A Revenue-maximizing Share-a-Ride Problem with Consideration of Recipients' Roaming Locations

Shenglin Liu[a], Qian Ge*[a], Ke Han†[b], Lei Yu[a]

[a]*School of Transportation and Logistics, Southwest Jiaotong University, Xi'an 999, Pidu District, Chengdu, 611756, Sichuan, China*
[b]*School of Economics and Management, Southwest Jiaotong University, No. 111, North Section 1, 2nd Ring Road, Chengdu, 610031, Sichuan, China*

Abstract

This paper addresses the share-a-ride problem with parcels' roaming delivery locations (SARP-PRDL), where a single vehicle fleet serves both passenger and parcel requests, allowing parcels to be sent to one of the multiple candidate locations based on recipients' roaming routes. The problem is divided into static and dynamic request scheduling, both sharing similar model structure and solution routine. We formulate the static problem as a mixed-integer nonlinear program (MINLP) to maximize the service platform's revenue, which is then converted into a mixed-integer linear program (MILP) via linearization. For dynamic requests, we apply a reoptimization strategy to identify potential route improvements. Employing a branch-and-price (B&P) framework, we develop a matheuristic approach and a local search-inspired heuristic for effective column generation in the pricing subproblem. A case study in the Chengdu network shows our solution's effectiveness in static request scheduling. Incorporating roaming delivery locations in static scheduling increases request completions and revenue. For dynamic request scheduling, the proposed vehicle route reoptimization strategy is found to outperform the straightforward greedy insertion approach when fewer idle vehicles are present after the static request scheduling. But this advantage may diminish with the increase of idle vehicles.

Keywords: Share-a-Ride Problem; Roaming delivery locations; Branch-and-Price framework; Matheuristic

1. Introduction

Despite decades of rapid growth, the global e-commerce market continues to expand, with retail e-commerce sales reaching an estimated $5.8 trillion in 2023 and projected to surpass $8 trillion by 2027 (Statista, 2023). E-commerce platforms coordinate most of the world's parcel deliveries, with the number of deliveries expected to reach 256 billion by 2027 (Statista, 2022). This massive volume poses significant challenges to current logistics systems, which remain vulnerable to external disruptions. Moreover, both shippers and recipients now demand higher service quality. Leading platforms like Amazon and Alibaba offer premium services, such as 72-hour, 24-hour, same-day, or even 1-hour delivery, exclusively to "prime" members. While these restrictions align with platforms' revenue strategies via subscription services, they are also driven by the rising demand for fast, reliable last-mile delivery (DHL, 2023).

Challenges in last-mile delivery, including the rising parcel demand and customers' expectations for fast arrivals, have driven e-commerce platforms to strengthen partnerships with logistics providers. Alibaba, for instance, established Cainiao Network to enhance logistics efficiency through smart solutions and digital supply chains. However, despite innovations,

*Corresponding author. E-mail address: geqian@swjtu.edu.cn
†Corresponding author. E-mail address: kehan@swjtu.edu.cn

the high costs and environmental impact of dedicated delivery trucks remain problematic. To address these issues, Arslan et al. (2018) introduced *crowdsourced delivery*, leveraging private passenger vehicles for parcel transport via platforms like Kanga, Renren Kuaidi, and Amazon Flex. This approach offers the potential for more efficient, cost-effective, and sustainable deliveries.

Crowdsourced delivery has gained popularity in both industry and academia (Alnaggar et al., 2021). Researchers suggest that utilizing occasional drivers through crowdshipping effectively addresses environmental and economic challenges in logistics (Mancini et al., 2022; Voigt et al., 2022). However, inefficiency from missed deliveries persists due to the unavailability of recipients at the destinations or at the scheduled delivery time windows, thereby requiring multiple delivery attempts. To address this, Reyes et al. (2017) propose *roaming delivery locations*, where goods are delivered to the trunk of the recipient's vehicle by tracking its position. This concept has been implemented by companies like Volvo and Audi. Volvo, for example, used digital key technology to allow customers to select their cars as delivery points, significantly reducing failed first-attempt deliveries (Volvo Cars, 2014; Audi, 2015).

Integrating crowdshipping with roaming delivery offers a promising solution for efficient, cost-effective, and sustainable deliveries while reducing failed first-attempt deliveries. Taxis, particularly ride-hailing taxis, are well-suited for crowdshipping if the shipments are profitable and passenger services remain unaffected. The stagnation in ride-hailing market growth further motivates drivers to adopt crowdshipping. The simultaneous transportation of passengers and parcels is modeled as the *share-a-ride problem* (SARP) (Li et al., 2014), benefiting drivers, riders, and shippers/recipients alike. SARP has been extensively studied, addressing diverse objectives and complexities, such as carpooling (Yu et al., 2018), stochastic travel times and delivery locations (Li et al., 2016a), multi-depot scenarios (Yu et al., 2023b), and the coordination of ride-hailing with logistics vehicles (Ji et al., 2024).

Building on existing research in share-a-ride and routing problems with roaming delivery, this study explores an integrated taxi-based service for the simultaneous transportation of passengers and parcels. Passengers and shippers/recipients can submit requests specifying pick-up and drop-off locations along with time windows through a dial-a-ride platform managing the taxi fleet. Dynamic requests may also be accommodated during operations, subject to available capacity after fulfilling pre-scheduled requests. When possible, route reoptimization is employed to integrate these dynamic requests. While passengers and parcels may share a vehicle, parcel handling is restricted when passengers are on board to prioritize passenger service. Both pre-scheduled and dynamic requests can register multiple candidate drop-off locations for recipients, offering flexibility in cases where the preferred delivery location is unavailable. This framework supports same-day or rapid 1-2 hour delivery, especially for recipients with mobile schedules. The operator's objective is to optimize task assignments to maximize the total revenue.

The main contributions of this study are as follows:

- We introduce a novel variant of the share-a-ride problem that incorporates parcels' roaming delivery locations and dynamic delivery requests, where each pre-scheduled or dynamic parcel is delivered to exactly one of several candidate locations specified by the recipient. The problem is formulated as a mixed-integer nonlinear programming (MINLP) and transformed into an equivalent mixed-integer linear programming (MILP) through linearization techniques. The MILP formulation can be efficiently solved using off-the-shelf solvers for small-scale instances.

- We propose an efficient matheuristic solution approach based on the branch-and-price (B&P) framework for large-scale instances of the introduced problem. While our approach builds on the existing heuristic for the Vehicle Routing Problem (VRP) and its extensions, we customize it by incorporating a heuristic search procedure within the B&P framework. The problem is reformulated into a path-based model, and our method

iteratively addresses the restricted master problem alongside the pricing subproblem. In each iteration of the pricing subproblem, a promising column is generated through a greedy insertion-based local search procedure.

- We validate the performance of the proposed model and solution approach using real-world instances. The numerical results demonstrate the effectiveness of our approach in scheduling static requests, outperforming the adaptive large neighborhood search (ALNS) algorithm in both solution quality and efficiency. Furthermore, incorporating the roaming delivery location strategy increases the number of completed requests, leading to higher revenue.

The remainder of this paper is organized as follows. Section 2 reviews the related literature. Section 4 presents the share-a-ride problem with roaming delivery locations of parcels. Section 5 is devoted to the B&P-based matheuristic solution method for the SARP-PRDL. Section 6 summarizes the computational results of the proposed model and solution method. Finally, section 7 concludes this work.

2. Literature review

We review the literature related to this study based on three parts: share-a-ride problem, last-mile delivery with roaming delivery locations and dynamic vehicle routing problem with reoptimization of vehicle routes.

2.1. Share-a-ride problem

The Share-a-Ride Problem extends the Dial-a-Ride Problem (DARP) by incorporating vehicle routing and scheduling for passengers who specify their pick-up and drop-off locations within certain time windows (Cordeau et al., 2007). For a comprehensive understanding of the DARP, readers are referred to Ho et al. (2018). Distinct from the DARP, the SARP integrates parcel delivery into passenger scheduling.

The seminal work on the SARP by Li et al. (2014) introduces a mixed-integer linear programming model, solvable with commercial solvers for small-scale instances, and a greedy heuristic for larger instances by integrating parcel requests into predefined passenger routes. Li et al. (2016a) extend SARP by incorporating stochastic travel times and delivery locations, while Li et al. (2016b) propose an ALNS algorithm, outperforming commercial solvers. Yu et al. (2018) introduce the General Share-a-Ride Problem (G-SARP), allowing multiple passengers, which is further extended to multi-depot settings in Yu et al. (2023b). Considering vehicle heterogeneity, Lu et al. (2022) employ a mixed fleet of electric and gasoline vehicles using a time-expanded network and MILP formulation. Yu et al. (2023a) design a novel matheuristic approach combining simulated annealing with mutation strategies (SAMS) and a set partitioning method, outperforming SAMS on large-scale instances. Zhan et al. (2023) explore passenger-and-parcel sharing with a mixed fleet of vehicles and electric motorcycles, proposing a two-level lexicographic multi-objective function from the platforms perspective. More recently, Gao et al. (2024) study the stochastic SARP in ride-hailing systems, prioritizing passengers under uncertain requests. Ji et al. (2024) develop an exact framework using a mixed fleet of ride-hailing and logistics vehicles, demonstrating potential in high-demand scenarios for integrated passenger and parcel services.

2.2. Last-mile delivery with roaming delivery locations

The concept of *roaming delivery location* is initially introduced in vehicle routing problem within the context of last-mile delivery. The pioneer work on vehicle routing problem with roaming delivery locations (VRPRDL) by Reyes et al. (2017) presents a mixed integer

programming formulation aimed at minimizing travel cost. They also design a neighborhood-based search combining construction and improvement heuristic algorithm to solve the VRPRDL based on the problem-specific techniques. Following this, Ozbaygin et al. (2017) reformulate the arc-based model as a set partition model and design a branch-and-price algorithm to address the VRPRDL, with considering a hybrid delivery strategy allowing a delivery to either the customer's home or to the trunk of customer's vehicle. Based on previous work, Ozbaygin et al. (2019) establish an iterative reoptimization framework to tackle the last-mile delivery problem. Their proposed framework adjusts sub-optimal or even infeasible solutions of planned delivery schedule incurred by the change of customer itineraries to feasible solutions, thereby adapting dynamic scenarios. He et al. (2020) investigate the last-mile delivery system with roaming delivery locations and stochastic travel times, formulating a two-stage stochastic programming model and devising an effective metaheuristic algorithm that integrates a sampling strategy. Dragomir et al. (2022) expand the VRPRDL by considering the Pickup and Delivery Problem (PDP) with alternative locations and overlapping time windows, applying a multi-start, adaptive large neighborhood search with problem-specific operators to address the problem.

2.3. Dynamic vehicle routing problem with reoptimization of vehicle routes

Dynamic vehicle routing problem (DVRP) differs from classical vehicle routing due to its requests not fully known in advance. These requests arrive dynamically during the execution of planned routes, which may change existing routes if there is the insertion of requests. For a comprehensive review of the DVRP, one may refer to Zhang et al. (2023). When it comes to solution approach, there are many studies proposing reoptimization strategies to solve DVRP.

Chen and Xu (2006) extended column generation approach for solving dynamic vehicle routing problem. Their proposed approach generates new columns in real-time by leveraging previously generated columns, and at each decision point, it solves a set-partitioning formulation using the columns generated up to that moment. In Vonolfen and Affenzeller (2016), a reoptimization procedure begins when any new request arrives. They use a constructive heuristic to insert the new request into the best location of existing routes, and then a tabu search heuristic is applied to improve new route plan. Finally, they update waiting time at each location based on intensity-based waiting heuristic strategy. For the DVRP with roaming delivery locations, Ozbaygin et al. (2019) establish an iterative reoptimization framework to tackle the potential problem that sub-optimal or even infeasible solution of planned delivery schedule may be incurred by the change of customer itineraries. Steever et al. (2019) develop an auction-based heuristic approach to achieve the arrival-triggered reoptimization, and this approach identifies solutions that are both effective in meeting customer needs in the present and in preparing couriers to respond to future demand. Archetti et al. (2021) design a solution approach based on an insertion algorithm evaluating each request singularly. This insertion method is a foundation of their proposed reoptimization approach using variable neighborhood search algorithm. Sze et al. (2024) propose adaptive variable neighborhood search algorithm to generate routes in static environment and reoptimize routes based on the traffic information of urgency of the accidents.

This paper distinguishes itself from prior studies in two key ways: (a) We identify a new variant of the SARP, which accounts for roaming delivery locations. If a parcel cannot be delivered to the recipients preferred location, it can be redirected to alternate candidate locations. Additionally, we develop a tailored matheuristic approach for static request scheduling and implement a reoptimization strategy to accommodate dynamic requests within the existing schedule. (b) Our model supports multiple sharing modespassenger-passenger, passenger-parcel, and parcel-parcel sharingwhile prioritizing passenger service by restricting parcel operations when passengers are on board.

3. Problem statement

We study a share-a-ride problem with parcels' roaming delivery locations (SARP-PRDL), where passengers and parcels can be transported by the same vehicle fleet. In this study, we consider a dial-a-ride service platform serving both passenger and parcel requests. Both passengers and parcels being transported within the same city. Parcel requests belong to express delivery within the same city, meaning that parcels must be picked up and delivered on the same day. Furthermore, the parcels should be delivered exactly based on recipients' current locations.

Customers may submit static requests before the service begins. A static passenger request includes the trips origin, destination, pickup time window within the platform's specified operating hours, latest arrival time, and any personal belongings. Similarly, shippers submit static parcel requests with details such as the pickup location and parcel size, while recipients specify delivery locations and time windows. Recipients may register multiple delivery locations based on their planned stops for the day. The platform optimizes schedules to maximize the fulfillment of these static requests. Additionally, dynamic requests can be submitted during operations, with feasible ones integrated through route reoptimization of remaining segments.

