# A math-heuristic and exact algorithm for first-mile ridesharing problem with passenger service quality preferences

He, Ping; Jin, Jian Gang; Trépanier, Martin; Schulte, Frederik

**DOI**
10.1016/j.tre.2024.103749

**Publication date**
2024

**Document Version**
Final published version

**Published in**
Transportation Research Part E: Logistics and Transportation Review

**Citation (APA)**
He, P., Jin, J. G., Trépanier, M., & Schulte, F. (2024). A math-heuristic and exact algorithm for first-mile ridesharing problem with passenger service quality preferences. *Transportation Research Part E: Logistics and Transportation Review*, *192*, Article 103749. https://doi.org/10.1016/j.tre.2024.103749

Transportation Research Part E 192 (2024) 103749

Contents lists available at ScienceDirect

Transportation Research Part E

journal homepage: www.elsevier.com/locate/tre

# A math-heuristic and exact algorithm for first-mile ridesharing problem with passenger service quality preferences

Ping He [a,e], Jian Gang Jin [a,b,*], Martin Trépanier [c,d], Frederik Schulte [e]

[a] School of Ocean and Civil Engineering; and State Key Laboratory of Ocean Engineering, Shanghai Jiao Tong University, Shanghai, China
[b] MOE Key Laboratory of Marine Intelligent Equipment and System, Shanghai Jiao Tong University, Shanghai, China
[c] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Montreal, Canada
[d] Department of Mathematics and Industrial Engineering, Polytechnique Montreal, Montreal, Canada
[e] Faculty of Mechanical Engineering, Delft University of Technology, Delft, The Netherlands

ARTICLE INFO

*Keywords:*
First-mile ridesharing
Service quality preferences
Multiple passenger groups
Ride-time minimization
Labeling algorithm with novel dominance rules
Column generation
Branch-and-price

ABSTRACT

With the growing demand for high-quality mobility services, transportation service providers need to offer transit services that not only fulfill passengers' basic travel needs but also ensure an appealing quality of service. During rush hours, fleet sizes are often insufficient to cater to all passenger preferences on service quality, such as ride time and number of co-riders, leading to the sacrifice of service quality for some passengers. Motivated by these practices, we investigate a first-mile ridesharing problem incorporating passenger service quality preferences. This problem involves intricate decisions about the match between requests and vehicles, vehicle routing, and route schedules. To solve this problem, we first develop an arc-based mixed-integer linear programming (MILP) model for this problem. For obtaining near-optimal solutions within practical computation time requirements, we reformulate the MILP model as a trip-based set-partitioning model and propose a math-heuristic algorithm. This algorithm builds upon the column-generation algorithm and tailored bidirectional labeling algorithms with novel dominance rules. Additionally, we introduce a proposition to determine the best schedule for each ridesharing route. To obtain the optimal solution for large-scale instances, we introduce a branch-and-price exact algorithm. Computational experiments based on real-world road networks and randomly generated instances confirm the effectiveness and efficiency of the proposed approaches, demonstrating that the proposed matheuristic finds near-optimal solutions within 40 s for all instances. The results also show that the presented approach significantly improves the quality of first-mile services for prioritized riders, with the ratio of satisfied requests increasing by 23% even when the fleet is generally insufficient.

## 1. Introduction

Ridesharing has experienced rapid expansion in recent years, emerging as an efficient, cost-effective, and environmentally friendly mode of transportation. With the increasing popularity of ridesharing services, there has been a continuous rise in the number of users, leading to greater diversity among passengers in terms of age, gender, and income (Si et al., 2023). Consequently, the range of passenger service preferences, including ride time, the number of co-riders, and travel costs, has become more pronounced. In response to these diverse preferences for service quality, transportation network companies (TNCs), such as Lyft, Uber, and Didi, have diversified their ridesharing services. For instance, Didi offers ridesharing options with fewer co-riders and

* Correspondence to: Dongchuan 800, Shanghai Jiao Tong University, Shanghai 200240, China.
*E-mail address:* jiangang.jin@sjtu.edu.cn (J.G. Jin).

https://doi.org/10.1016/j.tre.2024.103749
Received 22 January 2024; Received in revised form 25 July 2024; Accepted 27 August 2024
Available online 5 September 2024

shorter trip times for passengers seeking higher service quality, while providing services with more co-riders and longer trip times for those prioritizing cost-effectiveness.

When designing ridesharing services for passengers heading to the same metro or train station, also known as the first-mile ridesharing problem (Bian and Liu, 2019; Bian et al., 2020, 2022; Zheng and Pantuso, 2023), it is essential to consider passenger preferences on service quality. During rush hours, the fleet size may struggle to meet the service quality needs of all passengers, potentially compromising the service quality of some passengers. For instance, private operators such as TNCs typically prioritize offering high-quality first-mile services to riders who generate high revenue (Bian et al., 2020). In contrast, public transit agencies may prioritize delivering high-quality services to high-demand corridors or specific passenger groups, such as the elderly, with the goal of promoting equity (Zuo et al., 2020). These groups rely heavily on public transportation and have limited access to alternative travel modes compared to general travelers. The elderly, for instance, may have reduced accessibility to driving or cycling due to physical limitations (Brown et al., 2021).

Given the short travel distances between first-mile travelers and the station, private or TNCs-affiliated drivers often exhibit reluctance to provide first-mile travel services due to the perceived lower revenue potential of these short trips (Human Transit). Consequently, addressing the first-mile travel challenge becomes an inescapable responsibility for public transportation agencies. Operating a dedicated fleet for first-mile travel services is financially impractical for public transit agencies (Currie and Fournier, 2020). As a result, public transit agencies are exploring cost-effective solutions, such as providing subsidies to drivers or establishing partnerships with TNCs to facilitate first-mile ridesharing services (Chandler, Arizona; RTA Mobility Pilots). For instance, Solano Mobility and Lyft have partnered to provide 80% off Lyft rides (up to $25) to transit centers, train stations, and express bus stops (Solano Mobility).

Building on the aforementioned analysis, this paper investigates the first-mile ridesharing problem with passengers' service quality preferences, encompassing tolerable ride times and maximum number of co-riders. Unlike previous studies that treated service quality requirements as hard constraints, we relax them as soft constraints. Specifically, if the ridesharing service exceeds a passenger's service quality preference, a penalty is imposed, with the penalty value determined by the extent of the deviation. This relaxation ensures that more passengers can access first-mile ridesharing services, especially during fleet shortages. Additionally, we categorize passengers into general and high-priority groups to better accommodate their service preferences. Priority is given to meeting the service quality preferences of high-priority passengers, particularly in cases of fleet insufficiency.

Incorporating service quality preferences into the first-mile ridesharing problem adds complexity to decision-making. For instance, when considering passengers' ride time preferences as soft constraints, alongside hard constraints such as earliest service times and latest arrival times, the optimal departure times of vehicles and pickup times of passengers must be determined for a ridesharing route. This optimal schedule aims to satisfy passengers' ride time preferences as much as possible, thereby minimizing penalty costs. In contrast, traditional first-mile ridesharing problems typically set passenger ride times as hard constraints, requiring only a feasible schedule. The 'time slack' method, commonly used in the dial-a-ride problem (Cordeau and Laporte, 2003; Sohrabi et al., 2024), can be applied to determine a feasible schedule (He et al., 2023). However, this method becomes inadequate for determining an optimal schedule. Consequently, it is essential to explore alternative approaches to identify an optimal schedule for first-mile ridesharing routes.

To address this challenging problem, we first present two mathematical models, both aiming to minimize total operating costs. Then, we develop a branch-and-price exact algorithm to obtain optimal solutions and a column-generation math-heuristic algorithm to identify near-optimal solutions within a short computation time. The exact and matheuristic algorithms build upon a dedicated bidirectional labeling algorithm with a novel dominance rule for generating columns and a problem-specific strategy for reusing columns. In addition, a tailored label extension procedure is introduced to check the feasibility and identify the best schedule for each route. More importantly, our models and algorithms cover more practically relevant aspects, such as earliest pickup times and service quality preferences, that have never been combined into a single model or algorithm. Accordingly, the contributions of this work are summarized as follows.

(1) We investigate a novel first-mile ridesharing problem that incorporates passenger service quality preferences, which covers more practically relevant aspects and necessitates additional decisions to identify the best schedule for each ridesharing route. To model this problem, we develop an arc-based mixed-integer linear programming (MILP) model and then reformulate it as a trip-based set-partitioning model.

(2) We introduce a proposition for determining the optimal schedule for a given first-mile ridesharing route. We then introduce a bidirectional labeling algorithm to identify promising trips (ridesharing routes with optimal schedules) based on the proposition. A novel dominance rule is designed to eliminate unpromising labels, which builds upon the characteristic where penalties associated with a label present a piecewise-linear convex function during label extension processes.

(3) We develop a branch-and-price algorithm to obtain exact solutions for this challenging problem and a column-generation math-heuristic algorithm to yield high-quality solutions within practical computation time requirements. Several problem-specific strategies are devised to expedite the solving process of the math-heuristic algorithm, which may be applicable to related ridesharing and routing problems with heterogeneous fleets and soft constraints on travel times.

The remainder of this study is organized as follows. Section 2 reviews related work and positions our work. Section 3 elaborates on the first-mile ridesharing problem with passenger service quality preferences and introduces two mathematical models. A column-generation math-heuristic algorithm and a branch-and-price algorithm are devised in Section 4. Real-world case studies are conducted in Section 5. Conclusions and future work are presented in Section 6.

## 2. Literature review

There are two areas of research related to this study: research on heterogeneous transportation services and research on the first-mile ridesharing problem.

