары пунктов приемки и доставки

и доставки (например, пункты 2 и 3 образуют запрос r_1)

На рис. 2 представлено решение задачи, изображённой на рис. 1. На нём замкнутой последовательностью стрелок одинакового цвета

изображен маршрут прохождения пунктов для определённой машины (начиная последовательность «депо → … является одним из маршрутов в решении задачи).

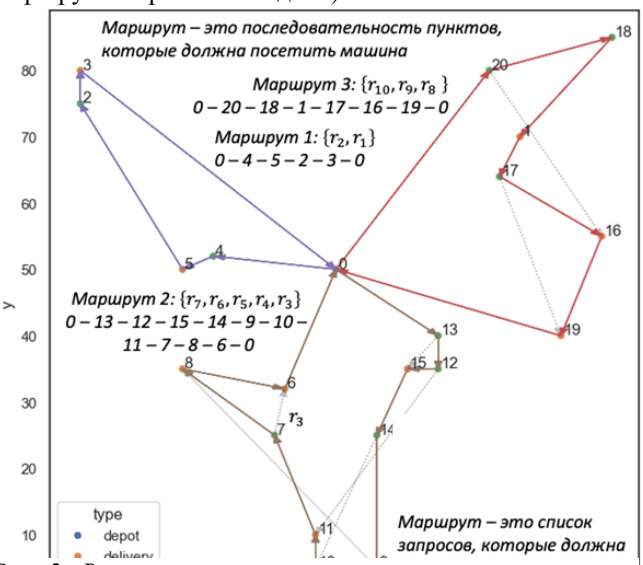

Рис. 2. Решение задачи о приемке и доставке, для пунктов, расположенных на рис. 3.1. В решении используется 3 машины, для которых стрелками обозначены маршруты прохождения пунктов

3.1. Различные варианты задачи

В рассматриваемой нами базовой постановке задачи PDPTW предполагается, что транспортные средства идентичны – имеют одинаковую вместимость, а каждый запрос соответствует доставке одного груза из одного пункта приемки в один пункт доставки, при этом каждый пункт привязан только к одному запросу. На практике встречаются различные варианты этой задачи; ниже приведены некоторые из них, и описаны способы перехода к базовой постановке задачи.

1) Несколько пунктов приёмки – один пункт доставки

Пусть для какого-нибудь запроса r стоит задача сбора нескольких грузов $q_r^1, q_r^2, \dots, q_r^m$ из соответствующих пунктов приемки $p_r^1, p_r^2, \dots, p_r^m$ и последующей доставки их в общий пункт d_r, для которого $q_{d_r} = -(q_r^1 + q_r^2 + \dots + q_r^m)$. Наглядно это изображено на рис. 3.

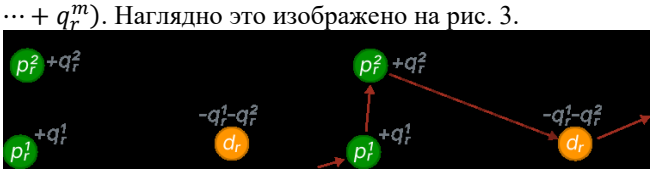

Рис. 3. Слева – ситуация, когда для одного запроса два пункта приёмки и один пункт доставки. Справа – предполагаемый маршрут выполнения этого запроса.

Такая ситуация сводится к базовой постановке задачи путем создания фиктивных *(dummy)* запросов $r^1, r^2, \dots, r^{m-1}$ с соответствующими копиями $d_r^1, d_r^2, \dots, d_r^{m-1}$ пункта доставки d_r, таким образом, чтобы каждый запрос обслуживал только один груз (рис. 4):

- Исходный запрос r теперь обслуживает только один груз q_r^1, который нужно доставить из p_r^1 в d_r
- Фиктивный запрос r^1 обслуживает груз q_r^2, пункт приемки p_r^2, пункт доставки d_r^1
- Запрос r^2 связан с грузом q_r^3 и пунктами p_r^3 и d_r^2 и т.д.
- Запрос r^{m-1} связан с грузом q_r^m и пунктами p_r^m и d_r^{m-1}.

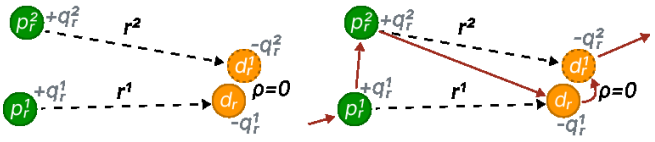

Рис. 4. Слева – разбиение запроса с двумя пунктами приемки путем создания фиктивного запроса с фиктивного пункта доставки. Справа – предполагаемый маршрут выполнения запросов. Маршрут совпадает с тем, что на рис. 3.

При этом расстояние, расходы и время в пути между фиктивными пунктами доставки, а также реальным и фиктивными пунктами доставки равно нулю:

$$\rho_{d_r^h,d_r^l} = c_{d_r^h,d_r^l} = t_{d_r^h,d_r^l} = 0, \qquad h,l = 1,\dots,m-1$$
$$\rho_{d_r^h,d_r} = c_{d_r^h,d_r} = t_{d_r^h,d_r} = 0, \qquad h = 1,\dots,m-1$$

Дополнительно вводим ограничения, чтобы машина сначала посетила реальный пункт доставки, а потом после него фиктивные:

$$\rho_{d_r,j} = c_{d_r,j} = t_{d_r,j} = \infty, \qquad j \in V\backslash\{d_r\}$$
$$\rho_{i,d_r^h} = c_{i,d_r^h} = t_{i,d_r^h} = \infty,$$
$$i \in V\backslash\{d_r\}, \quad h = 1,\dots,m-1$$

Наконец, для фиктивных пунктов доставки обнуляем время на обслуживание, а временной интервал обслуживания оставляем таким же, как и для реального пункта:

$$s_{d_r^h} = 0, \qquad h = 1,\dots,m-1$$
$$TW_{d_r^h} = TW_{d_r}, \qquad h = 1,\dots,m-1$$

2) Один пункт приёмки – несколько пунктов доставки

Ситуация диаметрально противоположная случаю с несколькими пунктами приёмки и одним пунктом доставки. В данной ситуации создаются фиктивные пункты приемки и все манипуляции проводятся аналогичным образом.

3) Транспортные средства имеют разную вместимость

Пусть в парке m машин с разной вместимостью $Q_1, Q_2, \dots, Q_m$ и пусть $Q_{max} = \max\limits_{1\le i\le m}\{Q_i\}$.