In a share-a-ride transport mode, three sharing types are permitted within the same vehicle: passenger-and-passenger, passenger-and-parcel, and parcel-and-parcel sharing. However, detours from passenger-and-passenger and passenger-and-parcel sharing can impact the passenger experience. To mitigate such inconvenience, parcel pickups and deliveries are performed only when no passengers are on board, ensuring that passengers are prioritized, and their travel experience remains unaffected. Hence, a passenger's travel experience may only be influenced by the ride-sharing with other passengers.

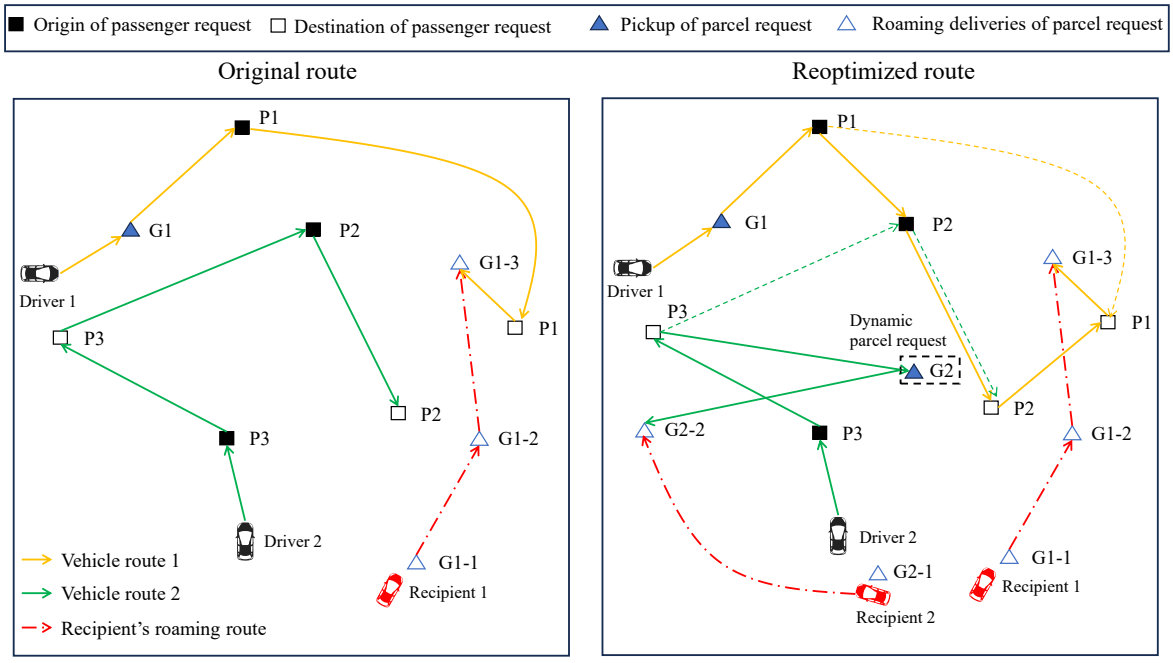


Figure 3.1: An illustrative example of share-a-ride operation mode. **Left**: original route scheme in the time period $[0, t]$. **Right**: reoptimized route scheme in the time period $[t^*, T]$.

As shown in Figure 3.1 and Figure 3.2, we present an example with three passenger requests and two parcel requests to demonstrate the share-a-ride mode. The former figure displays the scheduled routes before and after the appearance of a dynamic request "G2" while the latter figure presents the corresponding timelines based on these routes. In the initial plan, Passenger 2 is assigned to Driver 2; however, with the introduction of a reoptimization procedure, this request is reassigned to Driver 1's route. In the updated schedule, Driver 2, after completing

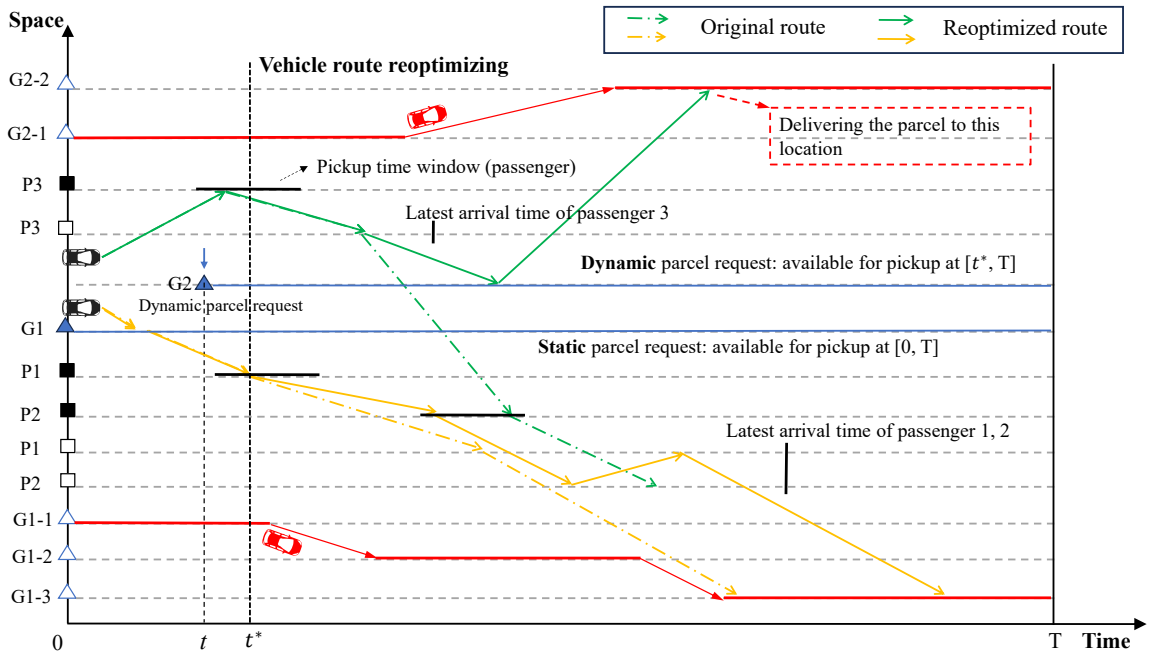


Figure 3.2: An illustrative space-time figure of original and reoptimized vehicle routes.

Passenger Request 3, serves Parcel Request 2 instead of Passenger Request 2. Assuming capacity constraints are met, the reoptimized route accommodates all requests without violating time constraints.

The travel fare for each passenger or shipper is determined by multiplying the fare per unit distance by the shortest travel distance between the specified origin and destination. Passengers who experience inconvenience due to detours will receive a fare discount (e.g., 20% off the original fare), whereas parcels are not eligible for this discount.

In this study, the revenue of the dial-a-ride platform is derived from the total travel fares paid by passengers and shippers. Maximizing this revenue is established as the primary objective of the SARP-PRDL. Given a specific number of vehicles, we aim to determine the optimal routes for picking up and delivering passengers and parcels to achieve this maximum revenue. The following basic assumptions are made for this problem:

(a) Passengers and parcel recipients submit their requests through the service platform. Additionally, recipients must provide their scheduled routes or stops along with their submissions. Parcel suppliers are also required to share their pickup locations with the platform.

(b) A vehicle cannot pick up or drop off parcels with passengers onboard.

(c) The parcel cannot be delivered to a candidate location if the recipient is unavailable, whether the recipient have already moved on to the next stop or have not yet arrived at that location.

(d) Each vehicle travels at a constant speed.

4. Mathematical model formulation

In this section, we present the mathematical model formulation for the share-a-ride problem with parcels' roaming delivery locations. We begin by introducing and clarifying the notation used in the model. Following this, we define the objective function and outline a series of constraints within the mathematical model, which collectively determine vehicle routes and the allocation of vehicles to requests in an objective manner.

4.1. Notations

The notations used in the model are listed in Table 1. The road network is represented as a complete directed graph $G(N, E)$ where $N = \mathcal{M} \cup (\bigcup_{k\in\mathcal{K}} k^+) \cup (\bigcup_{k\in\mathcal{K}} k^-)$. All origins and destinations of passenger and parcel requests are included in N. The shortest distance between any two vertices can be pre-calculated, and it is assumed that the distance is symmetric. For each vehicle, it has an initial point and a dummy point, which corresponds to the last destination of the requests to be delivered. The model also incorporates various characteristics of each vehicle, including maximum capacity for passengers, maximum size for luggage and parcels, and average speed.

4.2. Static mathematical model

The mathematical model aims to maximize the total revenue of the platform charged from travel fares of requests (including passenger and parcel requests) and determines the vehicle routes based on the objective value.

Before introducing the model, we formally define the roaming delivery locations as follows. We let $\mathcal{M}_{c^-}$ be the set of roaming delivery locations of the parcel request c, and each location $m \in \mathcal{M}_{c^-}$ has a non-overlapping time window $[ET_m, LT_m]$. There exists a sequence of location $m_1, m_2, ..., m_k \in \mathcal{M}_{c^-}$ based on time windows if the number of vertices in $\mathcal{M}_{c^-}$ is k. Reyes et al. (2017) assumed that the time period for serving requests was $[0, T]$ and let all roaming delivery locations belonging to a same request form a circle, which means that $m_1 = m_k$. Different from their assumption, we set a roaming route belonging to a recipient, which does not require $m_1 = m_k$. Meanwhile, time windows of the vertices in $\mathcal{M}_{c^-}$ satisfy:

$$ET_{m_1} = 0 \tag{4.1}$$

$$LT_{m_k} = T \tag{4.2}$$

$$ET_{m_j} = LT_{m_{j-1}} + \frac{L_{m_{j-1},m_j}}{V}, \quad j = 2, ..., k \tag{4.3}$$

4.2.1. Static formulation

Definition 4.1 (Static requests)**.** *Assuming that a vehicle fleet needs to serve a series of passenger and parcel requests in the time period $T = [\underline{t}, \overline{t}]$, these requests must be submitted to platform in advance. All the requests submitted before the time point $\underline{t}$ are considered static requests. This clarification facilitates the design of vehicle routes based on the information available in advance, optimizing task assignment before service commencement.*

Given a set of static requests, the arc-based model of the SARP-PRDL is formulated as follows. This model incorporates several critical constraints, including capacity, arrival time, and visiting sequence constraints, to ensure the feasibility and efficiency of the proposed routing solutions.

Objective

$$\max \quad \alpha(\sum_{r\in\mathcal{R}} p_r + \sum_{c\in\mathcal{C}} p_c) \tag{4.4}$$

The maximum profit of platform is obtained based on the maximum travel fares paid by passengers and parcel owners, and α is the percentage of commission drawn by platform from travel fares (e.g., 20%). In this study, the travel fare p_r paid by a passenger r is defined as follows:

$$\begin{aligned} p_r &= \beta_R L_r \sum_{k\in\mathcal{K}} \sum_{v\in\mathcal{M}} x^k_{r^+,v}(y^k_{r^+} + \gamma(1 - y^k_{r^+})) \\ &= \gamma\beta_R L_r \sum_{k\in\mathcal{K}} \sum_{v\in\mathcal{M}} x^k_{r^+,v} + (1-\gamma)\beta_R L_r \sum_{k\in\mathcal{K}} \sum_{v\in\mathcal{M}} x^k_{r^+,v} y^k_{r^+}, \end{aligned} \tag{4.5}$$

Table 1: Key parameters and variables

Indices and sets	
$\mathcal{R}$	Set of passenger requests
$\mathcal{C}$	Set of parcel requests
$\mathcal{R}^o$	Set of origins of passenger requests in $\mathcal{R}$
$\mathcal{R}^d$	Set of destinations of passenger requests in $\mathcal{R}$
$\mathcal{C}^o$	Set of origins of parcel requests in $\mathcal{C}$
$\mathcal{C}^d$	Set of destinations of parcel requests in $\mathcal{C}$
$\mathcal{K}$	Set of vehicles
E	Set of edges in the graph
u	Index of requests
k	Index of vehicles
k^+	Starting point of the vehicle k
k^-	Dummy point the vehicle k
r^+	Origin of the passenger request r
r^-	Destination of the passenger request r
c^+	Origin of the parcel request c
$\mathcal{M}_{c^-}$	Set of roaming delivery locations of the parcel request c, $c \in \mathcal{C}$, $\mathcal{M}_{c^-} \subseteq \mathcal{C}^d$
$\mathcal{M}$	Set of origins and destinations of passenger requests and parcel requests, $\mathcal{M} = \mathcal{R}^o \cup \mathcal{R}^d \cup \mathcal{C}^o \cup \mathcal{C}^d$
N	Set of origins and destinations of passenger requests, parcel requests, and vehicles, $N = \mathcal{M} \cup (\bigcup_{k\in\mathcal{K}} k^+) \cup (\bigcup_{k\in\mathcal{K}} k^-)$
Parameters and constants	
α	Percentage of commission drawn by platform from travel fares paid by passengers and parcel owners
β_R	Travel fare for passengers per unit travel distance
β_C	Travel fare for parcel owners per unit travel distance
γ	Fare discount percentage for passengers when taking a detour
$L_{i,j}$	Shortest travel distance from vertex i to vertex j
V	Average speed of vehicles
ET_j	Earliest pickup time of the vertex j
LT_j	latest pickup time of the vertex j
LAT_u	Latest arrival time of the request u
n_u	Number of passengers of request u
q_u	Size of luggage or parcel of request u
g_j	Service time at the vertex j, $j \in \mathcal{M}$
Φ_k	Maximum capacity of vehicle k for passengers
Ψ_k	Maximum size of vehicle k for parcels
M	A sufficiently large number
Decision variables	
$x^k_{i,j}$	Equals 1 if vehicle k passes from vertex i to vertex j, and 0 otherwise
$y^k_{r^+}$	Equals 1 if vehicle k serves a passenger request r without detouring, and 0 otherwise
τ^k_j	Departure time of vehicle k from vertex j
C^k_j	Number of passengers on vehicle k after vehicle k departs from vertex j
Q^k_j	Size of luggage and parcels on vehicle k after vehicle k departs from vertex j
μ_j	Visiting order of vertex j
Auxiliary variable	
$w^k_{r^+,j}$	Equals 1 if both $x^k_{r^+,j}$ and $y^k_{r^+}$ are equal to 1, and 0 otherwise

where β_R is the travel fare for passengers per unit travel distance, and L_r is the shortest travel distance of the passenger request r between its origin and destination. This equation indicates that a passenger will obtain the fare discount due to an inevitable detour if the passenger r shares a ride with other passengers. But no discount will be applied to travel fares for parcel requests, regardless of whether there exists a detour or not. Similarly, the travel fare p_c paid by a parcel owner c is:

$$p_c = \sum_{k\in\mathcal{K}} \sum_{m\in\mathcal{M}_{c^-}} \sum_{v\in\mathcal{M}\setminus\mathcal{M}_{c^-}} \beta_C L_{cm} x^k_{v,m}, \tag{4.6}$$

where β_C is the travel fare for parcel owners per unit travel distance, and L_{cm} is the shortest travel distance of the parcel request c between its origin and roaming delivery location m. The travel fare for a parcel request may vary depending on different roaming delivery locations.