### 2.1. Related research on heterogeneous transportation services

Some studies have explored the provision of transportation services with heterogeneous service qualities. For instance, Wong et al. (2008) explored an urban taxi service with multiple passenger groups (low- and high-income) and vehicle types (luxury and normal taxis). Smith et al. (2010) addressed a dynamic vehicle routing problem that utilized multiple types of vehicles to service demands with multiple priority classes and heavily penalized service delays for high-priority demand classes. Zhen et al. (2016) investigated a multiperiod yard template planning problem considering the heterogeneous periodicities of vessels and developed a local branching-based approach, as well as a Particle Swarm Optimization (PSO) algorithm to solve this problem. Doan et al. (2021) proposed a vehicle routing problem with relaxed priority rules for several groups of customers, where customers with the highest priorities are typically served before the lower priority ones. Lin et al. (2023) focused on the post-harvest preprocessing problem for fruits and vegetables. They employed a heterogeneous fleet to provide multiple preprocessing and transportation services. He et al. (2024) investigated a flexible airport bus routing problem that considers service priorities for business travelers and developed two heuristic algorithms to obtain high-quality solutions. These studies show that providing heterogeneous services for different passenger groups is helpful in reducing total operating costs or improving service quality for passengers.

In some practical situations, transportation service providers have to offer multiple types of services for different passenger groups due to limited fleet size or vehicle capacity. For instance, Bulhoes et al. (2018) addressed a vehicle routing problem with service level constraints for several groups of delivery demands. In this scenario, the delivery service provider must prioritize serving a subset of requests according to their relative profitability since the fleet size or vehicle capacity is limited. Beirigo et al. (2022a) and Beirigo et al. (2022b) explored a ridesharing problem with service-level contracts for heterogeneous passengers, where idle autonomous vehicles could be dynamically hired to inflate the fleet and offer ridesharing services. They found that dynamically hiring free-floating vehicles can meet the strict service level constraints for different passenger groups, but service quality suffers without a sufficient fleet.

### 2.2. Related work on first-mile ridesharing problem

In recent years, the shared economy has facilitated the rapid growth of on-demand travel services for riders, like ridehailing and ridesharing. Ridesharing, in particular, is a flexible, cost-effective, and high-quality transportation mode, making it an effective means of addressing first-mile travel challenges (Masoud et al., 2017; Shen et al., 2018; Stiglic et al., 2018; Ma et al., 2019; Bian and Liu, 2019; Bian et al., 2020, 2022; Chen et al., 2020; Kumar and Khani, 2021; Grahn et al., 2022; Ye et al., 2022). For instance, Stiglic et al. (2018) explored the potential benefits of integrating ridesharing services and public transportation systems and found that this kind of service can potentially enhance the first-mile and last-mile connectivity with public transit systems as well as increase the use of public transportation.

In real-world scenarios, obtaining a ridesharing solution quickly is often imperative, leading to the prevalent use of simulation or metaheuristic algorithms in current studies. For instance, Bian employed solution-pool (SP) approaches to seek feasible solutions (Bian and Liu, 2019; Bian et al., 2020, 2022). Chen et al. (2020) devised a clustering-based heuristic algorithm for rapid solution acquisition. Kumar and Khani (2021) developed matching algorithms capable of solving the problem within five minutes. Huang et al. (2022) introduced a rule-based matching algorithm to address the problem efficiently. However, these methods suffer from unstable solution quality. Bian pointed out that the quality of solutions obtained by the SP algorithm is slightly worse than those obtained by CPLEX within three hours for instances with fewer than 41 nodes (Bian et al., 2022).

Recently, math-heuristic algorithms have become popular for identifying high-quality ridesharing solutions within practical computation time. For instance, Lu et al. (2022) introduced a network partitioning math-heuristic algorithm that determines near-optimal solutions within six minutes for the passenger and parcel ridesharing problem. He et al. (2024) developed a Benders decomposition math-heuristic algorithm to solve the flexible bus and last-mile ridesharing problem from the airport, with numerical experiments demonstrating that this algorithm can obtain optimal or near-optimal solutions within practical computation time. Therefore, we also resort to the math-heuristic algorithm to identify high-quality solutions in real time for the first-mile ridesharing problem.

Additionally, as summarized in Table 1, existing studies on first-mile ridesharing tend to disregard passengers' earliest pickup time constraints and set passenger ride times as hard constraints. If the earliest pickup time constraint is not considered (Shen et al., 2018; Bian and Liu, 2019; Bian et al., 2020; Beirigo et al., 2022b; Grahn et al., 2022; Zheng and Pantuso, 2023), the passenger's travel time is not affected by the vehicle's departure time. To satisfy the latest arrival time requirement, vehicles can depart as early as possible. However, in the presence of earliest pickup time constraints, this strategy may lead to vehicle waiting times to serve some passengers, which may increase the travel time of passengers already on board (He et al., 2023; Wang et al., 2023).

When considering earliest pickup times and ride times as hard constraints (He et al., 2023), a feasible schedule for a given first-mile ridesharing route must be determined, including the vehicle departure time and passenger pickup times. A feasible schedule can be determined based on the 'time slack' method commonly employed in the dial-a-ride problem (Cordeau and Laporte, 2003;

**Table 1**
Overview of literature in first-mile ridesharing.

| Reference | Main consideration | | | | | | | Solution method |
|---|---|---|---|---|---|---|---|---|
| | Fleet | Route schedule | Hete. service | Service quality | MRT | MCR | LAT | |
| Shen et al. (2018) | Homo. | | | Hard con. | ✓ | | | Agent-based simulation |
| Stiglic et al. (2018) | Homo. | | | Hard con. | ✓ | | ✓ | Enumeration |
| Bian and Liu (2019) | Homo. | | | Hard con. | ✓ | ✓ | ✓ | Solution pooling heuristic |
| Bian et al. (2020) | Homo. | | ✓ | Hard con. | ✓ | | ✓ | Solution pooling heuristic |
| Chen et al. (2020) | Homo. | | | Hard con. | ✓ | | | Cluster-based algorithm |
| Kumar and Khani (2021) | Homo. | | | Hard con. | ✓ | | ✓ | A matching algorithm |
| Bian et al. (2022) | Hete. | | ✓ | Hard con. | ✓ | ✓ | ✓ | Solution pooling heuristic |
| Huang et al. (2022) | Homo. | | | Hard con. | ✓ | | ✓ | Dynamic pooling algorithm |
| Grahn et al. (2022) | Hete. | | | Hard con. | ✓ | | | Matching-routing algorithm |
| Ye et al. (2022) | Homo. | | ✓ | Hard con. | | | ✓ | Cluster-based algorithm |
| He et al. (2023) | Homo. | ✓ | ✓ | Hard con. | ✓ | | ✓ | ALNS algorithm |
| Zheng and Pantuso (2023) | Hete. | | ✓ | Hard con. | | | ✓ | Evolutionary algorithm |
| This study | Hete. | ✓ | ✓ | Soft con. | ✓ | ✓ | ✓ | Column generation heuristic |

* Homo.: Homogeneous; Hete.: Heterogeneous; Hard con.: Hard constraint; Soft con.: Soft constraint; MRT: Maximum ride time of a request; MCR: request's maximum number of co-riders; LAT: request's latest arrival time; ALNS: Adaptive large neighborhood search.

Bongiovanni et al., 2024; Sohrabi et al., 2024). For example, He et al. (2023) designed a route feasibility check algorithm that incorporates the idea of 'time slack'.

To ensure that more passengers can be served during peak hours, we relax passenger ride time preferences to soft constraints. In this case, passenger ride time preferences can be unsatisfied (but incur a penalty cost), necessitating the determination of an optimal schedule. However, the 'time slack' method is unable to identify an optimal schedule under these conditions. To fill this gap, we introduce a proposition and design a bidirectional labeling algorithm suitable for this scenario.

## 3. First-mile ridesharing with passenger service quality preferences

### 3.1. Problem description

Consider a public transit agency that operates a ridesharing platform for delivering on-demand first-mile ridesharing services. Both drivers and requests submit their first-mile trip details to the platform in advance. Drivers who are willing to share the ride to the station provide essential information, encompassing the departure location (also known as ready location), earliest departure time (ready time), latest arrival time, and the number of available seats. For each request, the first-mile travel information includes the pickup location, number of riders, earliest pickup time, latest arrival time, maximum willingness to share the trip with others (also known as the maximum number of co-riders), and detour tolerance (maximum ride time).

The platform determines the best ridesharing scheme after collecting information on drivers and requests. The objective of the agency is to minimize total operating costs, encompassing a fixed subsidy for each vehicle employed, transportation cost per kilometer for each vehicle, and the penalties associated with unmet requests. A penalty cost is incurred if a request's service quality preferences for co-riding and detouring cannot be met. The service provider categorizes passengers into types (general or high-priority groups) according to their registration or order information. To ensure service priority for high-priority requests, the penalty coefficient for them is larger than that for general requests. Both riders and drivers are notified of their first-mile service when the ridesharing scheme is determined. Each driver (vehicle) departs to pick up matched passengers after the earliest departure time from the parking garage and must arrive at the station before the latest arrival time of the driver. Similarly, each request can be picked up after its earliest pickup time and must arrive at the station before its latest arrival time.

The transit agency can employ the rolling horizon method to partition the entire planning horizon into a sequence of time intervals with uniform durations and fixed spacing between adjacent intervals (Chen et al., 2020; He et al., 2023), such as 10 min. This approach reduces decision-challenging and enables dynamic responsiveness to incoming requests and available vehicles. At the beginning of each time horizon, the agency determines the best ridesharing scheme for requests whose earliest pickup time falls within that interval and for drivers whose earliest departure time is smaller than the end time of the interval.

Fig. 1 illustrates a ridesharing scheme for six requests and two available vehicles, with requests 1, 2, and 3 assigned to vehicle 1, and other requests assigned to vehicle 2. To better meet the service quality preferences of high-priority requests, it may sacrifice the service quality of general requests, especially when the fleet is insufficient. For instance, to meet the detour tolerance of high-priority request 6, vehicle 2 first picks up request 5 and then heads to pick up request 6. If the system does not consider service heterogeneity, vehicle 2 will first pick up request 6 and then request 5.