Тогда для перехода к базовой постановке определим m фиктивных запросов $r_1', \dots, r_m'$ с пунктами приёмки $p_1', \dots p_m'$ и доставки $d_1', \dots, d_m'$ соответственно, при этом время на обслуживание в каждом таком фиктивном пункте будет нулевым, а временной интервал обслуживания будет таким же как у депо:

$$s_{p_h} = s_{d_h} = 0, \qquad h = 1,\dots,m$$
$$TW_{p_h'} = TW_{d_h'} = TW_0 = TW_{2n+1}, \qquad h = 1,\dots,m$$

Каждый фиктивный запрос r_h', $h = 1,\dots,m$ будет обслуживать фиктивный груз $q_{p_h'} = -q_{d_h'} = Q_{max} - Q_h$. Суть такой манипуляции состоит в том, чтобы машина в начале своего маршрута взяла фиктивный груз, тем самым ограничив свою вместимость.

Опишем дополнительные ограничения на условия задачи. В каждый фиктивный пункт приемки можно приехать только из депо:

$$\rho_{0,p_h'} = t_{0,p_h'} = c_{0,p_h'} = 0, \qquad h = 1,\dots,m$$

$$\rho_{i,p'_h} = t_{i,p'_h} = c_{i,p'_h} = \infty, \qquad h = 1, \dots, m, \ \ i \in N$$

а из каждого фиктивного пункта доставки можно отправиться только в депо:

$$\rho_{d'_h,2n+1} = t_{d'_h,2n+1} = c_{d'_h,2n+1} = 0, \qquad h = 1, \dots, m$$

$$\rho_{d'_h,j} = t_{d'_h,j} = c_{d'_h,j} = 0, \qquad h = 1, \dots, m, \ \ j \in N$$

При этом между обычным пунктом приемки или доставки и фиктивным расстояние, расходы или время в пути должны быть такими же, как до депо:

$$\rho_{i,d'_h} = \rho_{i,2n+1}; \ \ t_{i,d'_h} = t_{i,2n+1}; \ \ c_{i,d'_h} = c_{i,2n+1}, \quad h = 1, \dots, m, \ \ i \in N$$

$$\rho_{p'_h,j} = \rho_{0,j}; \ \ t_{p'_h,j} = t_{0,j}; \ \ c_{p'_h,j} = c_{0,j}, \quad h = 1, \dots, m, \ \ j \in N$$

Наконец, расстояние, расходы и время в пути и между различными фиктивными пунктами приемки и доставки должны быть нулевыми:

$$\rho_{d'_h,d'_l} = c_{d'_h,d'_l} = t_{d'_h,d'_l} = 0, \qquad h, l = 1, \dots, m$$

$$\rho_{p'_h,d'_l} = c_{p'_h,d'_l} = t_{p'_h,d'_l} = 0, \qquad h, l = 1, \dots, m$$

$$\rho_{p'_h,p'_l} = c_{p'_h,d'_l} = t_{p'_h,d'_l} = 0, \qquad h, l = 1, \dots, m$$

3.2. 3-индексная формулировка

Классической для задачи о приемке и доставке с временными ограничениями является 3-индексная формулировка [11], [12]. Вводятся четыре группы переменных:

- $x^k_{i,j} = 1$, если машина k следует из пункта i в пункт j, иначе *0 (три индекса: два пункта и машина)*
- $y_{k,r} = 1$, если запрос r выполняется k-й машиной, иначе 0
- S_i показывает время прибытия в пункт $i \in V$ ($S_0 = 0$)
- Q_i показывает загрузку машины после выезда из пункта $i \in V$, ($Q_0 = Q_{2n+1} = 0$)

Задача о приемке и доставке с временными ограничениями формулируется следующим образом:

$$\text{minimize} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{i,j} x^k_{i,j} \qquad (1)$$

с учетом ограничений:

$$\sum_{j \in V \setminus \{i\}} x^k_{i,j} = y_{k,r(i)}, \qquad k \in K, \ \ i \in N \qquad (2)$$

$$\sum_{j \in V \setminus \{i\}} x^k_{j,i} = y_{k,r(i)}, \qquad k \in K, \ \ i \in N \qquad (3)$$

$$\sum_{j \in N} x^k_{0,j} \le 1, \qquad k \in K \qquad (4)$$

$$\sum_{k \in K} y_{k,r} = 1, \qquad r \in R \qquad (5)$$

$$S_j \ge S_i + s_i + t_{i,j} - M\left(1 - \sum_{k \in K} x^k_{i,j}\right), \qquad i, j \in V \qquad (6)$$

$$a_i \le S_i \le b_i, \qquad i \in V \qquad (7)$$

$$S_{d_r} \ge S_{p_r} + s_{p_r} + t_{p_r,d_r}, \qquad r \in R \qquad (8)$$

$$Q_j \ge Q_i + q_j - Q_{max}\left(1 - \sum_{k \in K} x^k_{i,j}\right), \qquad i, j \in V \qquad (9)$$

$$x^k_{i,j}, y_{k,r} \in \{0,1\}, \qquad i, j \in V, \ \ k \in K, \ \ r \in R \qquad (10)$$

$$0 \le S_i \le M, \qquad i \in V \qquad (11)$$

$$0 \le Q_i \le Q_{max}, \qquad i \in V \qquad (12)$$

Целевая функция (1) минимизирует общие транспортные расходы. Ограничения (2) и (3) гарантируют, что каждый клиентский пункт посещается только одной машиной. Ограничения (4) требуют, чтобы в решении использовалось не более $|K|$ машин. Ограничения (5) необходимы для обозначения того факта, что каждый запрос на доставку обслуживается только одной машиной, а ограничение (8) – что пункт доставки посещается после пункта приемки. Ограничения (6) и (7) вводятся для обеспечения того, чтобы каждая машина посещала клиентский пункт, согласно его времени работы (временному интервалу). Ограничение (9) является гарантией того, что для любой машины не будет превышена её вместимость в процессе прохождения маршрута. Ограничения (10), (11) и (12) описывают характер переменных. M – достаточно большое число, которое может быть задано максимальным временем возврата в депо b_{2n+1}.

3-индексная формулировка имеет существенное достоинство: она явным образом описывает взаимодействие ключевых объектов в задаче в формулировке *«k-я машина следует из пункта i в пункт j»*. Однако число переменных в задаче достаточно велико: $O(|R|^2|K|)$, т.е. оно растет с кубической скоростью, что делает применение данной формулировки на практике затруднительным при больших размерностях (раздел 3.6, рис. 7)

3.3. 2-индексная формулировка

Для решения проблемы числа переменных в задаче в статьях [13], [14] авторы предложили формулировку, в которой у бинарной переменной x_{ij}, определяющей факт прохождения дуги (i, j) нет индекса машины. Возникает проблема: по условию задачи требуется, чтобы пункты приёмки и доставки для любого запроса обслуживались одной и той же машиной. Пример невыполнения такого требования изображен на рис. 5: красная стрелка указывает на запрос 6 → 2, выполнение которого противоречит логике задачи: разные машины посещают