Noted that our objective function is a nonlinear integer programming due to the equation 4.5, but it can be linearized by introducing an auxiliary variable $w^k_{r^+,v}$ and some linearization constraints. Here we add the following constraints:

$$w^k_{r^+,v} = x^k_{r^+,v} y^k_{r^+}, \forall r \in \mathcal{R}, v \in \mathcal{M}, k \in \mathcal{K} \tag{4.7}$$
$$w^k_{r^+,v} \leq x^k_{r^+,v}, \forall r \in \mathcal{R}, v \in \mathcal{M}, k \in \mathcal{K} \tag{4.8}$$
$$w^k_{r^+,v} \leq y^k_{r^+}, \forall r \in \mathcal{R}, v \in \mathcal{M}, k \in \mathcal{K} \tag{4.9}$$
$$w^k_{r^+,v} \geq x^k_{r^+,v} + y^k_{r^+} - 1, \forall r \in \mathcal{R}, v \in \mathcal{M}, k \in \mathcal{K} \tag{4.10}$$
$$x^k_{r^+,v}, y^k_{r^+}, w^k_{r^+,v} \in \{0,1\} \tag{4.11}$$

Then, our objective function can be rewritten as follows:

$$\begin{aligned} \max \quad & \alpha \Big[\sum_{r \in \mathcal{R}} \Big(\gamma \beta_R L_r \sum_{k \in \mathcal{K}} \sum_{v \in \mathcal{M}} x^k_{r^+,v} + (1-\gamma) \beta_R L_r \sum_{k \in \mathcal{K}} \sum_{v \in \mathcal{M}} x^k_{r^+,v} y^k_{r^+} \Big) + \\ & \sum_{c \in \mathcal{C}} \sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}_{c^-}} \sum_{v \in \mathcal{M} \setminus \mathcal{M}_{c^-}} \beta_C L_{cm} x^k_{v,m} \Big] \end{aligned} \tag{4.12}$$

Constraints

$$\sum_{k \in \mathcal{K}} \sum_{v \in \mathcal{M}} x^k_{u^+,v} \leq 1, \forall u \in \mathcal{R} \cup \mathcal{C} \tag{4.13}$$
$$\sum_{v \in \mathcal{M}} x^k_{k^+,v} = 1, \forall k \in \mathcal{K} \tag{4.14}$$
$$\sum_{v \in \mathcal{M}} x^k_{v,k^-} = 1, \forall k \in \mathcal{K} \tag{4.15}$$
$$\sum_{k \in \mathcal{K}} \sum_{v \in \mathcal{M}} x^k_{k^+,v} = \sum_{k \in \mathcal{K}} \sum_{v \in \mathcal{M}} x^k_{v,k^-} = 1 \tag{4.16}$$
$$\sum_{v \in \mathcal{M}} x^k_{r^+,v} - \sum_{v \in \mathcal{M}} x^k_{v,r^-} = 0, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.17}$$
$$\sum_{v \in \mathcal{M}} x^k_{c^+,v} - \sum_{m \in \mathcal{M}_{c^-}} \sum_{v \in \mathcal{M} \setminus \mathcal{M}_{c^-}} x^k_{v,m} = 0, \forall c \in \mathcal{C}, k \in \mathcal{K} \tag{4.18}$$
$$\sum_{v \in N} x^k_{i,v} - \sum_{v \in N} x^k_{v,i} = 0, \forall i \in \mathcal{M}, k \in \mathcal{K} \tag{4.19}$$
$$\sum_{k \in \mathcal{K}} \sum_{m \in \mathcal{M}_{c^-}} \sum_{v \in \mathcal{M} \setminus \mathcal{M}_{c^-}} x^k_{v,m} \leq 1, \forall c \in \mathcal{C} \tag{4.20}$$
$$\sum_{v \in (\mathcal{C}^o \cup \mathcal{C}^d)} x^k_{r^+,v} + \sum_{v \in (\mathcal{C}^o \cup \mathcal{C}^d)} x^k_{v,r^-} = 0, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.21}$$
$$\sum_{v \in \mathcal{R}^d} x^k_{c^+,v} + \sum_{m \in \mathcal{M}_{c^-}} \sum_{v \in \mathcal{M} \setminus (\mathcal{R}^d \cup \{c^+\})} x^k_{v,m} = 0, \forall c \in \mathcal{C}, k \in \mathcal{K} \tag{4.22}$$
$$\mu_{r^+} - \mu_v + 1 + M(\sum_{k \in \mathcal{K}} x^k_{r^+,v} - 1) \leq 0, \forall r \in \mathcal{R}, v \in (\mathcal{R}^o \cup \mathcal{R}^d) \setminus \{r^+\} \tag{4.23}$$
$$\mu_v - \mu_{r^-} + 1 + M(\sum_{k \in \mathcal{K}} x^k_{v,r^-} - 1) \leq 0, \forall r \in \mathcal{R}, v \in (\mathcal{R}^o \cup \mathcal{R}^d) \setminus \{r^-\} \tag{4.24}$$
$$\mu_{c^+} - \mu_v + 1 + M(\sum_{k \in \mathcal{K}} x^k_{c^+,v} - 1) \leq 0, \forall c \in \mathcal{C}, v \in \mathcal{C}^o \cup \mathcal{C}^d \cup \mathcal{R}^o \tag{4.25}$$

$$\mu_v - \mu_m + 1 + M(\sum_{k\in\mathcal{K}} x^k_{v,m} - 1) \leq 0, \forall m \in \mathcal{M}_{c^-}, v \in \mathcal{R}^d \cup \{c^+ | c^+ \in \mathcal{C}^o\} \tag{4.26}$$

$$\tau^k_v \geq \tau^k_{u^+} + \frac{L_{u^+,v}}{V} + g_v + M(x^k_{u^+,v} - 1), \forall k \in \mathcal{K}, u \in \mathcal{R} \cup \mathcal{C}, v \in \mathcal{M} \tag{4.27}$$

$$\tau^k_{r^-} \geq \tau^k_v + \frac{L_{v,r^-}}{V} + g_{r^-} + M(x^k_{v,r^-} - 1), \forall k \in \mathcal{K}, r \in \mathcal{R}, v \in \mathcal{M} \tag{4.28}$$

$$\tau^k_m \geq \tau^k_v + \frac{L_{v,m}}{V} + g_m + M(x^k_{v,m} - 1), \forall k \in \mathcal{K}, m \in \mathcal{M}_{c^-}, c \in \mathcal{C}, v \in \mathcal{M} \tag{4.29}$$

$$\tau^k_{r^+} - \tau^k_{r^-} \leq 0, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.30}$$

$$\tau_{c^+} - \tau_m - M(1 - \sum_{v\in\mathcal{M}\setminus\mathcal{M}_{c^-}} x^k_{v,m}) \leq 0, \forall m \in \mathcal{M}_{c^-}, c \in \mathcal{C}, k \in \mathcal{K} \tag{4.31}$$

$$ET_{u^+} \sum_{v\in\mathcal{M}} x^k_{u^+,v} \leq \tau^k_{u^+} \leq LT_{u^+} \sum_{v\in\mathcal{M}} x^k_{u^+,v}, \forall u \in \mathcal{R} \cup \mathcal{C}, k \in \mathcal{K} \tag{4.32}$$

$$\tau^k_{r^-} \leq LAT_r, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.33}$$

$$\sum_{m\in\mathcal{M}_{c^-}} ET_m \sum_{v\in\mathcal{M}} x^k_{m,v} \leq \tau^k_m \leq \sum_{m\in\mathcal{M}_{c^-}} LT_m \sum_{v\in\mathcal{M}} x^k_{m,v}, \forall m \in \mathcal{M}_{c^-}, c \in \mathcal{C}, k \in \mathcal{K} \tag{4.34}$$

$$C^k_{u^+} \geq C^k_v + n_u + M(x^k_{v,u^+} - 1), \forall v \in \mathcal{M}, u \in \mathcal{R} \cup \mathcal{C}, k \in \mathcal{K} \tag{4.35}$$

$$C^k_{r^-} \geq C^k_v - n_r + M(x^k_{v,r^-} - 1), \forall v \in \mathcal{M}, r \in \mathcal{R}, k \in \mathcal{K} \tag{4.36}$$

$$C^k_m \geq C^k_v - n_c + M(x^k_{v,m} - 1), \forall v \in \mathcal{M}, m \in \mathcal{M}_{c^-}, c \in \mathcal{C}, k \in \mathcal{K} \tag{4.37}$$

$$Q^k_{u^+} \geq Q^k_v + q_u + M(x^k_{v,u^+} - 1), \forall v \in \mathcal{M}, u \in \mathcal{R} \cup \mathcal{C}, k \in \mathcal{K} \tag{4.38}$$

$$Q^k_{r^-} \geq Q^k_v - q_r + M(x^k_{v,r^-} - 1), \forall v \in \mathcal{M}, r \in \mathcal{R}, k \in \mathcal{K} \tag{4.39}$$

$$Q^k_m \geq Q^k_v - q_c + M(x^k_{v,m} - 1), \forall v \in \mathcal{M}, m \in \mathcal{M}_{c^-}, c \in \mathcal{C}, k \in \mathcal{K} \tag{4.40}$$

$$C^k_v \leq \Phi_k, \forall v \in \mathcal{M}, k \in \mathcal{K} \tag{4.41}$$

$$Q^k_v \leq \Psi_k, \forall v \in \mathcal{M}, k \in \mathcal{K} \tag{4.42}$$

$$\sum_{k\in\mathcal{K}} y^k_{r^+} \leq 1, \forall r \in \mathcal{R} \tag{4.43}$$

$$M(1 - y_{r^+}) \geq C^k_{r^+} - n_r, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.44}$$

$$-My_{r^+} \leq C^k_{r^+} - n_r, \forall r \in \mathcal{R}, k \in \mathcal{K} \tag{4.45}$$

Constraints (4.13) guarantee that each request can be matched to at most one vehicle. Constraints (4.14) - (4.16) enforce that each vehicle departs from its starting point and end at its dummy point, and is not permitted to pass starting points or dummy points of other vehicles. Constraints (4.17) and (4.18) ensure that the origin and destination of request u should be visited by the same vehicle. Constraint (4.19) is the flow conservation constraint. Constraint (4.20) requires that at most one roaming delivery location of the parcel request c can be visited. Constraints (4.21) and (4.22) eliminate infeasible edges in the Graph G. Constraints (4.23) - (4.26) determine the order in which vertices are visited and eliminate closed loops of vehicle routes. Constraints (4.27) - (4.29) require that the vehicle k leaves a vertex later than arriving at one. Constraints (4.30) and (4.31) ensures that the departure time is later than the arrival time for all requests (including passenger requests and parcel requests). Constraints (4.32) - (4.34) set the time window for each vertex belonging to the set of requests. Constraints (4.35) - (4.37) determine the number of passengers in each vehicle when the vehicle picks up or delivers at a vertex. Constraints (4.38) - (4.40) determine the size of luggage and parcels in each vehicle when the vehicle picks up or delivers at a vertex. Constraint (4.41) and Constraint (4.42) set limits on the number of passengers and the total size of luggage or parcels that each vehicle can carry. Constraint (4.43) guarantees that there is at most one vehicle picking up a passenger request r if the passenger has a trip without a

detour. Constraints (4.44) and (4.45) indicate that there is only one passenger in the vehicle if this passenger has a trip without any detour.

4.2.2. A simple computational example in Manhattan grid network

In this subsection, we employ a simplified road network to evaluate our arc-based model in addressing the SARP-PRDL. The road network is structured as a 10 km × 10 km Manhattan grid, with each edge representing a distance of 1 km. Drawing on a static illustrative instance from Zhan et al. (2023), we utilize a subset of data generated randomly, which includes the origins and destinations of passenger and parcel requests, the size of each parcel request, the luggage size associated with each passenger request, and the starting locations of the vehicles. Additionally, the roaming delivery locations for each parcel request are also generated randomly. Given the nature of the proposed road network, the shortest distance between two vertices is calculated using the Manhattan metric.