### 3.2. Notations

Let $\mathcal{N}$ denote the set of all nodes, including request nodes, ready locations of available cars, and the station which can be denoted by 0. Let $\mathcal{A}$ denote the set of arcs connecting the nodes of $\mathcal{N}$, $\mathcal{A} = \{(i,j)|i,j \in \mathcal{N}, i \neq j\}$. Then, this problem can be defined on a directed graph $G = (\mathcal{N}, \mathcal{A})$. Other notations are defined in Table 2.

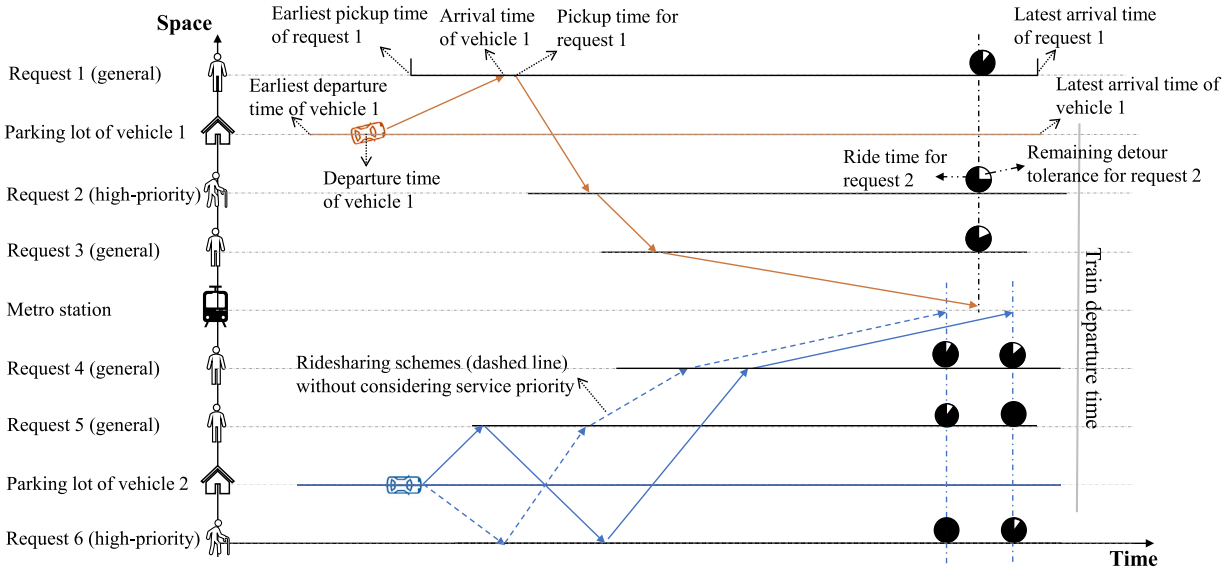



**Fig. 1.** An illustrative example of a ridesharing scheme for the first-mile ridesharing problem with service heterogeneity.

**Table 2**
Parameters and decision variables.

| Notations | Descriptions |
|---|---|
| **Sets** | |
| $\mathscr{K}$ | Set of available cars |
| $\mathscr{R}_D$ | Set of high-priority requests whose earliest pickup time is within the current time horizon |
| $\mathscr{R}_G$ | Set of general requests whose earliest pickup time is within the current time horizon |
| $\mathscr{R}$ | Set of all requests, $\mathscr{R} = \mathscr{R}_D \cup \mathscr{R}_G$ |
| $\mathscr{G}$ | Set of ready locations for cars, $\mathscr{G} = \{g_1, g_2, \ldots, g_{k=\lvert\mathscr{K}\rvert}\}$ |
| $\mathscr{N}$ | Set of all nodes, $\mathscr{N} = \mathscr{R} \cup \mathscr{G} \cup \{0\}$ |
| **Parameters** | |
| $q_k$ | Available seat numbers of car $k$ |
| $r_k$ | Ready time for car $k$, indicating when it will be available to offer first-mile service |
| $L_k$ | Latest arrival time at the station for the driver of car $k$ |
| $p_i$ | Number of riders of request $i$, $i \in \mathscr{R}$ |
| $s_i$ | Service time for request $i$, $i \in \mathscr{R}$ |
| $\varpi_i$ | Maximum number of co-riders for request $i$, $i \in \mathscr{R}$ |
| $u_i$ | Maximum ride time for request $i$, $i \in \mathscr{R}$ |
| $e_i$ | Earliest service start time for request $i$, $i \in \mathscr{R}$ |
| $a_i$ | Latest arrival time for request $i$, $i \in \mathscr{R}$ |
| $T$ | Maximum value of the latest arrival time among all requests |
| $P$ | Total number of riders among all requests |
| $c$ | Transportation cost ($) per kilometer for each vehicle |
| $\varsigma_g/\varsigma_d$ | Penalty factor ($) for the number of co-riders exceeding the co-riding willingness of a general /high-priority request |
| $\alpha_g/\alpha_d$ | Penalty factor ($) for the ride time exceeding the detour tolerance of a general/high-priority request |
| $d_{ij}$ | Distances (km) between nodes $i$ and $j$, $i, j \in \mathscr{N}$ |
| $f$ | Fixed cost ($) for hiring a vehicle |
| $v$ | Average travel speed (km/hour) for each vehicle |
| **Variables** | |
| $x_{ij}^k$ | $\in \{0,1\}$. 1 if vehicle $k$ traverses arc $(i,j)$; otherwise, 0. |
| $y_k$ | $\in \{0,1\}$. 1 if vehicle $k$ is hired; otherwise, 0. |
| $t_i^k$ | $\geq 0$. Service start time for request $i$ served by vehicle $k$. |
| $dt_k$ | $\geq 0$. Departure time of vehicle $k$. |
| $\varrho_k$ | $\in \mathbb{Z}^+$. Carried number of riders when car $k$ arrives at the metro station. |
| $\zeta_i$ | $\in \mathbb{Z}^+$. Number of co-riders exceeding the co-riding willingness for request $i$, $i \in \mathscr{R}$. |
| $\xi_i$ | $\geq 0$. Total ride time for request $i$, $i \in \mathscr{R}$. |
| $z_i$ | $\geq 0$. Ride time exceeding the detour tolerance for request $i$, $i \in \mathscr{R}$. |

### 3.3. Arc-based MILP model

Then, we can formulate this problem as an arc-based mixed-integer linear programming model:

$$min \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} c\ d_{ij}x_{ij}^k + \sum_{k\in\mathcal{K}} f\ y_k + \sum_{i\in\mathcal{R}_G} p_i(\alpha_g z_i + \varsigma_g\zeta_i) + \sum_{i\in\mathcal{R}_D} p_i(\alpha_d z_i + \varsigma_d\zeta_i) \tag{1}$$

subject to:

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{R}\cup\{0\}\setminus\{i\}} x_{ij}^k = 1, \forall i\in\mathcal{R} \tag{2}$$

$$\sum_{j\in\mathcal{R}\cup\{g_k\}\setminus\{i\}} x_{ji}^k - \sum_{j\in\mathcal{R}\cup\{0\}\setminus\{i\}} x_{ij}^k = 0, \forall i\in\mathcal{R}, \forall k\in\mathcal{K} \tag{3}$$

$$y_k = \sum_{j\in\mathcal{R}} x_{g_k,j}^k = \sum_{i\in\mathcal{R}} x_{i0}^k \le 1, \forall k\in\mathcal{K} \tag{4}$$

$$\varrho_k = \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{R}\cup\{0\}\setminus\{i\}} p_i\ x_{ij}^k \le q_k, \forall k\in\mathcal{K} \tag{5}$$

$$\zeta_i \ge \varrho_k - p_i - \varpi_i - P(1 - \sum_{j\in\mathcal{R}\cup\{0\}\setminus\{i\}} x_{ij}^k), \forall k\in\mathcal{K}, \forall i\in\mathcal{R} \tag{6}$$

$$t_j^k \ge t_i^k + d_{ij}/v + s_i - T\left(1 - x_{ij}^k\right), \forall i\in\mathcal{R}, \forall j\in\mathcal{R}\cup\{0\}, \forall k\in\mathcal{K} \tag{7}$$

$$t_j^k \ge dt_k + d_{g_k,j}/v - T\left(1 - x_{g_k,j}^k\right), \forall j\in\mathcal{R}, \forall k\in\mathcal{K} \tag{8}$$

$$e_i \le t_i^k \le a_i - d_{i0}/v - s_i, \forall i\in\mathcal{R}, \forall k\in\mathcal{K} \tag{9}$$

$$a_i + T(1 - \sum_{j\in\mathcal{R}\cup\{0\}} x_{ij}^k) \ge t_0^k, \forall k\in\mathcal{K}, i\in\mathcal{R} \tag{10}$$

$$L_k \ge t_0^k, \forall k\in\mathcal{K} \tag{11}$$

$$\xi_i \ge t_0^k - t_i^k - T(1 - \sum_{j\in\mathcal{R}\cup\{0\}} x_{ij}^k), \forall i\in\mathcal{R}, \forall k\in\mathcal{K} \tag{12}$$

$$z_i \ge \xi_i - u_i, \forall i\in\mathcal{R} \tag{13}$$

$$dt_k \ge r_k, \forall k\in\mathcal{K} \tag{14}$$

$$x_{ij}^k, y_k \in \{0,1\}, \forall i,j\in\mathcal{N}, \forall k\in\mathcal{K} \tag{15}$$

$$dt_k, t_i^k \ge 0, \forall i\in\mathcal{R}\cup\{0\}, \forall k\in\mathcal{K} \tag{16}$$

$$z_i, \xi_i \ge 0, \forall i\in\mathcal{R} \tag{17}$$

$$\zeta_i \in \mathbb{Z}^+, \forall i\in\mathcal{R} \tag{18}$$

$$\varrho_k \in \mathbb{Z}^+, \forall k\in\mathcal{K} \tag{19}$$

The objective function (1) aims to minimize the total costs associated with the first-mile service. These costs include transportation costs, fixed costs for hiring cars, and penalty costs. Transportation costs are calculated by multiplying the total travel distances by the mileage cost per kilometer. Penalty costs are incurred when the number of co-riders exceeds the maximum willingness to share the trip for each request, or when the actual ride time exceeds the maximum willingness to detour. The penalty costs for each request are determined based on the penalized number of co-riders or the penalized ride time, multiplied by the corresponding penalty coefficient. To ensure service priority for high-priority passenger groups, their penalty coefficients are higher than those for general requests.