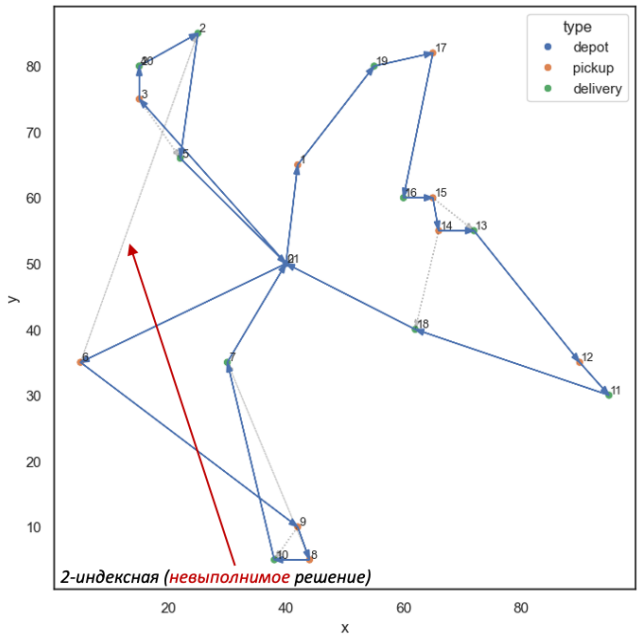

Рис. 2. Пример невыполнения условий задачи: один запрос выполняют разные машины: пункт 6 посещает одна машина, а пункт 2 – другая.

пункты, связанные с этим запросом.

Для корректной обработки пар «приёмка-доставка» вводятся ограничения парности и порядка *(pairing and precedence constraints)*, гарантирующие, что и приёмка, и доставка каждого запроса выполняются одной и той же машиной, и при этом доставка не предшествует приёмке. Определяется множество $\boldsymbol{S}$ всех подмножеств узлов $S \subset V$ таких, что $0 \notin S,\ p_r \in S,\ d_r \notin S$ и $2n+1 \in S$ хотя бы для одного запроса $r \in R$ (т.е. для определённого запроса в S могут лежать любые пункты приёмки, депо и пункты доставки, не связанные с этим запросом).

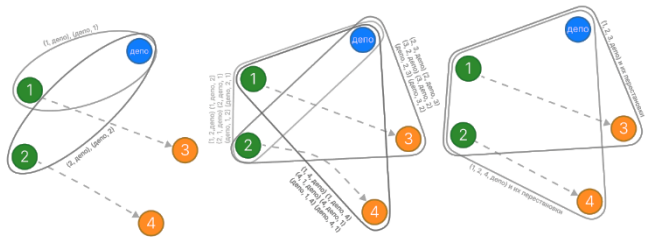

Рис. 6. Множество $\boldsymbol{S}$ для задачи PDPTW с 4-мя клиентскими пунктами (1,2 – пункты приемки, 3,4 – пункты доставки) состоит из подмножеств {1, депо}, {2, депо}, {1, 2, депо}, {1, 4, депо}, {2, 3, депо}, {1, 2, 3, депо} и {1, 2, 4, депо}, а также их различных перестановок

Примеры таких подмножеств S для задачи с 4-мя клиентскими пунктами изображены на рис. 7.

Также вводятся три группы переменных:

- $x_{i,j} = 1$, если какая-нибудь машина следует из пункта i в пункт j, иначе 0
- S_i показывает время прибытия в пункт $i \in V$ $(S_0 = 0)$
- Q_i показывает загрузку транспортного средства после выезда из пункта $i \in V$, $(Q_0 = Q_{2n+1} = 0)$

Тогда задача о приемке и доставке с временными ограничениями нами формулируется следующим образом:

$$\text{minimize} \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{i,j} \tag{13}$$

с учетом ограничений:

$$\sum_{i \in V} x_{i,j} = 1, \qquad j \in N \tag{14}$$

$$\sum_{j \in V} x_{i,j} = 1, \qquad i \in N \tag{15}$$

$$\sum_{j \in N} x_{0,j}^k \le |K| \tag{16}$$

$$\sum_{i \in \boldsymbol{S}} \sum_{j \in \boldsymbol{S}} \boldsymbol{x_{i,j}} \le |\boldsymbol{S}| - \boldsymbol{2}, \qquad \boldsymbol{S} \in \boldsymbol{\mathcal{S}} \tag{17}$$

$$S_j \ge S_i + s_i + t_{i,j} - M(1 - x_{i,j}), \qquad i,j \in V \tag{18}$$

$$a_i \le S_i \le b_i, \qquad i \in V \tag{19}$$

$$S_{d_r} \ge S_{p_r} + s_{p_r} + t_{p_r,d_r}, \qquad r \in R \tag{20}$$

$$Q_j \ge Q_i + q_j - Q_{max}(1 - x_{i,j}), \qquad i,j \in V \tag{21}$$

$$x_{i,j} \in \{0,1\}, \qquad i,j \in V \tag{22}$$

$$0 \le S_i \le M, \qquad i \in V \tag{23}$$

$$0 \le Q_i \le Q_{max}, \qquad i \in V \tag{24}$$

Целевая функция (13) и ограничения (14) – (16), (18) – (24) имеют тот же смысл, как и в 3-индексной формулировке.

Ограничения (17) – это *ограничения парности и порядка (pairing and precedence constraints)*. Отметим, что множество таких ограничений при увеличении числа клиентских пунктов само по себе растет с факториальной скоростью, поэтому его стоит использовать совместно с LP-релаксациями (в солверах для этого можно задавать т.н. *lazy constraints*):

1) Для начала необходимо решить задачу без ограничений (17), после чего построить маршруты для машин
2) Для каждой машины проверить корректность маршрута: после каждого пункта приёмки машина посещает пункт доставки, связанный запросом
3) Если нарушений нет – то это оптимальное решение.
4) Если существует маршрут, в котором есть пункт приемки, но нет пункт доставки, связанного запросом, то обозначим этот маршрут за S (и на самом деле $S \in \boldsymbol{S}$) и построим ограничение вида (17).
5) Решаем задачу повторно с новым ограничением.

Для данной формулировки число переменных уже кратно меньше: $O(|R|^2)$, т.е. оно растет уже с квадратичной скоростью (раздел 3.6, рис. 7).

3.4. 2-индексная компактная формулировка

Во избежание последовательного перебора ограничений вида (17), в статье [15] авторы ввели т.н. *компактную* 2-индексную формулировку, в которой вводится группа вспомогательных переменных v_i, определяющих индекс маршрута для каждого пункта и некоторые дополнительные ограничения, которые будут описаны ниже. Данная методика позволяет неявно зафиксировать принадлежность пар «приёмка-доставка» одному маршруту и упорядочить посещение пунктов.