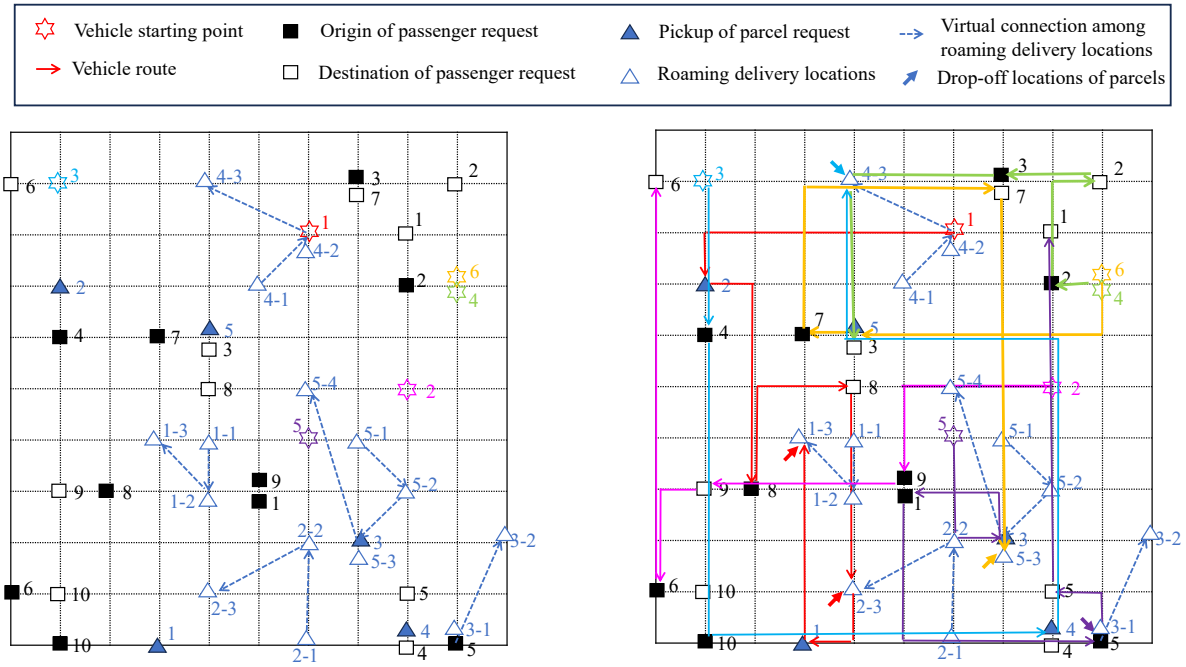


Figure 4.1: Left: Manhattan grid network comprising 10 passenger requests, 5 parcel requests, and 6 vehicles; **Right**: the optimal routes of solving instance using Gurobi.

Figure 4.1 on the left depicts our designed network comprising 10 passenger requests, 5 parcel requests, and 6 vehicles, corresponding to the instance "P10-G5-V6" in the Table 2. In this instance, each parcel request has at least 2 roaming delivery locations. Dashed blue lines are used to connect roaming delivery locations of each parcel in the Figure 4.1. It should be noted that the dashed blue line does not represent the actual moving route. Rather, it just depicts the sequence of movement at the roaming delivery locations. This small instance can be solved by Gurobi solver 10.0.0 and Figure 4.1 on the right depicts optimal routes of 6 vehicles. In this optimal solution, passenger request 10 is not served due to time constraints.

Then, we use the Gurobi solver to test the model on a set of randomly generated instances, as shown in Table 2. The small-scale instances are denoted by names in the format of "Px-Gy-Vz", where "x" represents the number of passenger requests, "y" indicates the number of parcel requests, and "z" denotes the number of vehicles. Table 2 demonstrates that the Gurobi solver efficiently finds the optimal solution in all small-scale instances with 5 parcel requests. However, as the number of parcel requests increases to 10, the solver fails to find an optimal solution within 7,200 seconds. This inefficiency is attributed to the exponential increase in the number of vertices, resulting from the roaming delivery locations associated with each parcel request.

Table 2: Computational results in small-scale instances by using Gurobi

Instances	NoV	Gurobi		
		Incumbent	Time(s)	Gap(%)
P5-G5-V6	30+12	31.6	0.6	0
P10-G5-V6	40+12	42.3	208	0
P10-G10-V6	58+12	49.6	>7200	19.15
P15-G5-V6	50+12	59.2	>7200	12.16
P15-G10-V6	68+12	64	>7200	32.2
P5-G5-V10	30+20	31.6	0.2	0
P10-G5-V10	40+20	43.6	1.7	0
P10-G10-V10	58+20	58.8	>7200	1.36
P15-G5-V10	50+20	68.6	8.7	0
P15-G10-V10	68+20	77.4	>7200	9.30

Notes:
NoV: the number of vertices in the graph, as denoted in the format of "A+B". **A**: the number of vertices consist of passenger and parcel requests. **B**: the number of vertices consist of vehicle starting points and dummy ending points.

4.2.3. A path-based formulation

Similar to the path-based formulation for VRP, SARP-PRDL can also be viewed as finding a set of optimal routes for vehicles to serve both passengers and parcels, except that not all vertices need to be visited. Therefore, the arc-based model for SARP-PDRL can be reformulated as a set-packing model by the Dantzig-Wolfe decomposition technique (Alidaee et al., 2008). This reformulation decomposes the original model into a master problem and a series of subproblems. Each subproblem identifies a promising route for a vehicle, while the master problem selects the optimal combination of these routes.

We let $\mathcal{S}$ denote the set of vehicle routes, and ϑ_s denote a binary variable indicating whether the route $s \in \mathcal{S}$ is selected. Here, additional parameters in the master problem are given as follows:

- $\varrho_{v,s} \in \{0,1\}$: 1 if the route s belonging to a vehicle visits the vertex v; 0 otherwise.
- ω_s: total travel fares paid by passengers and parcel owners on this route s, including basic travel fares without a detour for passengers, discounted travel fares with a detour for passengers and original travel fares for parcel owners.

The SARP-PRDL model can be rewritten as the following set packing model.

$$\max \quad \sum_{s\in\mathcal{S}} \omega_s \vartheta_s \tag{4.46}$$

subject to:

$$\sum_{s\in\mathcal{S}} \varrho_{i,s}\vartheta_s \leq 1, \forall i \in \mathcal{M} \tag{4.47}$$

$$\sum_{s\in\mathcal{S}} \varrho_{k^+,s}\vartheta_s = 1, \forall k \in \mathcal{K} \tag{4.48}$$

$$\sum_{s\in\mathcal{S}} \varrho_{k^-,s}\vartheta_s = 1, \forall k \in \mathcal{K} \tag{4.49}$$

$$\vartheta_s \in \{0,1\}, \forall s \in \mathcal{S} \tag{4.50}$$

The objective function (4.46) maximizes the total travel fares. Constraint (4.47) requires that each request is served at most one route. Constraints (4.48) and (4.49) ensures that each vehicle serves a route from its own starting point and dummy point. Constraint (4.50) defines the domain of decision variables.

4.2.4. Restricted master problem

The number of feasible columns is exponential, making it impractical to enumerate all possible columns when the set of vertices is extensive. Thus, focusing on the subset $\mathcal{S}' \subset \mathcal{S}$ proves to be an effective approach for solving the restricted master problems iteratively. The restricted master problem (RMP) can be formulated as follows:

$$\max \quad \sum_{s\in\mathcal{S}'} \omega_s \vartheta_s \tag{4.51}$$

$$\sum_{s\in\mathcal{S}'} \varrho_{i,s}\vartheta_s \leq 1, \forall i \in \mathcal{M} \quad \cdots\cdots \quad \pi_i \tag{4.52}$$

$$\sum_{s\in\mathcal{S}'} \varrho_{k^+,s}\vartheta_s = 1, \forall k \in \mathcal{K} \quad \cdots\cdots \quad \lambda_{k^+} \tag{4.53}$$

$$\sum_{s\in\mathcal{S}'} \varrho_{k^-,s}\vartheta_s = 1, \forall k \in \mathcal{K} \quad \cdots\cdots \quad \lambda_{k^-} \tag{4.54}$$

$$0 \leq \vartheta_s \leq 1, \forall s \in \mathcal{S}', \tag{4.55}$$

where the decision variable ϑ_s is relaxed to a continuous variable between 0 and 1.

4.3. Dynamic mathematical model

After a set of routes serving static passenger and parcel requests have been scheduled, the vehicle fleet will start service at the beginning of working hours. For those unmatched static requests, we do not consider scheduling them in the dynamic request scheduling because these passengers and shippers/recipients may change their travel or delivery plan. Generally, these pre-designed routes are expected to remain unchanged during working hours, and drivers are required to follow the planned routes. However, in practice, new requests may be submitted to the service platform while the vehicle fleet is already en-route. If feasible, these dynamic requests can be accommodated by the existing fleet, provided that suitable vehicles are available to service them. Consequently, modifying the planned routes of vehicles for static requests can enhance the platform's revenue by accommodating new requests.

4.3.1. Constructing and processing the extended subgraph

We divide the whole time period into multiple discrete time cycles, and an extended subgraph is developed to identify potential insertion for dynamically submitted passenger and parcel requests in each cycle.

Definition 4.2 (Dynamic requests)**.** *All passenger and parcel requests submitted during the vehicle service time $T = [\underline{t}, \overline{t}]$ are classified as dynamic requests.*

In time cycle $t \subset T$, let N_{kt}^f be the static requests' vertices set that vehicle k has visited in this cycle. Let E_{kt}^f be the edge set that vehicle k has traversed or is traversing. Let $\mathcal{N}_t$ be the vertices set of dynamic requests in the time cycle t. If we denote $\mathcal{R}_t^o, \mathcal{R}_t^d, \mathcal{C}_t^o, \mathcal{C}_t^d$ as the set of dynamic passenger requests' origins, the set of dynamic passenger requests' destinations, the set of dynamic parcel requests' pick-up locations, and the set of dynamic parcel requests' roaming delivery locations, then the vertices set of dynamic requests satisfies $\mathcal{N}_t = \mathcal{R}_t^o \cup \mathcal{R}_t^d \cup \mathcal{C}_t^o \cup \mathcal{C}_t^d$. Let $\mathcal{E}_t = \{(i,j) | \forall i \in \mathcal{N}_t, j \in N, i \neq j\}$ be the edges set containing edges between dynamic requests' vertices in $\mathcal{N}_t$ and existed vertices in N.

In the constraint (4.47), not all static requests can be served by a fixed vehicle fleet. When dynamic requests begin to enter the service platform, we do not consider vertices and edges corresponding to unmatched static requests in each time cycle. Let $\mathcal{N}_0^u$ be the vertices set of unmatched static requests. Then an extended subgraph (N_t, E_t) in the time cycle t can be created as follows:

$$N_t = (N_{t-1} \setminus \bigcup_{k\in\mathcal{K}} N_{kt}^f) \cup \mathcal{N}_t, \quad t = 1, 2, ..., n \tag{4.56}$$

$$E_t = (E_{t-1} \setminus \bigcup_{k \in \mathcal{K}} E_{kt}^f) \cup \mathcal{E}_t, \quad t = 1, 2, ..., n \tag{4.57}$$

$$N_0 = N \setminus \mathcal{N}_0^u, \quad E_0 = E \setminus \{(i,j) | \forall i \in \mathcal{N}_0^u, j \in N, i \neq j\} \tag{4.58}$$

We may still use the simple example shown in Figure 3.1 to describe the process of generating extended subgraph. Figure 4.2 on the left depicts current locations of two service vehicles at the time point t. On this moment, driver 1 and 2 are heading to pick up passenger 1 and 3 respectively. Meanwhile, a dynamic parcel request "$G2$" enters in the platform at the time point t, waiting for being assigned to a potential vehicle. Figure 4.2 on the right depicts an extended subgraph after "$G2$" enters. For simplicity, the figure just depicts partial edges while the actual number of edges is much higher in this extended subgraph. Additionally, the vertex "$G1$" and edge "$(G1, P1)$" are also eliminated as the vehicle 1 has visited this vertex and edge.

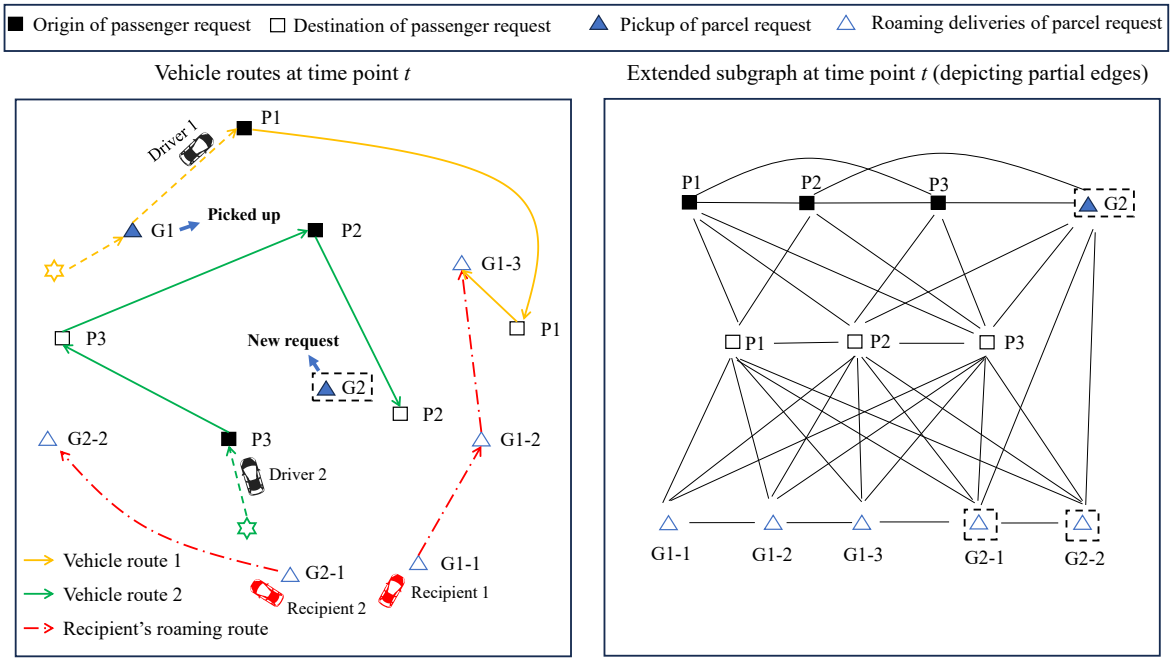


Figure 4.2: Left: Vehicle routes and current locations at time point t; **Right**: Extended subgraph with the introduction of dynamic parcel request $G2$.