Constraints (2) denote each request must be matched with a ridesharing route. Constraints (3) impose that each request is picked up and delivered by the same vehicle and ensure that the service flow is conserved. Constraints (4) denote whether vehicle $k$ is hired to render services. If hired, the vehicle will depart from the ready location ($g_k$) and ultimately arrive at the station. Constraints (5) ensure that the total number of riders to be picked up meets the capacity constraint of each car. Constraints (6) determine the number of co-riders that exceed the co-riding willingness for each request. Constraints (7)–(8) specify the pickup time at each node, the departure time of each car, and the relationship between the departure time and pickup time. Constraints (9) ensure that the

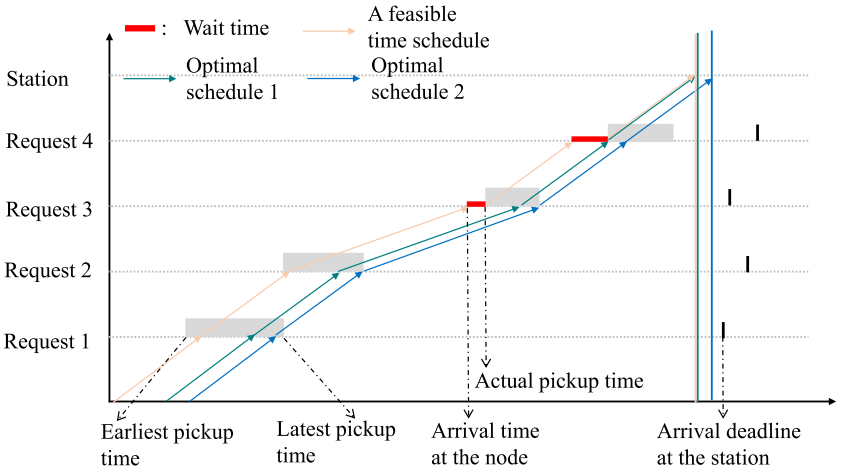



**Fig. 2.** A feasible and two optimal schedules for a route.

pickup time for each request must be larger than its earliest pickup time and smaller than the latest arrival time minus the direct travel time from that request to the metro station. Constraints (10) and (11) enforce that the actual arrival time at the station meets riders' requirements and is earlier than the latest arrival time of the vehicle. Constraints (12) identify the ride time for each request. Constraints (13) determine the penalized ride time of each request when the actual ride time exceeds its maximum tolerance towards the detour time. Constraints (14) mean that the departure time of each car is larger than its ready time. Constraints (15)–(19) define the domain of decision variables.

Note that although the model only considers two classes of passengers, it can easily be extended to scenarios with multiple passenger classes by simply modifying the passenger set parameters and penalty costs associated with different passenger classes in the objective function.

### 3.4. Route, schedule, and trip

Consider a given route $\mathbb{S}_k$ of vehicle $k$ with the route node sequence $\{i_0, i_1, \dots, i_{n-1}, i_n\}$, where $i_0$ denotes the ready location of vehicle $k$, $i_n$ is either a request or the station, and $i_1, \dots, i_{n-1}$ indicate requests served by the route. Without loss of generality, we assume that route $\mathbb{S}_k$ satisfies vehicle $k$'s capacity and requests' pickup time constraints. Then, we can check the feasibility of the latest arrival time and identify the best departure time for vehicle $k$ and the optimal pickup time for requests to minimize the total ride time penalty of route $\mathbb{S}_k$. We define the departure and pickup times as a schedule for route $\mathbb{S}_k$. Let $\mathbb{S}_k(t) = \{t_0, t_1, \dots, t_n\}$ denote a feasible schedule that meets the pickup time and latest arrival time requirements for each matched request. Here, $t_0$ refers to the departure time at the parking lot or the ready location of vehicle $k$, and $t_n$ is the pickup time at node $n$ or the arrival time at the station.

**Definition 1** (*Optimal Schedule of Route* $\mathbb{S}_k$)**.** Let $C(\mathbb{S}_k, \mathbb{S}_k(t))$ be the total cost of route $\mathbb{S}_k$ associated with schedule $\mathbb{S}_k(t)$, and let $\Omega(\mathbb{S}_k(t))$ be the set of feasible schedules for route $\mathbb{S}_k$. The schedule $\overline{\mathbb{S}_k(t)}$ is an optimal schedule for route $\mathbb{S}_k$ if and only if $C(\mathbb{S}_k, \overline{\mathbb{S}_k(t)}) \leq C(\mathbb{S}_k, \mathbb{S}_k(t))$, $\forall\ \mathbb{S}_k(t) \in \Omega(\mathbb{S}_k(t))$.

Note that multiple optimal schedules may exist for route $\mathbb{S}_k$. For instance, Fig. 2 presents two optimal schedules for the route with four requests. In such cases, we select the schedule with the earliest arrival time at last node $n$ as the best one, such as schedule 1 in Fig. 2, as it enables the vehicle to arrive at the destination earlier than other schedules. We refer to this type of optimal schedule as $\overrightarrow{\mathbb{S}_k(t)}$.

**Definition 2** (*Trip*)**.** Let route $\mathbb{S}_k$ of vehicle $k$ and an optimal schedule $\overline{\mathbb{S}_k(t)}$ denote a trip.

### 3.5. A set partitioning formulation

The MILP model aims to find the best trip for each car, which can be decomposed into a series of sub-problems to determine promising trips and a master problem to select the best combination between trips. Let $\mathbb{T}_k$ be the set of trips for vehicle $k$, and let $\delta_{\tau_k}(\tau_k \in \mathbb{T}_k)$ (a binary decision variable) denote whether the trip $\tau_k \in \mathbb{T}_k$ is selected. For a given trip $\tau_k$, let $\epsilon_{\tau_k}$ denote total costs. $\epsilon_{\tau_k}$ consists of transportation costs, penalty costs for requests matched by car $k$, and fixed costs of car $k$. The binary parameter $\chi^i_{\tau_k}$ is set to one if request $i$ ($i \in \mathscr{R}$) is matched by trip $\tau_k$ and zero otherwise. Then, the set partitioning model is formulated as

$$[MP]\ min \sum_{k \in \mathscr{K}} \sum_{\tau_k \in \mathbb{T}_k} \epsilon_{\tau_k} \delta_{\tau_k} \tag{20}$$

subject to:

$$\sum_{k\in\mathcal{K}}\sum_{\tau_k\in\mathbb{T}_k}\chi^i_{\tau_k}\delta_{\tau_k}=1,\forall i\in\mathcal{R} \tag{21}$$

$$\sum_{\tau_k\in\mathbb{T}_k}\delta_{\tau_k}\le 1,\forall k\in\mathcal{K} \tag{22}$$

$$\delta_{\tau_k}\in\{0,1\},\forall\tau_k\in\mathbb{T}_k,\forall k\in\mathcal{K} \tag{23}$$

The objective function (20) minimizes the total costs of selected trips. Constraints (21) denote that each request must be served. Constraints (22) denote that each car can be deployed on at most one route. Constraints (23) are the integrality constraint.

## 4. Solution methodology

Enumerating all trips in the MP formulation is impractical due to the exponential number of complete trips. Instead, a common approach is to relax the binary decision variable $\delta_{\tau_k}$ into a continuous variable, and then solve the restricted MP, also known as the RMP, focusing on a subset $\overline{\mathbb{T}}_k$ of $\mathbb{T}_k$ using column generation algorithm (Desaulniers et al., 2006; Zhen et al., 2022, 2024). To identify promising trips for the RMP, pricing sub-problems (PSP) associated with each vehicle are solved. In the following, we will first introduce the RMP and the PSP. Then, we will present a customized bidirectional labeling algorithm designed to solve the PSP. Finally, we will showcase two algorithms, a column generation heuristic and a branch-and-price exact algorithm, for solving the RMP and MP.

### 4.1. Restricted Master Problem (RMP)

To ensure that the RMP is always feasible, we introduce a set of auxiliary decision variables $v_i, i\in\mathcal{R}$ to represent the initial column set $\mathbb{T}_0$. If the value of $v_i, i\in\mathcal{R}$ is positive, it will incur a penalty cost in the objective function.

$$[RMP]\ min\sum_{k\in\mathcal{K}}\sum_{\tau_k\in\overline{\mathbb{T}}_k}\epsilon_{\tau_k}\delta_{\tau_k}+\sum_{i\in\mathcal{R}}Mv_i \tag{24}$$

subject to:

$$\sum_{k\in\mathcal{K}}\sum_{\tau_k\in\overline{\mathbb{T}}_k}\chi^i_{\tau_k}\delta_{\tau_k}+v_i=1,\forall i\in\mathcal{R} \tag{25}$$

$$\sum_{\tau_k\in\overline{\mathbb{T}}_k}\delta_{\tau_k}\le 1,\forall k\in\mathcal{K} \tag{26}$$

$$\delta_{\tau_k}\ge 0, v_i\ge 0,\forall\tau_k\in\overline{\mathbb{T}}_k,\forall k\in\mathcal{K} \tag{27}$$

The set partitioning models typically provide better lower bounds than those identified by original formulations (Costa et al., 2019), for instance, the arc-based MILP formulation of this problem.