Рассмотрим маршрут, который начинается пунктом $j \in N$, проходит через пункт i, и выполняет некоторый запрос $r \in R$:

$$0 \to j \to \cdots \to i \to p_r \to \cdots \to d_r \to \cdots \to 2n+1$$

Индекс маршрута определяется первым пунктом j, через который он проходит, а значит для этого маршрута следует, что:

$$v_j = v_i = v_{p_r} = v_{d_r} = j$$

По такой логике компактная 2-индексная формулировка для нашего случая задачи формулируется следующим образом:

$$\text{minimize} \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{i,j} \tag{25}$$

с учетом ограничений:

$$\sum_{i \in V} x_{i,j} = 1, \qquad j \in N \tag{26}$$

$$\sum_{j \in V} x_{i,j} = 1, \qquad i \in N \tag{27}$$

$$\sum_{j \in N} x_{0,j}^k \le |K| \tag{28}$$

$$S_j \ge S_i + s_i + t_{i,j} - M(1 - x_{i,j}), \qquad i,j \in V \tag{29}$$

$$a_i \le S_i \le b_i, \qquad i \in V \tag{30}$$

$$S_{d_r} \ge S_{p_r} + s_{p_r} + t_{p_r,d_r}, \qquad r \in R \tag{31}$$

$$Q_j \ge Q_i + q_j - Q_{max}(1 - x_{i,j}), \qquad i,j \in V \tag{32}$$

$$\boldsymbol{v_{d_r} = v_{p_r}}, \qquad \boldsymbol{r \in R} \tag{33}$$

$$\boldsymbol{v_j \ge j x_{0,j}}, \qquad \boldsymbol{j \in N} \tag{34}$$

$$\boldsymbol{v_j \le j x_{0,j} - |N|(x_{0,j} - 1)}, \qquad \boldsymbol{j \in N} \tag{35}$$

$$\boldsymbol{v_j \geq v_i + |N|(x_{i,j} - 1)}, \qquad \boldsymbol{i,j \in N} \qquad (\mathbf{36})$$

$$\boldsymbol{v_j \leq v_i + |N|(1 - x_{i,j})}, \qquad \boldsymbol{i,j \in N} \qquad (\mathbf{37})$$

$$x_{i,j} \in \{0,1\}, \qquad i,j \in V \qquad (38)$$

$$0 \leq S_i \leq M, \qquad i \in V \qquad (39)$$

$$0 \leq Q_i \leq Q_{max}, \qquad i \in V \qquad (40)$$

$$1 \leq v_i \leq 2n, \qquad i \in N \qquad (41)$$

Ограничения (33) гарантируют, что пункт приемки и пункт доставки запроса принадлежат одному и тому же маршруту. Ограничения (34) и (35) требуют, чтобы индекс маршрута принимался за индекс первого посещенного пункта. Ограничения (36) и (37) нужны для гарантии того, чтобы индекс первого посещенного пункта был передан следующим пунктам маршрута. Значение $|N|$ представляет собой количество клиентских пунктов и равно $2n$.

Число переменных в такой формулировке по-прежнему остаётся $O(|R|^2)$. Эта формулировка удобна тем, что для неё не нужно явным образом вводить ограничения, которые растут с факториальной скоростью при увеличении числа клиентских пунктов.

3.5. Ассиметричная формулировка

При решении задачи PDPTW с транспортными средствами с одинаковой вместимостью неизбежно возникает *симметрия решений* – ситуация, когда возникает множество решений с одинаковым значением целевой функции, при этом сами решения отличаются перестановками маршрутов между машинами. Такая ситуация является большой проблемой, ведь усложняется оптимизационный процесс, увеличивается объем вычислений, дублируется дерево решений для метода ветвей и отсечений (который будет описан далее)

В статье [10] представлена ассиметричная формулировка *(asymmetric representatives formulation)*, призванная устранить такую проблему. Авторы предложили упорядочить запросы по их индексу и ввести множество кластеров запросов $K = \{k_1, k_2, \dots, k_n\}$ таким образом, чтобы каждый запрос был определен в каждый кластер с индексом, не превышающим индекс запроса:

$$\begin{aligned} k_1 &= \{r_1, \ r_2, \ r_3, \ \dots, \ r_n\} \\ k_2 &= \{r_2, \ r_3, \ \dots, \ r_n\} \\ k_3 &= \{r_3, \ \dots, \ r_n\} \\ &\dots \\ k_n &= \{r_n\} \end{aligned}$$

В остальном, формулировка очень похожа на 3-индексную, за исключением того, что все рассуждения ведутся относительно кластеров, а не машин. Вводятся также четыре группы переменных:

- $x_{i,j}^k = 1$, если $i = p_r$ или $j = d_r$ для запроса $r \in k$, иначе *0*
- $y_{k,r} = 1$, если $r \in \tilde{k}$, иначе *0*
- S_i показывает время прибытия в пункт $i \in V$ $(S_0 = 0)$
- Q_i показывает загрузку транспортного средства после выезда из пункта $i \in V$, $(Q_0 = Q_{2n+1} = 0)$

Исходя из логики распределения запросов кластерам, около половины переменных сразу же обнуляются:

- $y_{k,r} = 0$, если $r < k$, для любых $k \in K, \ r \in R$
- $x_{i,j}^k = 0$, если $r(i) < k$ или $r(j) < k$, для любых $k \in K, \ i,j \in N$

Тогда задача формулируется следующим образом:

$$\text{minimize} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{i,j}^k \qquad (43)$$

с учетом ограничений:

$$\sum_{j \in V \setminus \{i\}} x_{i,j}^k = y_{k,r(i)}, \qquad k \in K, \quad i \in N \qquad (44)$$

$$\sum_{j \in V \setminus \{i\}} x_{j,i}^k = y_{k,r(i)}, \qquad k \in K, \quad i \in N \qquad (45)$$

$$\sum_{j \in N} x_{0,j}^k \leq 1, \qquad k \in K \qquad (46)$$

$$\sum_{\boldsymbol{k \in K}} \boldsymbol{y_{k,k}} \leq \boldsymbol{|K|} \qquad (\mathbf{47})$$

$$\sum_{k \in K} y_{k,r} = 1, \qquad r \in R \qquad (48)$$

$$\sum_{\boldsymbol{r \in R:\, r > k}} \boldsymbol{y_{k,r} \leq y_{k,k}(|R| - k)}, \qquad \boldsymbol{k \in K} \qquad (\mathbf{49})$$

$$S_j \geq S_i + s_i + t_{i,j} - M\left(1 - \sum_{k \in K} x_{i,j}^k\right), \qquad i,j \in V \qquad (50)$$

$$a_i \leq S_i \leq b_i, \qquad i \in V \qquad (51)$$

$$S_{d_r} \geq S_{p_r} + s_{p_r} + t_{p_r,d_r}, \qquad r \in R \qquad (52)$$

$$Q_j \geq Q_i + q_j - Q_{max}\left(1 - \sum_{k \in K} x_{i,j}^k\right), \qquad i,j \in V \qquad (53)$$

$$x_{i,j}^k, y_{k,r} \in \{0,1\}, \qquad i,j \in V, \quad k \in K, \quad r \in R \qquad (54)$$

$$0 \leq S_i \leq M, \qquad i \in V \qquad (55)$$

$$0 \leq Q_i \leq Q_{max}, \qquad i \in V \qquad (56)$$

Логика ограничений такая же, как и в 3-индексной формулировке. Ограничения (47) требуют, что итоговое число кластеров в решении было не больше числа машин. Ограничения (49) гарантируют, что в кластерах с индексом k будут только запросы с индексом не ниже k.