4.3.2. Formulating periodic RMP

In each time cycle t, we need to develop a new master problem for dynamic requests. Obviously, we can use each route $\overline{s}$ for which $\vartheta_{\overline{s}} = 1$ in the previous problem, and reoptimize the as-yet unexecuted parts of routes to find potential insertion for dynamic requests. Let $\overline{s}_f$ be the unexecuted part of route $\overline{s}$, $\omega_{\overline{s}_f}$ be the travel fare of route $\overline{s}_f$, then we have $\omega_{\overline{s}_f} \leq \omega_{\overline{s}}$. In this new set of master problems, we focus on the subset $\mathcal{S}_f'$ consisting of routes that vehicles are about to execute. Then, the RMP in the time cycle t can be formulated as follows:

$$\max \quad \sum_{s_f \in \mathcal{S}_f'} \omega_{s_f} \vartheta_{s_f} \tag{4.59}$$

$$\sum_{s_f \in \mathcal{S}_f'} \varrho_{i,s_f} \vartheta_{s_f} = 1, \quad \forall i \in N \setminus \mathcal{N}_0^u \cup \left\{k^- | \forall k \in \mathcal{K}\right\} \tag{4.60}$$

$$\sum_{s_f \in \mathcal{S}_f'} \varrho_{i,s_f} \vartheta_{s_f} \leq 1, \quad \forall i \in \mathcal{N}_t \tag{4.61}$$

$$0 \leq \vartheta_{s_f} \leq 1, \quad \forall s_f \in \mathcal{S}_f' \tag{4.62}$$

Constraints (4.60) guarantee all static requests matched successfully should be served by vehicles, and vehicles must visit their dummy ending points. Constraints (4.61) require that each dynamic request is served no more than once.

5. Solution approach

As mentioned in Section 4, the SARPRDL could be converted into an equivalent MILP by linearization technique though it was originally formulated as an MINLP. The MILP could be easily solved by off-the-shelf solvers (e.g., Gurobi) in the small-scale instances. However, using the off-the-shelf solvers for instances with medium number of passengers and parcel requests could be time-consuming. Thats mainly due to exponential number of integer variables and big-M constraints. To obtain a high-quality solution for static and dynamic problems efficiently, we design a solution approach based on branch-and-price framework. Meanwhile, the adaptive large neighborhood search algorithm is used to evaluate our test results in the static instances.

5.1. The overview of branch-and-price matheuristic approach

The branch-and-price matheuristic approach employs the solution framework of the branch-and-price algorithm. The distinction lies in the methodology employed to address the pricing subproblem. In our approach, exact solutions are not a prerequisite in both static and dynamic path-based models, given the potential for vehicle routes to be rescheduled in accordance with time cycles that are continuously rolling. Consequently, we have opted to use a heuristic pricing strategy to resolve the subproblem until the termination procedure of column generation in the subproblem is activated. The flow chart of our BPM algorithm is illustrated in Figure 5.1.

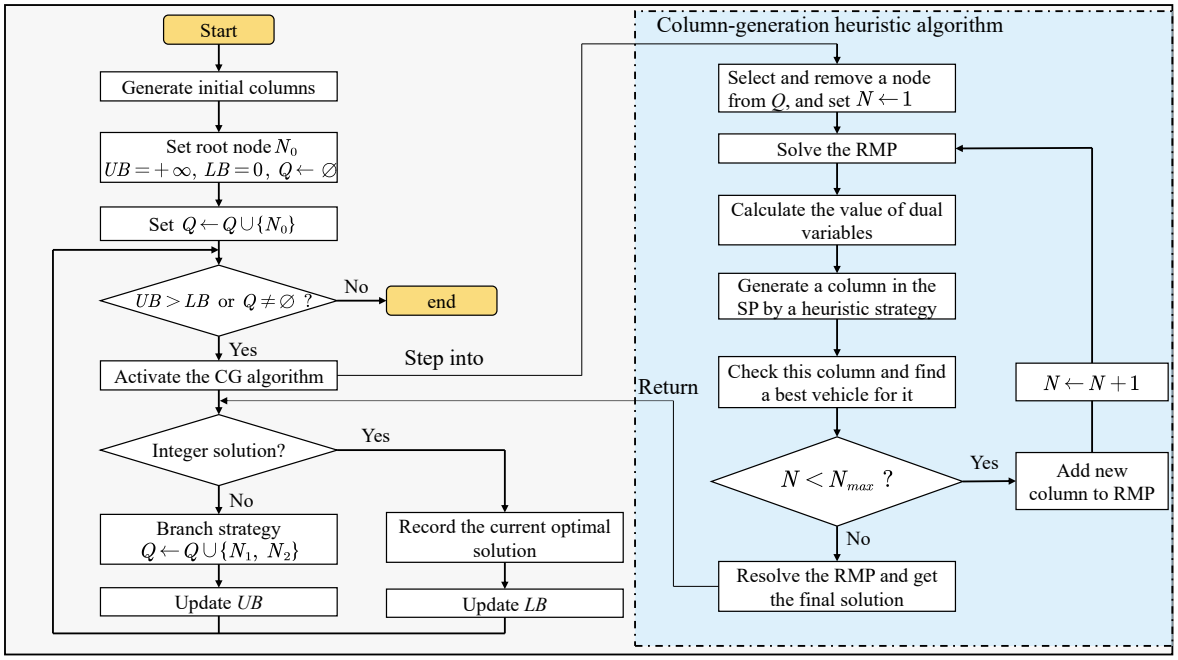


Figure 5.1: Flow chart of the BPM approach.

5.2. Pricing subproblem

In the pricing subproblem, we calculate the reduced cost of each route generated by pricing strategy. Let π_i, λ_{k^+}, λ_{k^-} denote dual variables corresponding to constraints (4.52), (4.53) and

(4.54) of static RMP. Then, for a vehicle k, the reduced cost of the route s can be calculated as follows.

$$\tilde{\omega}_{sk} = \omega_s - \sum_{i \in \mathcal{M}} \varrho_{i,s}\pi_i - \varrho_{k^+,s}\lambda_{k^+} - \varrho_{k^-,s}\lambda_{k^-} \tag{5.1}$$

To find a promising route, the connection between the pricing subproblem and arc-based model is needed. For each vehicle, we let $x_{i,j}$ denote whether the arc (i, j) is traversed by this vehicle. The route s's travel fare ω_s can be expressed using the variable $x_{i,j}$.

$$\omega_s = \sum_{r \in \mathcal{R}} \sum_{v \in \mathcal{M}} \delta_{r^+} x_{r^+,v} + \sum_{c \in \mathcal{C}} \sum_{m \in \mathcal{M}_{c^-}} \sum_{v \in \mathcal{M} \setminus \mathcal{M}_{c^-}} \delta_{c^-} x_{v,m} \tag{5.2}$$

Here, $\delta_{r^+} = \beta_R L_r(y_{r^+} + \gamma(1 - y_{r^+}))$, $\delta_{c^-} = \beta_C L_{cm}$. Based on the information above, we aim to find a column with the most positive reduced cost $(s^*, k^*) = \arg\max\{\tilde{\omega}_{sk} | \tilde{\omega}_{sk} > 0, k \in \mathcal{K}\}$, which is conducive to improving the objective value in the static RMP model.

As for the periodic RMP, we also calculate the reduced cost based on the route s_f's travel fare and dual variables.

$$\tilde{\omega}_{s_f k} = \omega_{s_f} - \sum_{i \in (N \setminus \mathcal{N}_0^u) \cup \mathcal{N}_t} \varrho_{i,s_f}\pi_i - \varrho_{k^-,s_f}\lambda_{k^-} \tag{5.3}$$

5.3. Heuristic pricing approach for subproblem

By the route costs and dual values obtained from the restricted master problem, we can generate a promising column in the pricing problem for the master problem. In the static problem, an entire route from vehicle's initial location to its dummy ending point is generated, whereas a subroute from a vehicles current location to vehicle's dummy ending point is generated in the dynamic problem.

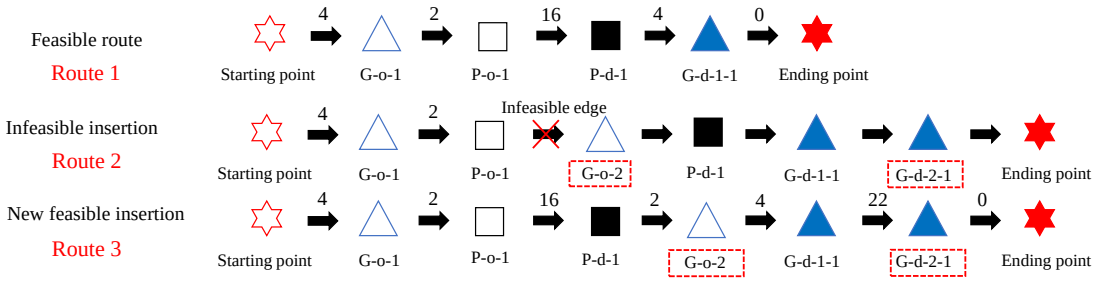
(1) Column generation in static model

In the static model (4.51), we generate a promising column with maximum positive reduced cost to increase the objective value. Each column should be an complete route from a starting point to an ending point. In each loop, we decide to insert a request with the maximum difference between the travel fare of request and the sum of dual variables of request. Additionally, the feasibility of each generated column must be verified by the proposed algorithm.

We use a simple example depicted in Figure 5.2 to illustrate the insertion process. The vertex "P(G)-o-x" denotes the vertex of origin of the passenger (parcel) request "x". Similarly, the designation "P-d-x" signifies the destination of the passenger request "x", while "G-d-x-1" denotes the first candidate delivery location of the parcel request "x". It is assumed that the service times for the passenger and parcel vertices are zero and one minute, respectively. The entire time span is given by the interval $[0, 70]$. Route 1 is a provisional route derived by the Algorithm 2 following the insertion of two requests, with the numbers indicating travel times between vertices. A further request insertion will be explored based on this route. Before inserting a new request, the total travel time is 38 minutes. It is possible to incorporate a new request into this route without violating time constraint. Parcel request 2 is selected for insertion into route 1. However, route 2 is deemed infeasible due to the infeasible edge ("P-o-1", "G-o-2") that prohibits the vehicle from picking up a parcel when a passenger is onboard. Nevertheless, modifying the sequence of "G-o-2" and "P-d-1" generates Route 3, a new feasible route. Furthermore, the time constraints can be met post-insertion.

(2) Column generation in dynamic model

When it comes to insert dynamic requests into off-the-shelf vehicle routes based on the static solution, we can only reoptimize the as-yet unexecuted parts of vehicle routes. Take the route 1 in Figure 5.2 as example, if vehicle is traversing the edge ("G-o-1", "P-o-1"), then we will only reoptimize the partial route after vertex "P-O-1" under the premise of not losing



Node	Time window	Arrival time	Departure time
Starting point	[0, 70]	0	0
G-o-1	[0, 70]	4	5
P-o-1	[3, 13]	7	7
P-d-1	[13, 41]	33	33
G-d-1-1	[24, 40]	37	38
Ending point	[0, 70]	38	38

Node	Time window	Arrival time	Departure time
Starting point	[0, 70]	0	0
G-o-1	[0, 70]	4	5
P-o-1	[3, 13]	7	7
P-d-1	[13, 41]	33	33
G-o-2	[0, 70]	35	36
G-d-1-1	[24, 40]	40	41
G-d-2-1	[46, 70]	63	64
Ending point	[0, 70]	64	64

Feasible route 1 ⟶ New feasible route 3

Figure 5.2: An illustrative example when inserting a new request.

vertex "G-d-1-1". Therefore, we generate a new route starting from vertex "P-o-1", and this new partial route can be combined with the executed part to form a new entire route.

We can summarize the process of heuristic pricing approach in Algorithm 1.

Algorithm 1: Generating a promising column in subproblem

Input: RMP model, set of passenger requests $\mathcal{R}$, set of parcel requests $\mathcal{C}$, set of origins of passenger requests $\mathcal{R}^o$, set of destinations of passenger requests $\mathcal{R}^d$, set of origins of parcel requests $\mathcal{C}^o$, set of roaming deliveries of parcel requests $\mathcal{C}^d$

Output: A column with maximum reduced cost $\tilde{\omega}_s$

Solve the RMP model and obtain the dual variables;

if *an entire route should be generated in the static problem* **then**

  Initialize an empty route s;

  $\tilde{\omega}_s$, $s = HeuristicPricing(s, \mathcal{R}, \mathcal{C}, \mathcal{R}^o, \mathcal{R}^d, \mathcal{C}^o, \mathcal{C}^d)$;

if *a unexecuted route should be generated in the dynamic problem* **then**

  Let $s_e = [i_1, i_2, ..., i_k]$ be a sequence of vertices, where i_k is the vertex towards which the vehicle is heading;

  Initialize $s \leftarrow [i_{k+1}]$;

  if *there is a pick-up vertex i_p in the sequence s_e but corresponding drop-off vertex i_d is not included in s_e* **then**

    Add vertex i_d into s;

  $\tilde{\omega}_s$, $s = HeuristicPricing(s, \mathcal{R}, \mathcal{C}, \mathcal{R}^o, \mathcal{R}^d, \mathcal{C}^o, \mathcal{C}^d)$;

  Merging s_e and s to form a complete route;

5.4. Branching strategy

Each time the CG algorithm is activated, a linear relaxation of the RMP model is solved, which may result in a non-integer solution. To achieve an integer solution, it is necessary to develop a branch-and-bound tree. Our branching strategy employs a rule that branches on the arc flow variables associated with the graph G. Upon discovering a fractional solution at a node of the branch-and-bound tree, two child nodes may be generated from it. Regarding the search rule in the branch-and-bound tree, we utilize best-first search to efficiently explore and prune branches that can be cut off.