### 4.2. Pricing subproblem (PSP)

This problem involves a heterogeneous fleet, where each vehicle has different numbers of available seats, ready times, and ready locations. Therefore, it is necessary to determine promising trips for each vehicle, including the number of matched requests, the departure time, passenger pickup times, and the arrival time at the station. The pricing subproblem of the RMP model serves to find trips that can be added to the current set of columns with negative reduced costs for each car, which can be formulated as an elementary shortest path problem with resource constraints (ESPPRC). Consequently, the detailed matching strategies between requests and vehicles are determined by this pricing problem and the associated solving algorithm. Let $\pi_i(i\in\mathcal{R})$ and $\omega_k(k\in\mathcal{K})$ be the dual variables of constraints (25) and (26), respectively. Then, the reduced cost of trip $\tau_k$ associated with car $k$ is $\overline{\epsilon_{\tau_k}}=\epsilon_{\tau_k}-\sum_{i\in\mathcal{R}}\chi^i_{\tau_k}\pi_i-\omega_k$. Let $x_{ij}$ be a binary variable that denotes whether the arc $(i,j)$ is traversed. Therefore, the subproblem for vehicle $k$ can be formulated as:

$$[\mathbf{PSP}]\ min\ \overline{\epsilon_{\tau_k}}=\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}}c\ d_{ij}\ x_{ij}+f\ +\sum_{i\in\mathcal{R}_G}p_i(\alpha_g z_i+\varsigma_g\zeta_i)+\sum_{i\in\mathcal{R}_D}p_i(\alpha_d z_i+\varsigma_d\zeta_i)-\sum_{i\in\mathcal{R}}\chi^i_{\tau_k}\pi_i-\omega_k \tag{28}$$

Constraints for the PSP model can be obtained directly by dropping index $k$ in constraints (2)–(19).

The ESPPRC is a well-known NP-hard problem (Dror, 1994), and the common method to solve the ESPPRC as well as its variants to optimality is to employ the labeling algorithm. However, this PSP problem, in addition to deciding which requests to serve and in what order, also involves schedule decisions regarding the vehicle's departure time and the pickup times of matched requests. To improve the efficiency of solving the PSP, we devise a tailored bidirectional labeling algorithm. This approach involves generating labels, or partial paths, from both the forward and backward directions, and concatenating the resulting forward and backward labels to form complete routes.

We will introduce the forward and backward labeling algorithms in Sections 4.4 and 4.5, respectively. As mentioned in Righini and Salani (2006), the computation time of bidirectional labeling algorithms can be reduced by setting a critical resource, for instance, the medium time of the planning horizon. We also select the medium time as the critical resource. To speed up the entire computational efficiency, we propose two heuristic labeling algorithms in Sections 4.4 and 4.5 and integrate them into a PSP-solving procedure for the column generation algorithm.

**Table 3**
Notations for defining forward labels.

| | |
|---|---|
| $\mathbb{P}(\mathcal{L}_i^k)$ | Set of nodes visited by label $\mathcal{L}_i^k$; |
| $C(\mathcal{L}_i^k)$ | Total reduced cost of the partial path $\mathbb{P}(\mathcal{L}_i^k)$; |
| $\mathcal{V}(\mathcal{L}_i^k)$ | Number of riders that vehicle $k$ carried after visiting node $i$; |
| $\mathbb{U}(\mathcal{L}_i^k)$ | Vector of visited requests and inextensible requests for this label; |
| $LA(\mathcal{L}_i^k)$ | Latest arrival time for requests visited by label $\mathcal{L}_i^k$, with an initial value of $L_k$; |
| $DR_n(\mathcal{L}_i^k)$ | Minimum trip duration from node $n$ to $i$ along the route nodes of label $\mathcal{L}_i^k$; |
| $\Omega_n(\mathcal{L}_i^k)$ | Remaining number of co-riders without co-riding penalty for request $n$ of $\mathcal{L}_i^k$; |
| $\Delta_n(\mathcal{L}_i^k)$ | Remaining ride time without exceeding the detour tolerance for request $n$ of $\mathcal{L}_i^k$; |
| $EVT_n(\mathcal{L}_i^k)$ | Earliest pickup time at node $n$ considering $LA(\mathcal{L}_i^k)$; |
| $LVT_n(\mathcal{L}_i^k)$ | Latest pickup time at node $n$ considering $LA(\mathcal{L}_i^k)$. |

### 4.3. Identifying the optimal schedule for a given ridesharing route

For a given ridesharing route, the eight-step 'time slack' procedure widely applied in dial-a-ride problems (Cordeau and Laporte, 2003) can obtain a feasible schedule that meets the hard constraint of passenger ride times. However, in this problem, it is necessary to identify the best schedule for minimizing total ride time penalties since the ride time constraint is relaxed from hard to soft. As a result, the 'time slack' procedure is not applicable to this first-mile ridesharing problem.

To address this, we first check the feasibility of latest arrival times for served passengers. The 'as early as possible' (AEAP) strategy (Sohrabi et al., 2024), vehicles departing as early as possible at each route node, can be applied to determine the earliest pickup time for each passenger. Additionally, the earliest pickup time at the last node is also the earliest arrival time for served requests. If the earliest arrival time exceeds passengers' latest arrival times, the given route has no feasible schedule. Otherwise, a feasible schedule with the earliest arrival time at the last node can be obtained. We refer to this schedule as $\widehat{\mathbb{S}_k(t)}$, which serves as a basis to identify the optimal schedule. We will first introduce a proposition and then explain how to derive the optimal schedule based on $\widehat{\mathbb{S}_k(t)}$ and the proposition.

**Proposition 1.** *For a given feasible schedule $\widehat{\mathbb{S}_k(t)}$, the pickup time can be adjusted in reverse order to minimize the ride time for each node, thus obtaining an optimal schedule $\overrightarrow{\mathbb{S}_k(t)}$.*

**Proof.** See Appendix.

Using the feasible schedule depicted in Fig. 2 as an illustrative example, we can first start from request 4 and transfer its waiting time to its predecessor node, request 3, by postponing the pickup time at request 3. Accordingly, the reduction in ride time for request 3 equals the decrease in waiting time at request 4. Then, we can also postpone the pickup time at request 2 to eliminate waiting time at request 3, without introducing waiting time at request 2. As a result, all the requests achieve the shortest ride time, allowing us to attain an optimal schedule $\overrightarrow{\mathbb{S}_k(t)}$, which corresponds to optimal route schedule 1.

In summary, to determine an optimal schedule for a first-mile ridesharing route $\mathbb{S}_k$, we can first try to derive a feasible schedule $\widehat{\mathbb{S}_k(t)}$ with the AEAP strategy (Sohrabi et al., 2024). If schedule $\widehat{\mathbb{S}_k(t)}$ exists, we can then utilize Proposition 1 to derive an optimal $\overrightarrow{\mathbb{S}_k(t)}$ based on the $\widehat{\mathbb{S}_k(t)}$ schedule by postponing pickup times as much as possible for predecessors of each node.

### 4.4. Forward labeling algorithm

We will introduce the notation for the forward labeling algorithm in Section 4.4.1 and provide a detailed explanation of the customized label extension procedure in Section 4.4.2. Throughout the labeling extension process, the dominance rule plays a crucial role in eliminating unpromising labels. However, this pricing problem involves minimizing detouring and co-riding penalties for riders, which makes the dominance rule used in conventional routing problems inapplicable. Fortunately, we found that the objective value associated with this pricing problem exhibits the characteristics of a piecewise-linear convex function during the labeling extension process. Leveraging this characteristic, we propose exact and heuristic dominance rules in Sections 4.4.3 and 4.4.4, respectively.

#### 4.4.1. Label definition

In this ad-hoc labeling algorithm, we establish a set of notations, as shown in Table 3, to define a label $\mathcal{L}_i^k = \{\mathbb{P}, C, \mathcal{V}, \mathbb{U}, LA, \{DR_n, \Omega_n, \Delta_n, EVT_n, LVT_n, \forall n \in \mathbb{P}\backslash\{g_k\}\}\}$ ending at node $i$ and associated with vehicle $k$.

#### 4.4.2. Label extension and feasibility check

Let $\mathcal{L}_0^k = (\{g_k\}, 0, \emptyset, \emptyset, L_k, \{0\}, \{\varpi_k\}, \{u_k\}, \{r_k\}, \{L_k\})$ be the initial label for car $k$. Then, we can extend this label to all reachable requests. A new label $\mathcal{L}_j^k$ can be obtained by extending label $\mathcal{L}_i^k$ along an arc $(i, j)$ with travel time $t_{ij}$. During the label extension process, we need to update several pieces of information, such as the minimum route duration, the number of riders and the remaining number of co-riders, as well as the departure and remaining ride times of served requests. In addition, it is crucial to verify the feasibility of the new label, which is more complex than the conventional label extension procedure. As a result, we propose a tailored label extension procedure that consists of the following eight steps.

**Step 1:** Partial information update after the label extension along $arc(i,j)$:

$$\mathbb{P}(\mathcal{L}_j^k) = \mathbb{P}(\mathcal{L}_i^k) \cup \{j\} \tag{f1}$$

$$\mathcal{V}(\mathcal{L}_j^k) = \mathcal{V}(\mathcal{L}_i^k) + p_j \tag{f2}$$

$$EVT_j(\mathcal{L}_j^k) = max\left\{EVT_i(\mathcal{L}_i^k) + t_{ij} + s_i, e_j\right\} \tag{f3}$$

$$LVT_j(\mathcal{L}_j^k) = \begin{cases} min\{max\{e_j, LVT_i(\mathcal{L}_i^k) + t_{ij} + s_i\}, a_j - t_{j0} - s_j\} & LVT_i(\mathcal{L}_i^k) > EVT_i(\mathcal{L}_j^k) \\ EVT_j(\mathcal{L}_j^k) & LVT_i(\mathcal{L}_i^k) = EVT_i(\mathcal{L}_i^k) \end{cases} \tag{f4}$$

Eqs. (f1) and (f2) respectively update the visited requests and the occupied capacity for the new label. Eq. (f3) renews the earliest pickup time at node $j$, and it should be the maximum value between the earliest pickup time of request $j$ and the earliest arrival time from node $i$. Eq. (f4) updates the latest pickup time at node $j$ based on two cases: (a) If the earliest pickup time equals the latest pickup time at node $i$, this relationship is maintained at node $j$; (b) Otherwise, $LVT_j(\mathcal{L}_j^k)$ is set to the minimum value between the latest pickup time for request $j$ (which equals the latest arrival time of request $j$ minus the service duration and the direct travel time to the metro station) and the latest arrival time from request $i$.