Число переменных в задаче достаточно велико: $O(|R|^3)$, (раздел 3.6, рис. 7), однако время решения методом ветвей и отсечений меньше ввиду отсутствия симметрии решений.

3.6. Сравнение вычислительной сложности формулировок

На рис. 7 показано, как увеличивается количество переменных для трех различных формулировок с ростом размерности задачи. Как видно, при 2-индексных формулировках вычислительная сложность задачи растет заметно меньше при увеличении размерности входных данных, что делает её более предпочтительной для использования в точных методах.

Для поиска точного решения использовался метод ветвей и отсечений *(Branch-and-Cut)*, сочетающий в себе классическую схему ветвления *(branching)*, характерную для метода ветвей и границ, с последовательным построением линейных ограничений, которые отсекают *(cuts)* решения LP-релаксаций, не удовлетворяющие целочисленным условиям (в научной литературе алгоритм построения отсечений также известен, как алгоритм Гомори).

Учитывая высокую вычислительную сложность

задачи PDPTW и значительный рост числа переменных при увеличении размера входных данных, критически важным становится использование методик, призванных уменьшить область поиска выполнимых решений, ускорить сходимость метода ветвей и отсечений и повысить шанс нахождения оптимального решения в разумные время. Исследователи часто прибегают к удалению дуг, приводящих к невыполнимым решениям;

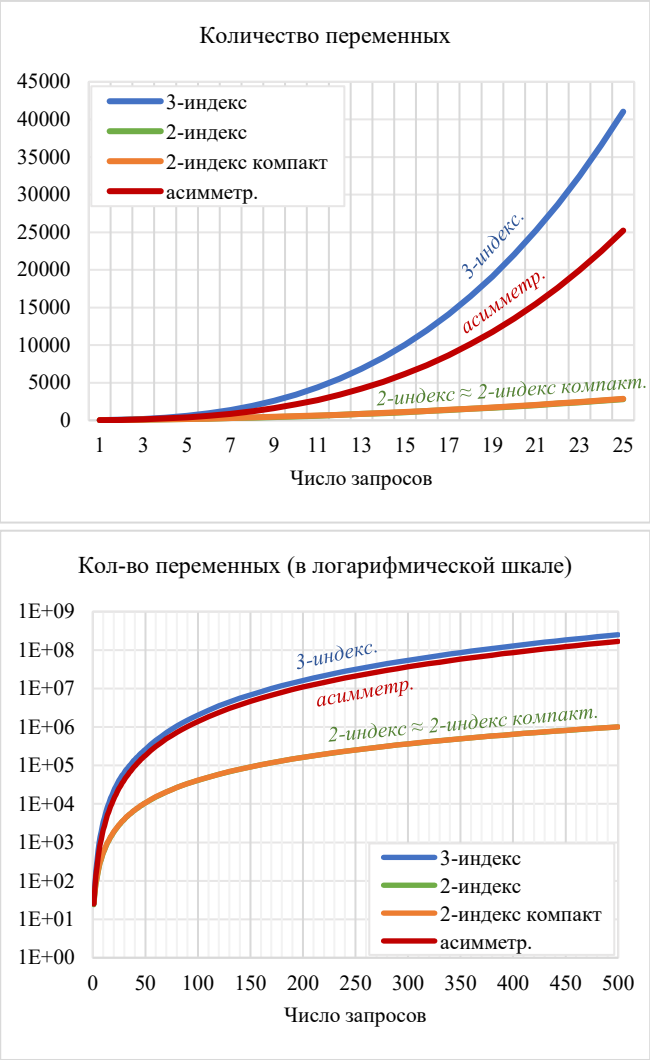

Рис. 7. Графики роста числа переменных в зависимости от размерности задачи

исключению подциклов *(eliminating subtours)* [10], [12]; добавлению более строгих ограничений на вместимость и fork ограничений [5], [12].

IV. Численное исследование

Для эмпирической оценки эффективности различных формулировок задачи PDPTW и метода ветвей и отсечений было проведено численное исследование с использованием модифицированного датасета Li & Lim Benchmark [17], [18]. Изначально в датасете нет примеров задач размерностью меньше, чем 100 клиентских пунктов и 25 машин, поэтому для генерации меньших размерностей был рассмотрен каждый пример задачи размерностью [100 пунктов, 25 машин] и из неё случайным образом были отобраны запросы, чтобы сгенерировать датасеты на 24 и 50 клиентских пунктов, при этом число машин было уменьшено до 8 и 15 соответственно. Все задачи в датасете содержат набор клиентских пунктов, депо, ограничения на вместимость транспортных средств, временные интервалы работы клиентских пунктов, а также обозначения, какой пункт является пунктом приемки, и какой – доставки.

Датасет поделен на три части в зависимости от того, как расположены клиентские пункты на карте: LC, LR и LRC (рис. 8–10). Клиенты в задачах LC распределены группами по карте. В задачах LR клиенты распределены случайным образом равномерно по всей карте. В задачах LRC клиенты наполовину сгруппированы и наполовину случайно расположены по карте.

В качестве LP-солвера использовался Cardinal Optimizer (COPT) [20], обладающий встроенными средствами предобработки задачи и эвристиками, поддержкой ленивых ограничений и адаптивного ветвления. Для каждого варианта задачи было задано ограничение по времени: 1 час для задач размерностью до 100 клиентских пунктов и 2 часа размерностью 100. Эксперименты проводились на Apple M1 Pro (ARM, 8 ядер 3.22 ГГц, 16 Гб ОЗУ).

Начальное выполнимое решение генерировалось с помощью алгоритма адаптивного поиска в больших окрестностях (ALNS, 100 и 1000 итераций), что позволило получить хорошую верхнюю границу для метода ветвей и отсечений. Несмотря на то, что COPT может сам генерировать начальные решения, но в рамках задачи PDPTW в большинстве случаев эти решения были хуже, чем решения, полученные с помощью ALNS.

Анализ результатов для различных формулировок (табл. 1, 2) показал, что:

- **3-индексная формулировка** демонстрировала наихудшие результаты: она была вычислительно слишком затратной из-за большого числа переменных. За 1 час работы для каждого примера задачи солвер смог построить решения почти всех примеров размерности 24 и только для 14 из 60 для размерности 50.
- **Обычная 2-индексная формулировка** показала неоднозначные результаты. Ввиду того, что эта формулировка предполагает в себе LP-релаксации, то часто ILP-решения получались невыполнимыми из-за ситуации, когда для определённого запроса пункт приемки посещает одно транспортное средство, а пункт доставки – другое. Такие решения последовательно исключались из области выполнимых решений путём введения дополнительных ограничений вида (15), пока солвер не выдаст выполнимое ILP-решение. Тем не менее сохраняется тенденция, что примеры задач со сгруппированными клиентскими пунктами решаются быстрее. Однако для данного набора задач получилось, что 2-индексная формулировка решила задач не больше, чем 3-индексная, но при этом делала это быстрее.
- **Компактная 2-индексная формулировка и асимметричная формулировки** находят оптимальные решения заметно быстрее обычной 2-индексной формулировки. Ассиметричная формулировка в среднем была немного медленнее из-

за большей вычислительной сложности задачи, но при этом она решила больше примеров за отведенное время.