Algorithm 2: *HeuristicPricing*$(s, \mathcal{R}, \mathcal{C}, \mathcal{R}^o, \mathcal{R}^d, \mathcal{C}^o, \mathcal{C}^d)$

Initialize maximum reduced cost of route $\tilde{\omega}_{max} \leftarrow 0$;
while ***True*** **do**
  Initialize the best request for insertion $\xi \leftarrow None$, the best objective increase $\Delta^{obj}_{max} \leftarrow 0$, the best record of route and reduced cost $(s_{best}, \tilde{\omega}_{best}) \leftarrow (None, None)$;
  Calculate route s's reduced cost $\tilde{\omega}_s$;
  for $\xi \in \mathcal{R} \cup \mathcal{C}$ **do**
    Let new route's reduced cost $\tilde{\omega}^*_s = 0$, the set of feasible routes $F = \emptyset$;
    if ξ *is a passenger request* **then**
      Find the origin $\xi^+ \in \mathcal{R}^o$, and the destination $\xi^- \in \mathcal{R}^d$;
      for $i_p \in \{1, 2, ..., |s|\}$ **do**
        Insert ξ^+ into route s at the index i_p, generating a new route s_1;
        if *Time, capacity and infeasible edge constraints satisfy* **then**
          for $i_d \in \{i_p + 1, i_p + 2, ..., |s| + 1\}$ **do**
            Insert ξ^- into s_1 at the index i_d, generating a new route s_2;
            if *Time, capacity and infeasible edge constraints satisfy* **then**
              Calculate the reduced cost of route s_2; $F \leftarrow F \cup \{s_2\}$;
    if ξ *is a parcel request* **then**
      Find the pickup $\xi^+ \in \mathcal{C}^o$, and roaming deliveries $\mathcal{M}_{\xi^-} \subset \mathcal{C}^d$;
      for $i_p \in \{1, 2, ..., |s|\}$ **do**
        Insert ξ^+ into route s at the index i_p, generating a new route s_1;
        if *Time, capacity and infeasible edge constraints satisfy* **then**
          for $i_d \in \{i_p + 1, i_p + 2, ..., |s| + 1\}$ **do**
            for $m \in \mathcal{M}_{\xi^-}$ **do**
              Insert ξ^- into s_1 at the index i_d, generating a new route s_2;
              if *Time, capacity and infeasible edge constraints satisfy* **then**
                Calculate the reduced cost of route s_2; $F \leftarrow F \cup \{s_2\}$;
    Find all feasible routes in F and obtain a route s^* with maximum reduced cost $\tilde{\omega}^*_s = \max\{\tilde{\omega}_{\bar{s}} | \tilde{\omega}_{\bar{s}} > 0, \bar{s} \in F\}$;
    Calculate objective change $\Delta_{s^*} = \tilde{\omega}_{s^*} - \tilde{\omega}_s$;
    if $\Delta_{s^*} > \Delta^{obj}_{max}$ **then**
      $\xi_{best} \leftarrow \xi$; $\Delta^{obj}_{max} \leftarrow \Delta_{s^*}$; $(s_{best}, \tilde{\omega}_{best}) \leftarrow (s^*, \tilde{\omega}^*_s)$;
  if ξ_{best} *is not None* **then**
    if $\xi \in \mathcal{R}$ **then**
      $\mathcal{R} \leftarrow \mathcal{R} \backslash \{\xi_{best}\}$;
    if $\xi \in \mathcal{C}$ **then**
      $\mathcal{C} \leftarrow \mathcal{C} \backslash \{\xi_{best}\}$;
    $\tilde{\omega}_{max} \leftarrow \tilde{\omega}_{best}$; $s \leftarrow s_{best}$;
  else
    Break the loop;
Return $\tilde{\omega}_{max}, s$

Initially, for the node with a fractional solution, we need to compute the values of each arc flow variables according to the following equation:

$$x_{i,j} = \sum_{s \in \mathcal{S}', (i,j) \in s} \overline{\vartheta}_s \tag{5.4}$$

where $\overline{\vartheta}_s$ is the linear relaxation solution of the RMP.

Subsequently, we choose a branching arc based on the arc variable closest to 0.5, as determined by equation $(i_q, i_{q+1}) = \arg\min_{(i,j) \in E}\{|x_{i,j} - 0.5|\}$, and generate two child nodes, referred to as the left and right child nodes, respectively. We impose the branching constraint (5.5) to left child node and the branching constraint (5.6) to right child node.

$$\sum_{s \in \mathcal{S}', (i_q, i_{q+1}) \in s} \overline{\vartheta}_s = 0 \tag{5.5}$$

$$\sum_{s \in \mathcal{S}', (i_q, i_{q+1}) \in s} \overline{\vartheta}_s = 1 \tag{5.6}$$

Consequently, the left child node will prohibit the route traversing the arc (i_q, i_{q+1}), while another node must enforce at least one route passing through the arc (i_q, i_{q+1}).

6. Computational experiments

To evaluate the performance of our proposed solution approach, we conduct computational experiments based on a real-world case study. All the computational results reported below are based on a Microsoft Windows 10 with Intel Core i9 - 3.60GHz and 16GB RAM, using Python 3.10 and Gurobi 10.0.0. The maximum running time of the Gurobi is 3600s, and We also use ALNS algorithm (see Appendix A) to evaluate our proposed matheuristic. The maximum number of iterations of the ALNS algorithm is 2000. Furthermore, we let ALNS algorithm terminate if the best solution remains constant after 300 iterations. The values of critical parameters in the mathematical model and the ALNS algorithm are listed in Table 3 and 4.

Table 3: Values of main input parameters in the model

Parameter	Value
Travel fare per kilometer for passengers β_R	2.5
Travel fare per kilometer for parcels β_C	2
Fare discount coefficient for passengers when taking a detour γ	0.8
Average speed of vehicles V (km/h)	30
Service time for passengers/parcels at any vertex g (minute)	0/1
Number of passenger seats in the vehicle	4
Maximum size for pieces of luggage or parcels in the vehicle	10

Table 4: Values of main parameters in the ALNS algorithm

Parameter	Value
Maximum number of iterations	2000
The ratio of requests removed	0.2
Initial temperature	100
Lowest temperature	10
Linear cooling rate	0.01
Score of the operators	[40, 20, 5, 1]

6.1. Experiment settings

We use a partial road network of Chengdu city, China, for numerical experiment of the proposed model. In this road network, a taxi company offers both passenger delivery services and expedited parcel delivery. The passenger/parcel delivery data are extracted from real taxi trip data records collected during the peak hour of the study area, which spans from 8:00 to 9:00 on 1 August 2021.

To establish a baseline for the experimental study, the proportion of passenger requests is initially set at 75%, with parcel requests comprising the remaining 25%. Approximately 50% of the parcel requests have a fixed delivery location, while the other 50% require delivery according to the roaming routes of recipients, which are generated randomly within a one-hour period. The initial locations of the vehicles are uniformly distributed across the road network. The generated dataset is then divided into two distinct subsets: a static requests set and a dynamic requests set. Following the approach outlined by Lund et al. (1996), we introduce the concept of the *degree of dynamism* (DoD) to differentiate between these sets. The DoD is defined as $\delta = \frac{n_d}{n_s+n_d} \in [0,1]$, where the dynamism of an instance increases with the number of dynamic requests n_d and decrease with the number of static requests n_s. The initial value of the DoD is set to approximately 50%. In other words, the number of dynamic requests is approximately equal to that of static requests.

The time span for the experiment is set to one hour, covering the interval$[0, 60]$. The duration of each time cycle is set to 5 minutes, resulting in 12 time cycles in total. Each time cycle lasts 5 minutes, resulting in a total of 12 time cycles. In our experimental study, we randomly generate the submission times for dynamic parcel requests and dynamic passenger requests within the intervals $[0, 20]$ and $[0, 40]$, respectively. The requirement for parcel requests to be submitted at least 20 minutes in advance serves two purposes: (i) the time sensitivity of parcel pickups and deliveries is lower than that of passenger requests, and (ii) vehicles cannot perform parcel pick-up and delivery services while servicing passengers. By submitting parcel requests early, vehicles can collect more packages before picking up passengers, thereby allowing them to fulfill a greater number of parcel delivery requests. After generating the submission times, we will define the time windows and latest arrival times for each dynamic request.

6.2. The test performance on static requests scheduling

6.2.1. Small-scale test performance

Six small-sized instances were generated for testing the proposed matheuristic algorithm. Table 5 shows the results obtained by Gurobi, ALNS and the BP matheuristic, respectively. We denote $\mathcal{N}_r$ as the number of static requests, which includes both passenger and parcel requests. The set of vehicle fleets, denoted by VT, contains various numbers of vehicles. Specifically, we let the lengths of sets VT_1^s, VT_2^s and VT_3^s to be 5, 8 and 10, respectively. For each fixed instance and vehicle fleet, we run each heuristic algorithm five times and record the best and average results. To evaluate the performance of the heuristic algorithms, we calculate $\Delta_{gap} = \frac{|\Pi^{best} - \Pi^*|}{\Pi^*}$, which measures the gap between the approximate solution from the heuristic algorithm and the incumbent solution from Gurobi.

Table 5 shows that Gurobi is only able to efficiently solve instances within a limited range. The ALNS algorithm effectively reduces the total computation time for test instances that Gurobi could hardly solve. Furthermore, the proposed matheuristic algorithm can quickly solve all static instances, yielding high-quality solutions. When applying the matheuristic, approximately 70 percent of instances achieve the same best solution as Gurobi, and each instance is solved faster than with the ALNS algorithm. As the number of static requests increases to 20, Gurobi can find a dominant incumbent solution through time-consuming computation. In contrast, our matheuristic rapidly obtains an approximate optimal solution with a minor gap of $\Delta_{gap} < 3\%$. For Instance 4 with the vehicle fleet VT_1^s, the matheuristic produces a solution with a gap of $\Delta_{gap} = 3.4\%$, which is consistent with the result calculated

in Instance 3 with the vehicle fleet VT_1^s. This local optimality may arise when BP matheuristic searches for promising columns in the pricing subproblem.

Table 5: Computational results in small-scale instances

Instance	$\mathcal{N}_r$	VT	Gurobi			ALNS				BP matheuristic			
			Π^*	time(s)	Gap	Π_1^{best}	Π_1^{avg}	time(s)	Δ_{gap}^1	Π_2^{best}	Π_2^{avg}	time(s)	Δ_{gap}^2
1	10	VT_1^s	23.97	0.43	0	23.97	23.97	6.24	0	23.97	23.97	1.09	0
		VT_2^s	34.49	0.76	0	34.49	34.49	5.17	0	34.49	33.46	1.97	0
		VT_3^s	37.14	1.09	0	37.14	37.14	5.31	0	37.14	36.45	1.79	0
2	12	VT_1^s	26.86	4.18	0	26.86	26.86	10.15	0	26.86	26.86	1.29	0
		VT_2^s	39.16	5.01	0	39.16	39.16	8.48	0	39.16	38.78	3.86	0
		VT_3^s	41.93	2.14	0	41.93	41.93	8.41	0	41.93	41.9	3.14	0
3	14	VT_1^s	29.41	9.16	0	28.92	28.92	11.09	1.7	28.92	28.92	3.76	1.7
		VT_2^s	47.1	42.7	0	47.1	45.78	12.25	0	47.1	46.85	5.28	0
		VT_3^s	53.26	20.73	0	53.26	52.97	9.64	0	53.26	52.60	6.11	0
4	16	VT_1^s	29.93	26.74	0	29.82	29.14	22.87	0.4	28.92	28.92	5.14	3.4
		VT_2^s	50.67	1293.42	0	50.53	49.92	30.51	0.3	50.67	50.44	8.19	0
		VT_3^s	58.57	>3600	0.2	58.57	58.28	30.46	0	58.57	58.07	6.56	0
5	18	VT_1^s	33.28	730.53	0	33.28	32.2	49.61	0	33.28	32.4	8.02	0
		VT_2^s	56.35	>3600	12.5	56.35	55.95	43.34	0	56.35	55.87	15.68	0
		VT_3^s	65.54	>3600	2.7	65.54	62.72	33.98	0	65.54	65.04	13.59	0
6	20	VT_1^s	35.69	879.87	0	34.58	33.60	96.73	3.1	34.81	34.52	11.77	2.5
		VT_2^s	61.35	>3600	17.7	60.25	59.51	91.34	1.8	60.54	59.95	16.75	1.3
		VT_3^s	71.75	>3600	4.4	69.8	68.7	96.36	2.7	70.94	69.29	16.33	1.1

6.2.2. Large-scale test performance

In the context of large-scale test instances, the Gurobi solver is no longer employed for calculating solutions, due to its inefficiency to find the optimum. The number of static requests varies from 30 to 60, with vehicle fleets VT_1^l of 20 vehicles and VT_2^l of 40 vehicles employed to serve passenger and parcel requests. For each instance, we conduct at least 5 repeated computations and record the best objective value and the average value. The experimental results are summarized in Table 6. It is observed that the computation time increases significantly as the number of requests grows. Additionally, the data results demonstrate that our proposed BP matheuristic algorithm achieves better solutions with reduced computing time. Notably, increasing the number of vehicles improves the objective value and shortens computing time. This enhancement occurs because new promising columns can be generated in the pricing subproblem, and thereby accelerating convergence. However, the use of the ALNS algorithm to solve instances is exceedingly time-consuming. This may be attributed to the large number of vertices generated by requests and the substantial time spent on removal and repair operators in each iteration.

6.3. Experimental study on dynamic requests scheduling

Once the final result of static requests scheduling has been determined, dynamic requests can be scheduled based on the as-yet unexecuted parts of vehicle routes. In the context of dynamic requests scheduling, two schemes are proposed for scheduling dynamic requests. Scheme I involves re-optimizing vehicle routes for dynamic requests, as detailed in Section 4.3. In contrast, Scheme II adopts a greedy heuristic approach, focusing on inserting as many requests as possible without replanning the existing routes (see Appendix B).