**Step 2:** Check if the service time window at request $j$ is satisfied, and the extension is infeasible if $EVT_j(\mathcal{L}_j^k) > a_j - t_{j0} - s_j$.

**Step 3:** Starting from the last node, update the pickup time window for predecessors of request $n$:

$$EVT_{n-1}(\mathcal{L}_j^k) = min\left\{EVT_n(\mathcal{L}_j^k) - s_{n-1} - t_{n-1,n}, a_{n-1} - s_{n-1} - t_{n-1,0}\right\} \tag{f5}$$

$$LVT_{n-1}(\mathcal{L}_j^k) = min\left\{LVT_n(\mathcal{L}_j^k) - s_{n-1} - t_{n-1,n}, a_{n-1} - s_{n-1} - t_{n-1,0}\right\} \tag{f6}$$

For each request $n$, the earliest and latest pickup times are updated via Eqs. (f5) and (f6). Using the earliest and latest pickup times identified in Step 1 with the AEAP strategy, Step 3 builds upon Proposition 1 to identify an optimal schedule ($EVT_n$, $\forall n \in \mathbb{P}(\mathcal{L}_j^k)$) by postponing the pickup time of each predecessor node as much as possible. Then, we can efficiently compute the minimum ride time for each node along the partial route in Step 4.

**Step 4:** Update the ride time for each request served by label $\mathcal{L}_j^k$ via Eq. (f7):

$$DR_n(\mathcal{L}_j^k) = EVT_j(\mathcal{L}_j^k) - EVT_n(\mathcal{L}_j^k), \forall n \in \mathbb{P}(\mathcal{L}_j^k) \tag{f7}$$

Note that we can also use the difference in latest pickup times between nodes $j$ and $j$ to compute the minimum ride time for request $n$.

**Step 5:** Update and check the latest arrival time for the new label $\mathcal{L}_j^k$:

The latest arrival time of the new label can be updated by Eq. (f8), and the extension is infeasible if $t_{j0} + s_j + EVT_j(\mathcal{L}_j^k) > min\{LA(\mathcal{L}_i^k), a_j\}$.

$$LA(\mathcal{L}_j^k) = min\{LA(\mathcal{L}_i^k), a_j\} \tag{f8}$$

**Step 6:** Renew the remaining ride time and number of co-riders without penalty cost for each served request:

$$\Delta_n(\mathcal{L}_j^k) = max\{0, u_n - DR_n(\mathcal{L}_j^k)\}, \forall n \in \mathbb{P}(\mathcal{L}_j^k) \tag{f9}$$

$$\Omega_n(\mathcal{L}_j^k) = max\{0, \Omega_n(\mathcal{L}_i^k) - p_j\}, \forall n \in \mathbb{P}(\mathcal{L}_j^k) \tag{f10}$$

**Step 7:** Renew the inextensible nodes for label $\mathcal{L}_j^k$:

Let $\mathbb{U}(\mathcal{L}_j^k) \leftarrow \mathbb{P}(\mathcal{L}_j^k) \cup \mathbb{U}(\mathcal{L}_i^k)$, then the above steps can be utilized to check whether it is feasible to extend label $\mathcal{L}_j^k$ to each node $n \in \mathscr{R} \backslash \mathbb{U}(\mathcal{L}_j^k)$. If the extension is infeasible, the inextensible nodes will be updated: $\mathbb{U}(\mathcal{L}_j^k) \leftarrow \mathbb{U}(\mathcal{L}_j^k) \cup \{n\}$.

**Step 8:** Update the reduced cost of $\mathcal{L}_j^k$ and return the new label $\mathcal{L}_j^k$.

$$\begin{aligned} C(\mathcal{L}_j^k) = & \sum_{n=0}^{|\mathbb{P}(\mathcal{L}_j^k)|-1} c\ d_{\mathbb{P}(\mathcal{L}_j^k)(n),\mathbb{P}(\mathcal{L}_j^k)(n+1)} + f + \sum_{n \in \mathbb{P}(\mathcal{L}_j^k) \cap \{\mathscr{R}_G\}} p_n[\alpha_g(max\{0, \\ & DR_n(\mathcal{L}_j^k) - u_n\}) + \varsigma_g(max\{0, \mathcal{V}(\mathcal{L}_j^k) - p_n - \varpi_n\})] + \sum_{n \in \mathbb{P}(\mathcal{L}_j^k) \cap \{\mathscr{R}_D\}} p_n[\alpha_d(max\{0, \\ & DR_n(\mathcal{L}_j^k) - u_n\}) + \varsigma_d(max\{0, \mathcal{V}(\mathcal{L}_j^k) - p_n - \varpi_n\})] - \sum_{n \in \mathbb{P}(\mathcal{L}_j^k)} \pi_n - \omega_k \end{aligned} \tag{f11}$$

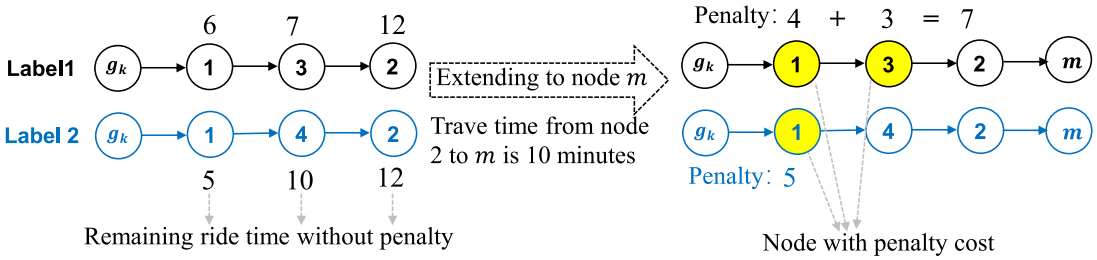



**Fig. 3.** Two different labels, ending with the same node 2, extend to the same node $m$ but result in different cost increases.

#### 4.4.3. Dominance rule

To avoid unnecessary extensions and accelerate the entire labeling algorithm, dominance rules are typically leveraged to eliminate unpromising labels that cannot constitute the optimal solutions. We first give some definitions and introduce Proposition 2. Let $\mathbb{Q}(\mathcal{L}_i^k)$ be the set of feasible extensions of label $\mathcal{L}_i^k$ to the station. Let $\mathcal{L}_i^k \oplus \ell$, $\ell \in \mathbb{Q}(\mathcal{L}_i^k)$ denote a feasible complete path by concatenating $\ell$ to label $\mathcal{L}_i^k$, and $C(\mathcal{L}_i^k \oplus \ell)$ is the resulting reduced cost. Then, the dominance rule concluded by Dabia et al. (2013) can be introduced and presented in Proposition 2.

**Proposition 2.** *For labels $\mathcal{L}_{i1}^k$ and $\mathcal{L}_{i2}^k$ ending at the same node $i$, $\mathcal{L}_{i1}^k$ dominates $\mathcal{L}_{i2}^k$ if conditions (a) and (b) hold.*

$$\mathbb{Q}(\mathcal{L}_{i2}^k) \subseteq \mathbb{Q}(\mathcal{L}_{i1}^k) \tag{a}$$

$$C(\mathcal{L}_{i1}^k \oplus \ell) \leq C(\mathcal{L}_{i2}^k \oplus \ell), \forall \ell \in \mathbb{Q}(\mathcal{L}_{i2}^k) \tag{b}$$

See Dabia et al. (2013) for the proof of Proposition 2. Condition (*a*) means that $\mathcal{L}_{i1}^k$ can generate more complete paths by concatenating them with more feasible extensions to the station. To ensure that condition (*a*) always holds in this pricing problem, we need to guarantee that $\mathcal{L}_{i1}^k$ has more vacant seats, fewer inextensible nodes, a larger latest arrival time, and a smaller earliest pickup time at request $i$. These conditions are ensured by Eqs. (*iii*) to (*vii*), respectively. For conventional shortest-path problems that minimize total transportation costs, the condition (*b*) can be easily ensured when $\mathcal{L}_{i1}^k$ has a smaller reduced cost than $\mathcal{L}_{i2}^k$, namely Eqs. (*i*). However, this cannot always ensure that condition (*b*) holds in this pricing problem, as the same extension cannot guarantee the same increases in penalty cost for $\mathcal{L}_{i1}^k$ and $\mathcal{L}_{i2}^k$. Fig. 3 is an illustrative example that focuses only on the ride-time penalty, with the penalty factor being one dollar per minute, and all requests being general requests. Prior to extending to request $m$, we assume that $C(\mathcal{L}_{i1}^k)$ is less than $C(\mathcal{L}_{i2}^k)$. However, after extending to request $m$, labels $\mathcal{L}_{m1}^k$ and $\mathcal{L}_{m2}^k$ incur \$7 and \$5 penalties, respectively, which may cause $C(\mathcal{L}_{m1}^k)$ to be greater than $C(\mathcal{L}_{m2}^k)$. As a result, additional conditions are necessary to ensure that condition (*b*) always holds. The simplest way is to check that each request served by $\mathcal{L}_{i1}^k$ and $\mathcal{L}_{i2}^k$ has a longer remaining ride time and a larger remaining number of co-riders in $\mathcal{L}_{i1}^k$, as shown in inequality (*ii*). Consequently, conditions (*i*)–(*vii*) need to be checked to establish the dominance relationship between labels $\mathcal{L}_{i1}^k$ and $\mathcal{L}_{i2}^k$.