целочисленных ограничений (ILP), а *lower bound* (нижняя граница) – значение целевой функции для задачи без целочисленных ограничений (LP-релаксации).

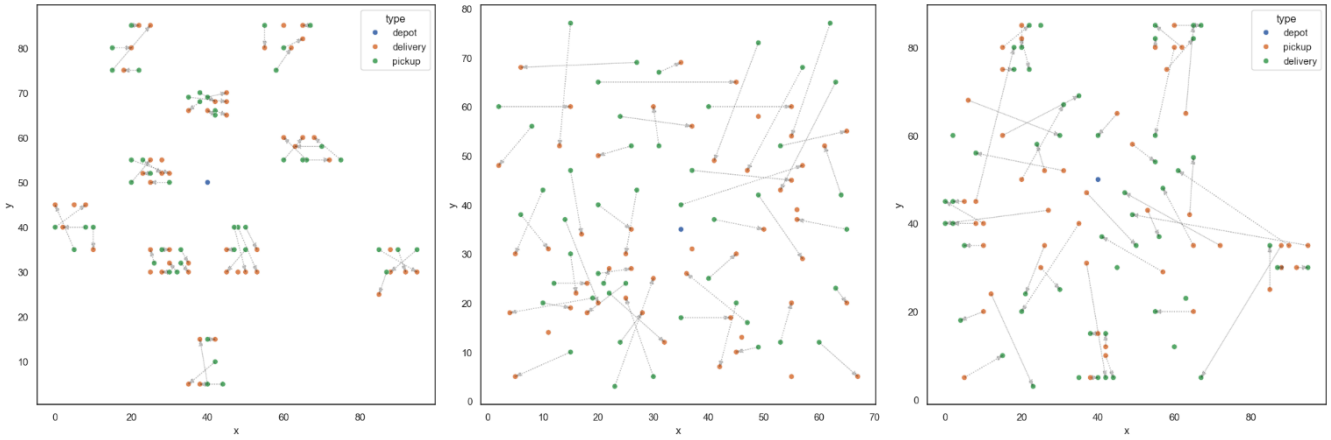

Рис. 8-10. Примеры задач из датасета. Слева направо: LC (клиенты расположены группами по карте), LR (клиенты равномерно распределены по карте) и LRC (сочетание группировки и случайного распределения)

Таблица 1. Количество примеров из датасета, успешно решенных методом ветвей и отсечений с помощью солвера COPT (прочерк – не проверялось)

Количество примеров из датасета, успешно решенных солвером									
Число пунктов и тип задачи		Формулировка и кол-во итераций ALNS для генерации начального решения							
		3-индексная		2-индексная		2-индекс. компакт.		Асимметричная	
		100	1 000	100	1 000	100	1 000	100	1 000
24	LC	20 из 20	20 из 20	20 из 20	20 из 20	20 из 20	20 из 20	20 из 20	20 из 20
	LR	19 из 20	20 из 20	16 из 20	17 из 20	20 из 20	20 из 20	20 из 20	20 из 20
	LRC	18 из 20	17 из 20	19 из 20	20 из 20	19 из 20	19 из 20	20 из 20	20 из 20
50	LC	7 из 20	8 из 20	7 из 20	9 из 20	15 из 20	15 из 20	16 из 20	16 из 20
	LR	4 из 20	4 из 20	0 из 20	0 из 20	8 из 20	8 из 20	9 из 20	9 из 20
	LRC	2 из 20	2 из 20	1 из 20	1 из 20	9 из 20	9 из 20	10 из 20	10 из 20
100	LC	—		12 из17	14 из 17	14 из 17	14 из 17	—	8 из 17
	LR			2 из 23	2 из 23	2 из 23	2 из 23		2 из 23
	LRC			0 из 16	0 из 16	0 из 16	0 из 16		1 из 16

Кроме того, было экспериментально показано, что метод ветвей и отсечений заметно быстрее решает примеры задач типа LC, где клиентские пункты распределены по карте группами. Причиной этому являются более удачный выбор начального выполнимого решения, в котором уже учтены оптимальные маршруты внутри групп пунктов, поскольку начальное выполнимое решение строится с помощью жадных вставок (табл. 1, 2)

Рассмотрим теперь примеры, когда солвер за отведенное время не нашел оптимального решения. Будем их рассматривать с двух аспектов: 1) как уменьшается целевая функция во время работы солвера (табл. 3) и 2) каков gap (табл. 4).

Напомним, что *gap = (best upper bound – lower bound)/ best upper bound*, где *upper bound* (верхняя граница) –

Таблица 2. Статистика по времени поиска точного решения для успешно решенных примеров (прочерк – недостаточно данных для оценки)

Среднее время работы солвера в секундах для успешно решенных примеров									
Число пунктов и тип задачи		Формулировка и кол-во итераций ALNS для генерации начального решения							
		3-индексная		2-индексная		2-индекс. компакт.		Асимметричная	
		100	1 000	100	1 000	100	1 000	100	1 000
24	LC	108 с.	92 с.	4 с.	4 с.	6 с.	4 с.	6 с.	5 с.
	LR	336 с.	442 с.	291с.	356 с.	71 с.	55 с.	47 с.	41 с.
	LRC	360 с.	345 с.	261 с.	218 с.	41 с.	31 с.	70 с.	52 с.
50	LC	157 с.	844 с.	395 с.	647 с.	114 с.	170 с.	242 с.	258 с.
	LR	—	—	—	—	65 с.	205 с.	122 с.	133 с.
	LRC	—	—	—	—	541 с.	605 с.	704 с.	859 с.
100	LC	—		437 с.	1016 с.	497 с.	197 с.	—	1149 с.
	LR			—	—	—	—		—
	LRC			—	—	—	—		—

Таблица 3. Статистика по улучшению целевой функции во время работы солвера относительно начального выполнимого решения, сгенерированного через ALNS путем прогона 100 и 1000 итераций (прочерк – недостаточно данных для оценки, нет цифры – все решения наилучшие)

Среднее улучшение целевой функции (примеры с Timeout при поиске) относительно начального решения									
Число пунктов и тип задачи		Формулировка и кол-во итераций ALNS для генерации начального решения							
		3-индексная		2-индексная		2-индекс. компакт.		Асимметричная	
		100	1 000	100	1 000	100	1 000	100	1 000
24	LC								
	LR	0,3%		4,1%	1,7%				
	LRC	20,9%	1,0%	7,5%	0,6%	17,9%	0,9%		
50	LC	20,5%	3,4%	13,8%	2,8%	25,6%	0,8%	14,0%	11,3%
	LR	8,3%	3,3%	6,7%	0,8%	14,8%	4,3%	13,9%	2,3%
	LRC	11,7%	1,9%	15,0%	3,6%	18,5%	6,8%	17,7%	3,7%
100	LC	—		8,5%	7,3%	0,0%	0,0%	—	1,5%
	LR			1,4%	0,5%	0,0%	0,0%		1,5%
	LRC			1,2%	1,4%	0,0%	0,4%		1,1%

В случае 2-индексных формулировок, нулевое

улучшение функции для размерности 100 при поиске методом ветвей и отсечений показывает, насколько велика вычислительная сложность задачи: солвер буквально не успел за 2 часа найти решение лучше, чем начальное выполнимое решение, сгенерированное эвристиками адаптивного поиска ALNS.