Table 7 presents the computational results for scheduling dynamic requests using both Scheme I and Scheme II. In this table, we let m be the number of total requests, including

Table 6: Computational results in large-scale instances

Instance	$\mathcal{N}_r$	VT	ALNS			BP matheuristic		
			Π_1^{best}	Π_1^{avg}	time(s)	Π_2^{best}	Π_2^{avg}	time(s)
1	30	VT_1^l	118.92	117.27	906.2	118.92	117.58	171.7
		VT_2^l	120.3	119.92	805.1	120.3	119.99	63.1
2	40	VT_1^l	156.03	154.29	5024.6	156.87	154.15	2821.8
		VT_2^l	162.41	162.22	2527.6	162.41	162.17	241.6
3	50	VT_1^l	172.83	168.44	14902.5	176.02	173.79	8797.8
		VT_2^l	193.77	193.66	11807.2	194.25	193.9	1443.1
4	60	VT_1^l	189.99	183.4	35125.4	195.26	192.51	8852.6
		VT_2^l	232.12	231.14	23261.2	232.6	232.36	5038.2

both static and dynamic requests; n_s be the number of static requests; n_d be the number of dynamic requests; N_{VT} be the number of vehicles in the fleet. For static request experiments, the table reports the objective value Π_1, the completion rate of static requests σ_s and the number of vehicles not in use $\widetilde{n}_s$. In dynamic requests experiments, the table reports the objective value, denoted as Π_2^w for Scheme I and Π_2^{wo})for Scheme II, and the completion rate of dynamic requests σ_d. The values Π_2^w and Π_2^{wo}) are calculated as the sum of Π_1 and the profit generated from dynamic requests.

Table 7 shows that when the vehicle fleet has limited time and capacity available after scheduling static requests, only a small portion of dynamic requests can be accommodated. In such cases, Scheme I generally achieves higher total revenue and completion rates than Scheme II. For example, when m=20 or 40 and $N_{VT} = 10$, Scheme I generates higher total revenue while maintaining the same completion rates for dynamic requests as Scheme II. When m=60 or 80 and $N_{VT} = 20$, Scheme I continues to demonstrate slight improvements in both total revenue and dynamic request completion rates compared to Scheme II. This can be attributed to two main factors: (i) Scheme I utilizes route reoptimization to reassign certain requests to other vehicles, thereby freeing up service time for high-revenue dynamic requests; and (ii) Scheme II employs a greedy insertion strategy that, although it accommodates requests in earlier iterations, may overlook opportunities to serve high-revenue dynamic orders in later iterations.

However, the advantages of Scheme I diminish when there are a large number of unused vehicles in the fleet after scheduling static requests. In these situations, the route reoptimization strategy of Scheme I becomes less effective compared to the greedy insertion strategy of Scheme II. For example, when m=20 and N_{VT}=20, or m=80 and N_{VT}=30, both schemes yield identical results. When m=60 and N_{VT}=30, Scheme I results in lower completion rate compared to Scheme II. There are two main reasons for this phenomenon, (i) the unused vehicles, instead of occupied vehicle, are assigned to serve the dynamic requests, and (ii) most of the vehicle routes still have sufficient time available to insert dynamic requests. Both decrease the importance of reoptimizing the routes for occupied vehicles. These factors reduce the importance of reoptimizing routes for occupied vehicles. Instead, the reoptimization strategy tends to continuously reorder unexecuted requests to free up service time for high-revenue requests, which can lead to longer travel distances and less flexible time for inserting new requests. While Scheme II may complete more tasks, the revenue generated may not be comparable. For instance, with m=60 and N_{VT}=30 for example, Scheme II completes 13.8% more dynamic requests than Scheme I, but the total revenue only increases by $\frac{174.46-171.24}{171.24} \approx 1.9\%$. This suggests that Scheme II often inserts a larger number of low-revenue dynamic requests.

Table 7: Computational results with **(Scheme I)** and **(Scheme II)**

m	n_s	n_d	N_{VT}	Static result			Scheme I		Scheme II	
				Π_1	σ_s	$\widetilde{n}_s$	Π_2^w	σ_d	Π_2^{wo}	σ_d
20	11	9	10	39.26	90.9%	0	43.66	22.2%	42.54	22.2%
			20	42.86	100%	11	74.66	100%	74.66	100%
40	20	20	10	71.79	80%	0	74.23	5%	72.91	5%
			20	83.05	100%	4	113.25	40%	111.97	40%
60	31	29	20	117.88	100%	0	137.36	17.2%	128.06	13.8%
			30	119.93	100%	9	171.24	37.9%	174.46	51.7%
80	40	40	20	156.87	100%	0	169.09	10%	161.11	5%
			30	161.4	100%	1	196.0	25%	196.0	25%
100	51	49	20	177.21	92.2%	0	182.73	4.1%	182.73	4.1%
			30	192.4	100%	0	223.22	20.4%	214.95	16.3%

6.4. The effect of varying DoD on revenue

In the preceding experiments, the DoD is set to $\delta \approx 50\%$, resulting in an approximately equal number of static and dynamic requests. We may examine the effect on revenue under the conditions that $\delta \approx 30\%$ and $\delta \approx 70\%$. We use the instances with requests varying from 20 to 100, while keeping the total number of vehicles fixed at 20. When $\delta \approx 50\%$, the computational results may refer to Table 7. The overall revenue Π from serving both static and dynamic requests is defined as the maximum of objective values obtained from Scheme I and Scheme II, expressed as $\Pi = \max\{\Pi_2^w, \Pi_2^{wo}\}$.

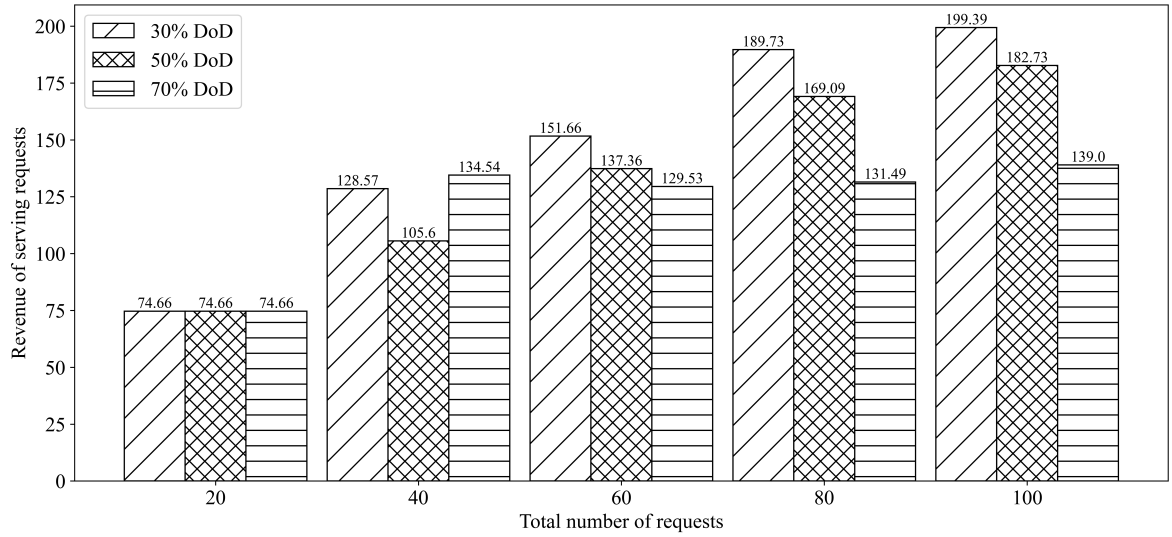


Figure 6.1: The total revenue of serving requests under different degrees of dynamism.

Figure 6.1 compares the revenue generated from servicing passenger and parcel requests under various of DoDs. Overall, when the DoD is at 30% or 50%, the revenue shows a steadily increasing trend, whereas at 70%, the revenue initially increases but eventually stabilizes. When the total number of requests is 20, the vehicle supply is sufficient to fulfill all requests, leading to consistent revenue across all levels of dynamism, with $\Pi = 74.66$. At a total request count of 40, the revenue at 70% DoD reaches 134.54, slightly larger than that of 30% DoD and significantly exceeding the revenue at 50% DoD. This is likely because, under a 70% DoD, after scheduling static requests, a larger number of vehicles remain unused, allowing for the servicing of more dynamic requests in subsequent operations. However, when the

total number of requests exceeds 40, vehicle supply falls short of demand. In this scenario, a lower DoD yields higher revenue, while a higher DoD results in lower revenue. This decline in revenue at higher DoD may be attributed to early-inserted dynamic requests consuming available vehicle service time, thereby preventing later-arriving high-revenue dynamic requests from being added to existing routes.

We also recorded the completion numbers for both types of requests, corresponding to the results in Figure 6.1. Figure 6.2 shows that under low DoD, the vehicle fleet primarily serves static requests, resulting in only a small number of dynamic requests being fulfilled. Consequently, prioritizing static requests contributes to higher overall revenue. In contrast, under high DoD, dynamic requests are more likely to be affected by the pre-scheduled routes for static requests. This is especially true when early-assigned low-revenue dynamic requests occupy service capacity, preventing later-arriving high-revenue requests from being accommodated. Therefore, when vehicle supply cannot meet demand, it is advantageous for enhancing revenue to serve static requests if there is a low DoD.

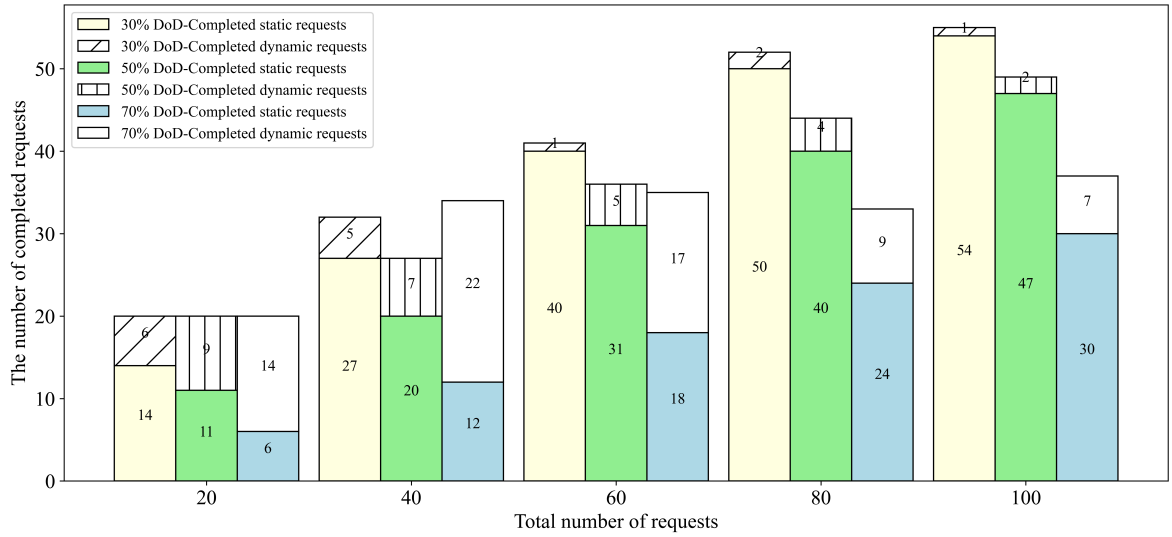


Figure 6.2: The number of completed static and dynamic requests under different DoDs.

6.5. The benefit of using parcels' roaming delivery locations

We now discuss the of incorporating parcels' roaming delivery locations within the share-a-ride operational mode. We continue to use the test instances from Table 7 , and the computational results considering parcels' roaming delivery locations can be obtained directly. In the experimental study without these roaming locations, we eliminated all recipients' roaming routes and randomly selected one stop from the roaming route as the delivery location for each recipient. The experiments show that unsuccessful deliveries may occur if drivers cannot drop off parcels within the specified time windows. However, by accounting for roaming routes, parcels can be delivered to alternative stops if a delivery fails at the mostly preferred location. Table 8 summarizes the computational results for scenarios with and without parcels' roaming delivery locations. Let Π_s be the best revenue of static request scheduling, Π_{sd} be the revenue after finishing dynamic request scheduling, and the revenue of dynamic part can be calculated by $\Pi_d = \Pi_{sd} - \Pi_s$. We use N_c to count the completed requests in the format of "$a + b$" where a and b denote the completed static and dynamic requests, respectively.

Table 8 illustrates the advantages of incorporating roaming delivery locations within the share-a-ride operational mode. Overall, considering roaming locations results in higher revenue, especially during the scheduling of static requests, as more parcel requests can be fulfilled. In contrast, scenarios using only fixed delivery locations often lead to unsuccessful deliveries,

thereby reducing revenue. Interestingly, the total number of completed requests remains similar regardless of roaming delivery locations. For example, with $m = 60$ and $N_{VT} = 20$, the total remains at 36 in both cases. While roaming locations enhance the fulfillment of static requests, their impact on dynamic scheduling is less pronounced. This may be due to the fleet's limited capacity, where available space for dynamic requests is better utilized when roaming delivery options are included.

Table 8: Computational results with considering parcels' roaming delivery locations and ignoring parcels' roaming delivery locations

m	n_s	n_d	N_{VT}	With roaming delivery locations				Without roaming delivery locations			
				Π_s	Π_d	Π_{sd}	N_c	Π_s	Π_d	Π_{sd}	N_c
20	11	9	10	39.26	4.4	43.66	10+2	39.26	4.4	43.66	10+2
			20	42.86	31.8	74.66	11+9	42.86	31.8	74.66	11+9
40	20	20	10	71.79	2.44	74.23	16+1	69.22	1.12	70.34	16+1
			20	83.05	30.2	113.25	20+8	79.01	34.32	113.33	19+10
60	31	29	20	117.88	19.48	137.36	31+5	105.28	28.59	133.87	28+8
			30	119.93	54.53	174.46	31+15	110.41	64.63	175.04	29+16
80	40	40	20	156.87	12.22	169.09	40+4	147.24	4.67	151.91	38+2
			30	161.4	34.6	196.0	40+10	154.12	38.44	192.56	39+10
100	51	49	20	177.21	5.52	182.73	47+2	165.45	6.01	171.46	45+3
			30	192.4	30.82	223.22	51+10	183.24	22.9	206.14	48+8

7. Conclusion

This study addresses the share-a-ride problem with parcels' roaming delivery locations, aiming to optimize total revenue from both passenger and parcel requests. We prioritize passenger service quality by prohibiting parcel pick-ups and drop-offs when passengers are on board. This approach allows parcels to be delivered according to recipients' roaming routes, reducing failed deliveries.