Lower reduced cost: $C(\mathcal{L}_{i1}^k) \leq C(\mathcal{L}_{i2}^k)$ (*i*)

$\Omega_n(\mathcal{L}_{i1}^k) \geq \Omega_n(\mathcal{L}_{i2}^k), and\ \Delta_n(\mathcal{L}_{i1}^k) \geq \Delta_n(\mathcal{L}_{i2}^k), \forall n \in \mathbb{P}(\mathcal{L}_{i1}^k) \cup \mathbb{P}(\mathcal{L}_{i2}^k)$ (*ii*)

Fewer occupied seats: $\mathcal{V}(\mathcal{L}_{i1}^k) \leq \mathcal{V}(\mathcal{L}_{i2}^k)$ (*iii*)

Fewer inextensible nodes: $\mathbb{U}(\mathcal{L}_{i1}^k) \leq \mathbb{U}(\mathcal{L}_{i2}^k)$ (*iv*)

Larger latest arrival time: $LA(\mathcal{L}_{i1}^k) \geq LA(\mathcal{L}_{i2}^k)$ (*v*)

Smaller earliest pickup time at request $j$: $EVT_j(\mathcal{L}_{i1}^k) \leq EVT_j(\mathcal{L}_{i2}^k)$ (*vi*)

Larger latest pickup time at request $j$: $LVT_j(\mathcal{L}_{i1}^k) \geq LVT_j(\mathcal{L}_{i2}^k)$ (*vii*)

**Dominance rule 1:** If inequalities (*i*)–(*vii*) hold and at least one is strict, then $\mathcal{L}_{i1}^k$ dominates $\mathcal{L}_{i2}^k$.

However, this dominance rule is not strong for eliminating unpromising labels since condition (*ii*) is weak. Accordingly, we propose an alternative dominance rule.

Let $\varphi_{\mathcal{L}_i^k}(u)$ and $\psi_{\mathcal{L}_i^k}(v)$ be penalty costs for requests served by $\mathcal{L}_i^k$ when $u$ riders and $v$ minutes ride time are added to $\mathcal{L}_i^k$, respectively. Then, we can ensure that condition (*b*) in Proposition 2 always holds with inequality (*c*) and (*i*).

$$\begin{aligned}&\varphi_{\mathcal{L}_{i1}^k}(u) \leq \varphi_{\mathcal{L}_{i2}^k}(u), \psi_{\mathcal{L}_{i1}^k}(v) \leq \psi_{\mathcal{L}_{i2}^k}(v), \forall u \in \mathbb{Z}^+, u \leq q_k - \mathcal{V}(\mathcal{L}_{i2}^k),\\ &0 < v \leq min\{LA(\mathcal{L}_{i1}^k) - EVT_j(\mathcal{L}_{i1}^k), LA(\mathcal{L}_{i2}^k) - EVT_j(\mathcal{L}_{i2}^k)\}\end{aligned} \tag{c}$$

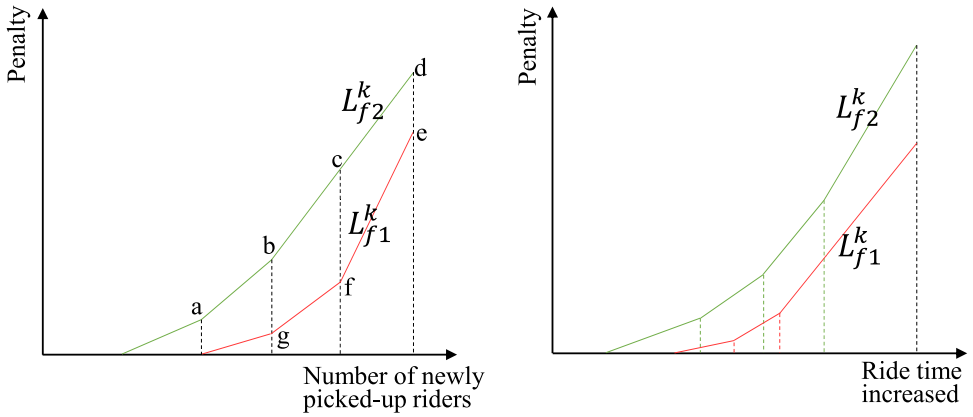



**Fig. 4.** Visualizations of functions $\varphi_{\mathcal{L}_i^k}(u)$ and $\psi_{\mathcal{L}_i^k}(v)$.

For each label $\mathcal{L}_i^k$, the penalty cost associated with ride time (or co-riding) for served requests follows a piecewise linear convex function pattern as the ride time (or the number of co-riders) increases during the label extension process, as shown in Fig. 4. Given the characteristics of the piecewise linear convex function, the inequality (*c*) can be met if the penalty of $\mathcal{L}_{i1}^k$ is always less than or equal to that of $\mathcal{L}_{i2}^k$ at each inflection point (such as a, b, c, d, e, f, and g in Fig. 4(a)) of the penalty functions. These inflection points are all attributed to the number of newly picked-up riders exceeding the remaining co-riding willingness of one or several requests. Accordingly, we just need to ensure that the penalty of $\mathcal{L}_{i1}^k$ is less than or equal to that of $\mathcal{L}_{i2}^k$ when the number of newly picked-up riders equals the remaining number of co-riders for each request in Labels $\mathcal{L}_{i1}^k$ and $\mathcal{L}_{i2}^k$, as well as when the number of newly picked-up riders equals the difference between the vehicle capacity minus the total number of on-board riders. Likewise, the penalty corresponding to the ride time can be treated in the same way. This simplifies the process of checking the inequality (*c*).

**Dominance rule 2:** If inequalities (*i*), (*c*), and (*iii*)–(*vii*) hold and at least one is strict, then $\mathcal{L}_{i1}^k$ dominates $\mathcal{L}_{i2}^k$.

#### 4.4.4. Two heuristic dominance rules

Even if dominance rule 2 can eliminate more unpromising labels, it is still not remarkably strong. To accelerate the computation speed of the forward labeling algorithm, we introduce two heuristic dominance rules.

**Heuristic dominance rule 1:** If inequalities (*i*),(*iii*), and (*v*)–(*vii*) hold, then $\mathcal{L}_{i1}^k$ dominates $\mathcal{L}_{i2}^k$.

As inequalities (*ii*) and (*iv*) are too strict in dominance rule 1, they can be removed to obtain a relaxed dominance rule, which helps eliminate more labels and therefore improves computational efficiency. Before presenting another heuristic dominance rule, we first give four new definitions.

- $\Omega(\mathcal{L}_i^k)$: remaining number of co-riders without co-riding penalty cost for label $\mathcal{L}_i^k$;
- $\Omega^d(\mathcal{L}_i^k)$: remaining number of co-riders without co-riding penalty cost for high-priority requests on label $\mathcal{L}_i^k$;
- $\Delta(\mathcal{L}_i^k)$: remaining ride time without exceeding detour tolerances for requests of label $\mathcal{L}_i^k$;
- $\Delta^d(\mathcal{L}_i^k)$: remaining ride time without exceeding detour tolerances for high-priority requests of label $\mathcal{L}_i^k$;

**Heuristic dominance rule 2:** If inequalities (*i*), (*iii*)–(*vii*), and (*d*) hold, then $\mathcal{L}_{i1}^k$ dominates $\mathcal{L}_{i2}^k$.

$$\Omega(\mathcal{L}_{i1}^k) \geq \Omega(\mathcal{L}_{i2}^k), \Omega^d(\mathcal{L}_{i1}^k) \geq \Omega^d(\mathcal{L}_{i2}^k), \Delta(\mathcal{L}_{i1}^k) \geq \Delta(\mathcal{L}_{i2}^k), \text{and } \Delta^d(\mathcal{L}_{i1}^k) \geq \Delta^d(\mathcal{L}_{i2}^k). \quad (d)$$

The inequality (*d*) relaxes (*ii*) since (*d*) focuses only on the minimum remaining ride time and remaining number of co-riders for general and high-priority requests on the label, which also helps to find high-quality routes and facilitates a reduction in computation time for the labeling algorithm. It is worth noting that these heuristic dominance rules may eliminate some promising labels that constitute the optimal solution. Therefore, the exact dominance rule will be applied to find trips with negative reduced costs when heuristic dominance rules do not work.

### 4.5. Backward labeling algorithm

In the backward labeling algorithm, labels originate from the station and extend along reachable arcs to generate new labels. Unlike the forward labeling algorithm, backward labels do not need to be associated with a specific car, so we can use $\mathcal{L}_i$ to denote a backward label corresponding to a partial path starting from the station and ending at node $i$. As with the forward labeling algorithm, we give the following definitions, as shown in Table 4, for a backward label $\mathcal{L}_i$. For any $\mathcal{L}_i$, the remaining ride times for requests served by $\mathcal{L}_i$ will not change during the backward extension process. As a result, there is no need to use the notation $\Delta_n(\mathcal{L}_i)$ defined in the forward labels. In addition, there is no need to update the earliest and latest pickup times for each served request, so we only keep track of the earliest and latest pickup times at the end node $i$.