Таблица 4. Статистика по *gap* для примеров, по которым не удалось найти точное решение методом ветвей и отсечений за требуемое время (прочерк – недостаточно данных для оценки, нет цифры – все решения наилучшие)

Средний Gap (для примеров с Timeout при поиске)									
Число пунктов и тип задачи		Формулировка и кол-во итераций ALNS для генерации начального решения							
		3-индексная		2-индексная		2-индекс. компакт.		Асимметричная	
		100	1 000	100	1 000	100	1 000	100	1 000
24	LC								
	LR	4,1%		10,2%	11,5%				
	LRC	18,9%	15,6%	4,2%	0,9%	17,7%	16,4%		
50	LC	24,6%	22,9%	24,8%	11,4%	19,3%	19,4%	33,5%	20,1%
	LR	30,4%	23,3%	26,5%	19,7%	24,5%	20,3%	28,5%	24,3%
	LRC	30,4%	23,1%	19,9%	13,1%	15,8%	15,6%	21,8%	18,9%
100	LC	—		50,7%	25,6%	55,9%	30,1%	—	30,7%
	LR			45,2%	27,4%	46,3%	27,8%		34,9%
	LRC			51,8%	28,6%	53,4%	28,9%		42,7%

Из таблиц 4, 5 видно, что увеличение числа итераций алгоритма ALNS с 100 до 1000 имеет положительный эффект: как gap, так и ошибка к наилучшему решению заметно ниже. Так же видим, что 2-индексные формулировки из-за меньшего числа переменных показывают лучшую производительность. В таблице 5 показано, что солвер в случае 2-индексной компактной формулировки находил решения с наименьшей ошибкой к оптимальному решению. Ассиметричная формулировка не показала значимых улучшений по сравнению с 2-индексной компактной формулировкой.

Таблица 5. Статистика по ошибке к наилучшему решению (нет цифры – нет ошибки, прочерк – недостаточно данных для оценки)

Средняя ошибка к наилучшему решению									
Число пунктов и тип задачи		Формулировка и кол-во итераций ALNS для генерации начального решения							
		3-индексная		2-индексная		2-индекс. компакт.		Асимметричная	
		100	1 000	100	1 000	100	1 000	100	1 000
24	LC								
	LR			0,6%	0,5%				
	LRC	0,2%	0,1%	0,1%					
50	LC	16,4%	6,4%	19,9%	4,1%	6,3%	3,9%	6,9%	0,4%
	LR	13,9%	4,0%	18,4%	8,4%	6,0%	2,3%	6,1%	2,3%
	LRC	16,6%	6,7%	15,1%	4,9%	2,8%	2,3%	4,7%	2,0%
100	LC	—		28,1%	2,6%	15,8%	3,5%	—	5,1%
	LR			50,6%	14,1%	52,6%	14,5%		12,7%
	LRC			72,0%	12,3%	75,0%	13,4%		11,3%

Выводом из данного анализа является рекомендация к использованию 2-индексных формулировок или асимметричной формулировки. Дополнительно необходимо применять методики по сокращению области поиска выполнимых решений, и строить начальные решения с помощью продвинутых эвристик, таких как ALNS, управляемый локальный поиск, генетические алгоритмы.

V. Заключение

В данной статье был рассмотрен точный подход к решению задачи о приемке и доставке с временными ограничениями. Были представлены 3-индексная, 2-индексная, компактная 2-индексная и асимметричная формулировки, проведено сравнение вычислительной сложности каждой из формулировок.

Результаты проведённого анализа и экспериментов позволяют сделать несколько важных выводов относительно применения точных методов к задаче PDPTW. Прежде всего, выбор формулировки задачи оказывает критическое влияние на скорость нахождения точного решения: переход от 3-индексной формулировки к компактной 2-индексной позволил существенно снизить количество переменных, а следовательно, и дерево поиска для метода ветвей и отсечений, тем самым ускорив его сходимость

Помимо этого, использование эвристик (например, ALNS) для генерации начального решения в сочетании с предобработкой графа и динамическим добавлением ограничений позволило существенно ускорить сходимость метода ветвей и отсечений, что было особенно заметно в задачах типа LC, где клиенты распределены группами по карте.

Метод ветвей и отсечений остаётся мощным инструментом для решения задачи PDPTW в условиях, когда важно найти точное оптимальное решение. Однако на практике, когда вычислительные ресурсы и время ограничены, стремление к нахождению точного решения не всегда целесообразно, и достаточно найти какое-то хорошее решение.

Библиография

[1] Toth, P., & Vigo, D. (2002). An Overview of Vehicle Routing Problems. The Vehicle Routing Problem, 1–26.

[2] Laporte, G. (2009). Fifty Years of Vehicle Routing. Transportation Science, 43(4), 408–416.

[3] Semet, F., & Taillard, E. (1993). Solving real-life vehicle routing problems efficiently using tabu search. Annals of Operations Research, 41(4), 469–488.

[4] Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. Computers & Operations Research, 34(8), 2403–2435.

[5] Naccache, S., Côté, J.-F., & Coelho, L. C. (2018). The multi-pickup and delivery problem with time windows. European Journal of Operational Research, 269(1), 353–362.

[6] Mancini, S. (2016). A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based methaeuristic. Transportation Research Part C: Emerging Technologies, 70, 100–112.

[7] Cordeau, J.-F. & Maischberger, M. (2012). A parallel iterated tabu search heuristic for vehicle routing problems. Computers & Operations Research, 39(9), 2033–2050.

[8] Ropke, S., & Pisinger, D. (2006). An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. Transportation Science, 40(4), 455–472.

[9] P. Shaw. (1998). Using constraint programming and local search methods to solve vehicle routing problems. International Conference on Principles and Practice of Constraint Programming, 417–431.