To tackle the SARP-PRDL, we propose an efficient matheuristic solution based on the branch-and-price framework. Our algorithm outperforms the ALNS algorithm in both small-scale and large-scale static request scheduling instances. The results from static scheduling serve as input for the dynamic scheduling process. We evaluate two dynamic schemes: one with vehicle route reoptimization and the other without. The scheme with reoptimization shows slightly better performance when vehicle supply cannot meet demand. However, when there are unused vehicles after static scheduling, the advantage of reoptimization diminishes.

Our numerical study highlights the effectiveness of the proposed method for scheduling static requests. However, its performance in handling dynamic requests is less obvious due to constraints imposed by existing vehicle routes, which limit the number of requests that can be added. Additionally, the solution approach tends to be short-sighted when inserting dynamic requests; early-arriving requests are prioritized, which can prevent later high-revenue requests from being served. This limitation represents a drawback of our study. In the future, we aim to explore algorithms that are better suited for the dynamic request insertion process.

CRediT authorship contribution statement

Shenglin Liu: Conceptualization, Formal analysis, Methodology, Investigation, Data curation, Validation, Writing - original draft. **Qian Ge**: Conceptualization, Formal analysis, Methodology, Validation, Writing- original draft, Writing - review & editing, Funding acquisition. **Ke Han**: Methodology, Writing - original draft, Supervision, Project administration, Funding acquisition. **Lei Yu**: Methodology, Investigation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets that support the findings of this study are available on request.

Acknowledgment

This work is supported by the National Natural Science Foundation of China through grants no. 72101215 and 72071163, the Natural Science Foundation of Sichuan Province through grant no. 2022NSFSC1906, and Technology Innovation and Development Project of Chengdu Science and Technology Bureau through grant no. 2022-YF05-00839-SN.

A. Appendix A: Scheduling static requests by ALNS algorithm

To evaluate the performance of our BP matheuristic algorithm, we apply the adaptive large neighborhood search (ALNS) algorithm when we solve the static problem. Generally, as summarized in Mara et al. (2022), the important parts of ALNS have: (i) an initial solution; (ii) a set of removal operators; (iii) a set of repair operators; (iv) an adaptive mechanism; (v) an acceptance criterion and (vi) an termination criterion.

In each iteration of ALNS, a combination of operators comprising a removal operator and a repair operator is selected using the roulette method. The removal operator eliminates both origins and destinations of certain requests from all routes and adds them to a set of unmatched requests. Subsequently, the repair operator inserts partial origins and destinations of requests chosen from the set of unmatched requests. Upon computing the objective value of the new insertion solution, an acceptance criterion derived from simulated annealing determines whether the new solution replaces the current one. The ALNS algorithm terminates when the specified criterion is satisfied. The framework of the ALNS algorithm for addressing our problem is depicted in Algorithm 3.

Algorithm 3: The framework of ALNS algorithm

Input: An initial solution s_0, the set of removal operators, and the set of repair operators
Output: Best solution s^*
Initialize the best solution $s^* \leftarrow s_0$, the current solution $s \leftarrow s_0$;
while *Termination criterion is not met* **do**
  Choose a combination of two types of operator according to the roulette method;
  Obtain a new insertion solution $s' \leftarrow repair(removal(s))$;
  Calculate the objective value of the route s' ;
  if s' *is better than* s^* **then**
    $s^* \leftarrow s'$, $s \leftarrow s'$;
  if *Acceptance criterion is met* **then**
    $s \leftarrow s'$;
  Update the scores of operators according to the adaptive mechanism;

The removal operators, repair operators and termination criterion are listed as follows.

A.1 Removal operators

- *Random removal.* Partial requests are randomly selected, after which the origins and destinations associated with these requests are subsequently eliminated from the current routes.

- *Shaw removal.* Three shaw removal indicators are utilized to measure the proximity of request u_1 and request u_2 such as travel distance, arrival time and the size of luggage or parcels.

- *Worst removal.* Initially, the objective function reduction for each request upon its removal from the current solution is calculated. Subsequently, the request with the smallest objective function reduction is eliminated. Finally, repeat this process until a given number of requests have been removed.

A.2 Repair operators

- *Greedy repair.* We select the candidate request with the highest travel fare and assess its feasibility for insertion at every available position. If the results indicate infeasibility at all possible positions, it indicates that no feasible insertion is possible for this request. Repeat this process until all routes can not be inserted.

- *Regret repair.* The regret repair operator uses the regret value to select requests and positions for insertion. First, the regret value for each request upon its insertion from the set of candidate requests is calculated. Then, the request with the maximum regret value is chosen for insertion and eliminated from the set of candidate requests. at last, repeat this process until all routes can not be inserted.

A.3 Termination criterion

The algorithm will terminate if the number of iterations reaches the maximum value defined in the input to the algorithm, or if the temperature of the simulated annealing process drops to a threshold value, or if the best solution remains constant after a specified number of iterations.

B. Appendix B: Heuristic insertion without reoptimization

In this appendix, we present a straightforward heuristic insertion method for dynamic requests scheduling, which does not consider the reoptimization of vehicle routes proposed in section 4.3.2. Upon completion of the static requests scheduling procedure, a set of vehicle routes is generated to be executed at the beginning of the planning period. In each time cycle t, we attempt to identify a potential insertion for each dynamic request, provided that it does not leave the system and regardless of whether it belongs to a new request or a backlogged request. In this approach, any matched static request can never be reassigned to another vehicle. We define S as the solution and $\Pi(S)$ the objective value associated with the solution S. The algorithm of heuristic insertion without reoptimization on dynamic requests scheduling is provided in Algorithm 4.

Function $HeuristicInsertion(S \cup \{i\})$ can determine whether the request i is able to be inserted or not. Firstly, the pick-up vertex of request i is inserted into any existed route in S, and then the drop-off vertex of request i is inserted after finishing insertion of the pick-up vertex. Finally, if this route satisfies time, capacity, and infeasible edges constraints, this new route will replace the old one.

Algorithm 4: Heuristic insertion without reoptimization of vehicle routes on dynamic requests scheduling

Input: An static solution S_0, a set of dynamic requests
Output: New solution S
Initialize $S \leftarrow S_0$;
Each time cycle $t \in T$ a subset of dynamic requests still unmatched is obtained;
for *every request* i *in the subset* **do**
  $S' \leftarrow HeuristicInsertion(S \cup \{i\})$;
  if S' *is feasible and* $\Pi(S') > \Pi(S)$ **then**
    $S \leftarrow S'$;

References

Alidaee, B., Kochenberger, G., Lewis, K., Lewis, M., Wang, H. (2008). A new approach for modeling and solving set packing problems. European Journal of Operational Research, 186(2), 504-512.

Alnaggar, A., Gzara, F., Bookbinder, J. H. (2021). Crowdsourced delivery: A review of platforms and academic literature. Omega, 98, 102139.

Archetti, C., Guerriero, F., Macrina, G. (2021). The online vehicle routing problem with occasional drivers. Computers & Operations Research, 127, 105144.

Arslan, A. M., Agatz, N., Kroon, L., Zuidwijk, R. (2018). Crowdsourced delivery-a dynamic pickup and delivery problem with ad hoc drivers. Transportation Science, 53(1), 222-235.

Audi, 2015. Audi, DHL and Amazon Deliver Convenience. Released in April. <https://www.audi-mediacenter.com/en/photos/detail/audi-dhl-and-amazon-deliver-convenience-2483>

Chen, Z. L., Xu, H. (2006). Dynamic column generation for dynamic vehicle routing with time windows. Transportation Science, 40(1), 74-88.

Cordeau, J. F., Laporte, G. (2007). The dial-a-ride problem: models and algorithms. Annals of operations research, 153, 29-46.

DHL, 2023. Last Mile Delivery: Solutions for Your Business. Released in June. <https://www.dhl.com/discover/en-us/global-logistics-advice/import-export-advice/last-mile-solutions>

Dragomir, A. G., Van Woensel, T., Doerner, K. F. (2022). The pickup and delivery problem with alternative locations and overlapping time windows. Computers & Operations Research, 143, 105758.

Gao, Y., Zhang, S., Zhang, Z., Zhao, Q. (2024). The stochastic share-a-ride problem with electric vehicles and customer priorities. Computers & Operations Research, 164, 106550.

He, Y., Qi, M., Zhou, F., Su, J. (2020). An effective metaheuristic for the last mile delivery with roaming delivery locations and stochastic travel times. Computers & Industrial Engineering, 145, 106513.

Ho, S. C., Szeto, W. Y., Kuo, Y. H., Leung, J. M., Petering, M., Tou, T. W. (2018). A survey of dial-a-ride problems: Literature review and recent developments. Transportation Research Part B: Methodological, 111, 395-421.

Ji, W., Liu, S., Han, K., Li, Y., Liu, T. (2024). The Share-a-Ride Problem with mixed ride-hailing and logistic vehicles. Transportation Research Part E: Logistics and Transportation Review, 192, 103758.

Li, B., Krushinsky, D., Reijers, H. A., Van Woensel, T. (2014). The share-a-ride problem: People and parcels sharing taxis. European Journal of Operational Research, 238(1), 31-40.

Li, B., Krushinsky, D., Van Woensel, T., Reijers, H. A. (2016). The share-a-ride problem with stochastic travel times and stochastic delivery locations. Transportation Research Part C: Emerging Technologies, 67, 95-108.

Li, B., Krushinsky, D., Van Woensel, T., Reijers, H. A. (2016). An adaptive large neighborhood search heuristic for the share-a-ride problem. Computers & Operations Research, 66, 170-180.

Lund, K., Madsen, O. B., Rygaard, J. M. (1996). Vehicle routing problems with varying degrees of dynamism. IMM, Institute of Mathematical Modelling, Technical University of Denmark.

Lu, C. C., Diabat, A., Li, Y. T., Yang, Y. M. (2022). Combined passenger and parcel transportation using a mixed fleet of electric and gasoline vehicles. Transportation Research Part E: Logistics and Transportation Review, 157, 102546.

Mancini, S., Gansterer, M. (2022). Bundle generation for last-mile delivery with occasional drivers. Omega, 108, 102582.

Mara, S. T. W., Norcahyo, R., Jodiawan, P., Lusiantoro, L., Rifai, A. P. (2022). A survey of adaptive large neighborhood search algorithms and applications. Computers & Operations Research, 146, 105903.

Ozbaygin, G., Karasan, O. E., Savelsbergh, M., Yaman, H. (2017). A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations. Transportation Research Part B: Methodological, 100, 115-137.

Ozbaygin, G., Savelsbergh, M. (2019). An iterative re-optimization framework for the dynamic vehicle routing problem with roaming delivery locations. Transportation Research Part B: Methodological, 128, 207-235.

Reyes, D., Savelsbergh, M., Toriello, A. (2017). Vehicle routing with roaming delivery locations. Transportation Research Part C: Emerging Technologies, 80, 71-91.

Retail e-commerce sales worldwide from 2014 to 2027. Released in June. <https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/>

Global parcel shipping volume between 2013 and 2027 (in billion parcels). Released in September. <https://www.statista.com/statistics/1139910/parcel-shipping-volume-worldwide/>

Steever, Z., Karwan, M., Murray, C. (2019). Dynamic courier routing for a food delivery service. Computers & Operations Research, 107, 173-188.

Sze, J. F., Salhi, S., Wassan, N. (2024). An adaptive variable neighbourhood search approach for the dynamic vehicle routing problem. Computers & Operations Research, 164, 106531.

Voigt, S., Kuhn, H. (2022). Crowdsourced logistics: The pickup and delivery problem with transshipments and occasional drivers. Networks, 79(3), 403-426.

Volvo Cars, 2014. Volvo Cars Demonstrates the Potential of Connected Cars with Deliveries Direct to Peoples Cars. Press Release, February. <https://www.media.volvocars.com/global/en-gb/media/pressreleases/139114/volvo-cars-demonstrates-the-potential-of-connected-cars-with-deliveries-direct-to-peoples-cars>.

Vonolfen, S., Affenzeller, M. (2016). Distribution of waiting time for dynamic pickup and delivery problems. Annals of Operations Research, 236, 359-382.

Yu, V. F., Purwanti, S. S., Redi, A. P., Lu, C. C., Suprayogi, S., Jewpanya, P. (2018). Simulated annealing heuristic for the general share-a-ride problem. Engineering Optimization, 50(7), 1178-1197.

Yu, V. F., Zegeye, M. M., Gebeyehu, S. G., Indrakarna, P. A., Jodiawan, P. (2023). A matheuristic algorithm for the share-a-ride problem. Expert Systems with Applications, 230, 120569.

Yu, V. F., Zegeye, M. M., Gebeyehu, S. G., Indrakarna, P. A., Jodiawan, P. (2023). The multi-depot general share-a-ride problem. Expert Systems with Applications, 213, 119044.

Zhan, X., Szeto, W. Y., Wang, Y. (2023). The ride-hailing sharing problem with parcel transportation. Transportation Research Part E: Logistics and Transportation Review, 172, 103073.

Zhang, J., Van Woensel, T. (2023). Dynamic vehicle routing with random requests: A literature review. International Journal of Production Economics, 256, 108751.