[10] Aziez, I., Côté, J.-F., & Coelho, L. C. (2020). Exact algorithms for the multi-pickup and delivery problem with time windows. European Journal of Operational Research, 284(3), 906–919

[11] Cordeau, J.-F. (2006). A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. Operations Research, 54(3), 573-586.
[12] Ropke, S., and Cordeau, J. F. (2009). Branch and cut and price for the pickup and delivery problem with time windows. Transportation Science, 43, 267–286.
[13] Ropke, S. & Cordeau, J.-F. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks, 49(4), 258-272.
[14] Lu, Q., & Dessouky, M. (2004). An exact algorithm for the multiple vehicle pickup and delivery problem. Transportation Science, 38, 503–514.
[15] Furtado, M. G. S., Munari, P., & Morabito, R. (2017). Pickup and delivery problem with time windows: A new compact two-index formulation. Operations Research Letters, 45(4), 334–341.
[16] Ascheuer, N., Fischetti, M., & Grötschel, M. (2001). Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. Mathematical Programming, 90(3), 475–506.
[17] Li, H & Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. International Journal of Artificial Intelligence Tools, 12(2), 160–170.
[18] Набор данных Li & Lim Benchmark. [Электронный ресурс]. URL: https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmark/ (дата обращения: 01.02.2025)
[19] Приложение. Результаты расчетов. [Электронный ресурс]. URL: https://docs.google.com/spreadsheets/d/1Dw42k1hHC4vJEfgBg6Q7IV4XkfCALOP_t6QpClgOuoA/edit?usp=sharing (дата обращения: 01.02.2025)
[20] D. Ge, Q. Huangfu, Z. Wang, J. Wu and Y. Ye. (2023). Cardinal Optimizer (COPT) user guide. [Электронный ресурс]. URL: https://guide.coap.online/copt/en-doc (дата обращения 01.02.2025)
[21] Baldacci, R., Bartolini, E., & Mingozzi, A. (2011). An Exact Algorithm for the Pickup and Delivery Problem with Time Windows. Operations Research, 59(2), 414–426.
[22] Sartori, C. S., & Buriol, L. S. (2020). Instances for the Pickup and Delivery Problem with Time Windows based on open data. Mendeley Data, V2. [Электронный ресурс]. URL: https://data.mendeley.com/datasets/wr2ct4r22f/2 (дата обращения 01.02.2025)

On methods for solving pickup and delivery problem with time windows. Part I: exact approach

N. Tsarkov, D. Golembiovsky

Abstract — **The paper considers an exact approach based on the branch-and-cut method to solve the pickup and delivery problem with time windows (PDPTW), an important class of vehicle routing problems in transportation logistics. Four mathematical formulations are analyzed in detail: 3-index, 2-index with pairwise and order constraints, compact 2-index and asymmetric representatives one, each of which has its own characteristics in terms of computational complexity and symmetry of solutions. Numerical experiments based on a modified Li & Lim dataset demonstrate that the compact 2-index formulation has the lowest computational complexity and the best speed of finding the exact solution, especially when using efficient ILP solvers. The paper is aimed at researchers and professionals interested in applying exact methods to solve PDPTW.**

Keywords — **VRP, PDPTW, ILP, logistics, linear programming, branch cut.**

REFERENCES

[1] Toth, P., & Vigo, D. (2002). An Overview of Vehicle Routing Problems. The Vehicle Routing Problem, 1–26.
[2] Laporte, G. (2009). Fifty Years of Vehicle Routing. Transportation Science, 43(4), 408–416.
[3] Semet, F., & Taillard, E. (1993). Solving real-life vehicle routing problems efficiently using tabu search. Annals of Operations Research, 41(4), 469–488.
[4] Pisinger, D., & Ropke, S. (2007). A general heuristic for vehicle routing problems. Computers & Operations Research, 34(8), 2403–2435.
[5] Naccache, S., Côté, J.-F., & Coelho, L. C. (2018). The multi-pickup and delivery problem with time windows. European Journal of Operational Research, 269(1), 353–362.
[6] Mancini, S. (2016). A real-life multi depot multi period vehicle routing problem with a heterogeneous fleet: Formulation and adaptive large neighborhood search based methaeuristic. Transportation Research Part C: Emerging Technologies, 70, 100–112.
[7] Cordeau, J.-F. & Maischberger, M. (2012). A parallel iterated tabu search heuristic for vehicle routing problems. Computers & Operations Research, 39(9), 2033–2050.
[8] Ropke, S., & Pisinger, D. (2006). An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. Transportation Science, 40(4), 455–472.
[9] P. Shaw. (1998). Using constraint programming and local search methods to solve vehicle routing problems. International Conference on Principles and Practice of Constraint Programming, 417–431.
[10] Aziez, I., Côté, J.-F., & Coelho, L. C. (2020). Exact algorithms for the multi-pickup and delivery problem with time windows. European Journal of Operational Research, 284(3), 906–919
[11] Cordeau, J.-F. (2006). A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. Operations Research, 54(3), 573-586.
[12] Ropke, S., and Cordeau, J. F. (2009). Branch and cut and price for the pickup and delivery problem with time windows. Transportation Science, 43, 267–286.
[13] Ropke, S. & Cordeau, J.-F. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks, 49(4), 258-272.
[14] Lu, Q., & Dessouky, M. (2004). An exact algorithm for the multiple vehicle pickup and delivery problem. Transportation Science, 38, 503–514.
[15] Furtado, M. G. S., Munari, P., & Morabito, R. (2017). Pickup and delivery problem with time windows: A new compact two-index formulation. Operations Research Letters, 45(4), 334–341.
[16] Ascheuer, N., Fischetti, M., & Grötschel, M. (2001). Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. Mathematical Programming, 90(3), 475–506.
[17] Li, H & Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. International Journal of Artificial Intelligence Tools, 12(2), 160–170.
[18] Nabor dannyh Li & Lim Benchmark. [Jelektronnyj resurs]. URL: https://www.sintef.no/projectweb/top/pdptw/li-lim-benchmark/ (data obrashhenija: 01.02.2025)
[19] Prilozhenie. Rezul'taty raschetov. [Jelektronnyj resurs]. URL: https://docs.google.com/spreadsheets/d/1Dw42k1hHC4vJEfgBg6Q7IV4XkfCALOP_t6QpClgOuoA/edit?usp=sharing (data obrashhenija: 01.02.2025)
[20] D. Ge, Q. Huangfu, Z. Wang, J. Wu and Y. Ye. (2023). Cardinal Optimizer (COPT) user guide. [Jelektronnyj resurs]. URL: https://guide.coap.online/copt/en-doc (data obrashhenija 01.02.2025)
[21] Baldacci, R., Bartolini, E., & Mingozzi, A. (2011). An Exact Algorithm for the Pickup and Delivery Problem with Time Windows. Operations Research, 59(2), 414–426.
[22] Sartori, C. S., & Buriol, L. S. (2020). Instances for the Pickup and Delivery Problem with Time Windows based on open data. Mendeley Data, V2. [Jelektronnyj resurs]. URL: https://data.mendeley.com/datasets/wr2ct4r22f/2 (data obrashhenija 01.02.2025)

Manuscript received May 5, 2025.
Nikita Tsarkov, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University (e-mail: tsarkov90@gmail.com).
Golembiovsky Dmitry, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University (email: golemb@cs.msu.ru)