**Table 4**
Notations for defining backward labels.

| | |
|---|---|
| $\mathbb{P}(\mathcal{L}_i)$ | Set of path nodes visited by label $\mathcal{L}_i$; |
| $C(\mathcal{L}_i)$ | Reduced cost of the partial path $\mathbb{P}(\mathcal{L}_i)$; |
| $\mathcal{V}(\mathcal{L}_i)$ | Number of riders after visiting node $i$; |
| $\mathbb{U}(\mathcal{L}_i)$ | Vector of visited requests and inextensible requests for this label; |
| $LA(\mathcal{L}_i)$ | Latest arrival time for all requests served by this partial path; |
| $DR_n(\mathcal{L}_i)$ | Minimum trip duration from node $n$ to the station along nodes of label $\mathcal{L}_i$; |
| $\Omega_n(\mathcal{L}_i)$ | Remaining number of co-riders without incurring the co-riding penalty for request $n$ served by $\mathcal{L}_i$; |
| $EVT(\mathcal{L}_i)$ | Earliest pickup time at node $i$; |
| $LVT(\mathcal{L}_i)$ | Latest pickup time at node $i$. |

Let $\mathcal{L}_0 = (\{0\}, 0, 0, \emptyset, T, \{0\}, \{P\}, 0, T)$ be the initial backward label. Then, we can extend this label to all reachable requests. Given a backward label $\mathcal{L}_i$, it can be extended along a feasible arc $(j, i)$ using the following extension function. Before proceeding with the extension, we need to check whether two inequalities are met: $e_j + s_j + t_{ji} \le LVT(\mathcal{L}_i)\ and\ EVT(\mathcal{L}_i) \ge ResourceBound$. If these conditions are not satisfied, the extension is infeasible.

$$\mathbb{P}(\mathcal{L}_j) = \mathbb{P}(\mathcal{L}_i) \cup \{j\} \tag{b1}$$

$$\mathcal{V}(\mathcal{L}_j) = \mathcal{V}(\mathcal{L}_i) + p_j \tag{b2}$$

$$EVT(\mathcal{L}_j) = max\left\{EVT(\mathcal{L}_i) - t_{ji} - s_j, e_j\right\} \tag{b3}$$

$$LVT(\mathcal{L}_j) = min\{max\{LVT(\mathcal{L}_i) - t_{ji} - s_j, e_j\}, a_j - t_{j0} - s_j\} \tag{b4}$$

$$LA(\mathcal{L}_j) = min\{LA(\mathcal{L}_i), a_j\} \tag{b5}$$

$$DR_j(\mathcal{L}_j) = DR_i(\mathcal{L}_i) + t_{ji} + s_j + max\{0, EVT(\mathcal{L}_i) - t_{ji} - s_j - (a_j - t_{j0} - s_j)\} \tag{b6}$$

Note: The extension is infeasible if $DR_j(\mathcal{L}_j) + EVT(\mathcal{L}_j) > a_j$.

$$\Omega_n(\mathcal{L}_j) = max\{0, \Omega_n(\mathcal{L}_i) - p_j\}, \forall n \in \mathbb{P}(\mathcal{L}_j)\backslash\{0\} \tag{b7}$$

$$\begin{aligned} C(\mathcal{L}_j) = &\sum_{n=0}^{|\mathbb{P}(\mathcal{L}_j)|-1} cd_{\mathbb{P}(\mathcal{L}_j)(n),\mathbb{P}(\mathcal{L}_j)(n+1)} + \sum_{n\in\mathbb{P}(\mathcal{L}_j)\cap\{\mathscr{R}_G\}} p_n[\alpha_g(max\{0, DR_n(\mathcal{L}_j) \\ &-u_n\}) + \varsigma_g(max\{0, \mathcal{V}(\mathcal{L}_j) - p_n - \varpi_n\})] + \sum_{n\in\mathbb{P}(\mathcal{L}_j)\cap\{\mathscr{R}_D\}} p_n[\alpha_d(max\{0, DR_n(\mathcal{L}_j) \\ &-u_n\}) + \varsigma_d(max\{0, \mathcal{V}(\mathcal{L}_j) - p_n - \varpi_n\})] - \sum_{n\in\mathbb{P}(\mathcal{L}_j)} \pi_n \end{aligned} \tag{b8}$$

Let $\mathbb{U}(\mathcal{L}_j) \leftarrow \mathbb{P}(\mathcal{L}_j) \cup \mathbb{U}(\mathcal{L}_i)$, then the above steps can be utilized to check whether it is feasible to extend label $\mathcal{L}_j$ to each node $n \in \mathscr{R}\backslash\mathbb{U}(\mathcal{L}_i)$. If the extension is infeasible, the inextensible nodes will be updated: $\mathbb{U}(\mathcal{L}_j) \leftarrow \mathbb{U}(\mathcal{L}_i) \cup \{n\}$. We also give the following definitions for using the heuristic dominance rule 2.

- $\Omega(\mathcal{L}_j)$: remaining number of co-riders without co-riding penalty cost for label $\mathcal{L}_j$; and
- $\Omega^d(\mathcal{L}_j)$: remaining number of co-riders without co-riding penalty cost for high-priority requests on label $\mathcal{L}_j$. $\Omega(\mathcal{L}_j)$ and $\Omega^d(\mathcal{L}_j)$ can be updated via Eqs. (b9) and (b10).

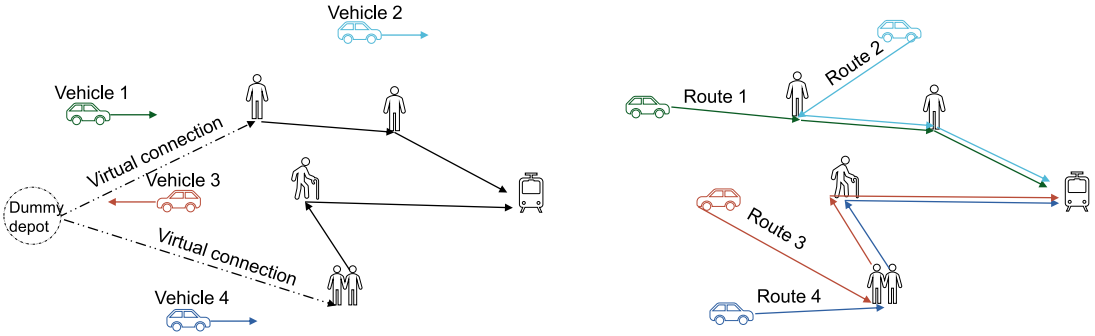
$$\Omega(\mathcal{L}_j) = min\{\Omega_n(\mathcal{L}_j), \forall n \in \mathbb{P}(\mathcal{L}_j)\backslash\{0\}\} \tag{b9}$$

$$\Omega^d(\mathcal{L}_j) = min\{\Omega_n(\mathcal{L}_j), \forall n \in \mathbb{P}(\mathcal{L}_j)\backslash\{0\} \cap \{\mathscr{R}_D\}\} \tag{b10}$$

The forward dominance rules can be directly applied to the backward labeling algorithm.

### 4.6. Merging forward and backward labels

When all forward and backward labels are generated, feasible routes can be constructed by merging the forward and backward labels. A forward label $\mathcal{L}_i^k$ and a backward label $\mathcal{L}_i$, both ending at the same node $i$, can be joined if the following conditions are satisfied:




(a) Generate routes for a virtual vehicle in a dummy depot via the labeling algorithm

(b) Match vehicles and routes to derive feasible trips for each vehicle

**Fig. 5.** Illustration of the "label-vehicle matching" strategy.

- $\mathcal{V}(\mathcal{L}_i^k) + \mathcal{V}(\mathcal{L}_i) - p_i \leq q_k$,
- $EVT_i(\mathcal{L}_i^k) \leq LVT(\mathcal{L}_i)$, and
- $EVT_i(\mathcal{L}_i^k) + DR_i(\mathcal{L}_i) + max\{0, EVT(\mathcal{L}_i) - LVT_i(\mathcal{L}_i^k)\} \leq LA(\mathcal{L}_i^k)$.

The first inequality ensures feasibility regarding vehicle capacity. The second inequality means that the earliest pickup time at node $i$ for the forward label $\mathcal{L}_i^k$ is less than the latest pickup time at node $i$ for the backward label $\mathcal{L}_i$. Otherwise, the merging is infeasible. The third inequality guarantees that each request's latest arrival time is met. The resulting label $\mathcal{L} = \mathcal{L}_i^k \oplus \mathcal{L}_i$ has the following attributes:

- $\mathbb{P}(\mathcal{L}) = \mathbb{P}(\mathcal{L}_i^k) \cup \mathbb{P}(\mathcal{L}_i)$,
- $\mathcal{V}(\mathcal{L}) = \mathcal{V}(\mathcal{L}_i^k) + \mathcal{V}(\mathcal{L}_i) - p_i$,
- $DR_n(\mathcal{L}) = DR_n(\mathcal{L}_i^k) + DR_i(\mathcal{L}_i) + max\{0, EVT(\mathcal{L}_i) - LVT_i(\mathcal{L}_i^k)\}, \forall n \in \mathbb{P}(\mathcal{L}_i^k)$, and
- $C(\mathcal{L})$, which can be obtained via Eq. (*f* 11). If the accumulated reduced cost $C(\mathcal{L})$ for the new label $\mathcal{L}$ is negative, then the corresponding route can be added to be the RMP route set. Otherwise, label $\mathcal{L}$ will be discarded.

### *4.7. Column generation math-heuristic algorithm*

To obtain near-optimal solutions for the MP model efficiently, we introduce a column-generation math-heuristic algorithm with a combined procedure for solving the PSP, which aims to reduce the number of times employing the exact labeling algorithm. First, we can utilize the heuristic labeling algorithm to generate a set of labels starting from a dummy depot with a dummy car. Then, we match the set of labels with each vehicle to generate a set of trips, from which we select the one with the most negative reduced cost as the best trip for the corresponding vehicle. We refer to this strategy as "label-vehicle matching". As shown in Fig. 5, we can use the labeling algorithm just once to generate feasible trips for multiple vehicles. Consequently, this strategy is highly efficient for reducing total computation time, especially when the fleet is large. Accordingly, we can define the following labeling algorithms.

- Labeling algorithm 1 (LA1): employing the bidirectional labeling algorithm with heuristic dominance rule 1 and the "label-vehicle matching" strategy to generate labels.
- Labeling algorithm 2 (LA2): leveraging the labeling algorithm with heuristic dominance rule 1 to obtain labels for each vehicle, respectively.
- Labeling algorithm 3 (LA3): leveraging the labeling algorithm with heuristic dominance rule 2 to determine labels for each vehicle, respectively.
- Labeling algorithm 4 (LA4): applying the labeling algorithm with the exact dominance rule to find labels for each vehicle, respectively.

During the process of iteratively solving the RMP, the PSP procedure is also executed repeatedly. If a labeling algorithm fails to find any trips with a negative reduced cost, then it can be skipped in the subsequent process of solving the PSP. To accomplish this, a *Boolean* flag can be assigned to each labeling algorithm, which is initialized to *true* and set to *false* when the algorithm cannot identify trips with negative costs. When all labeling algorithms fail to find columns with negative reduced costs, we solve the MP model based on these columns to obtain an high-quality integer solution (Desaulniers et al., 2006; Joncour et al., 2010). With these preparations, the combined procedure for solving the PSP problem and the column-generation algorithm can be depicted in the flowchart shown in Fig. 6.

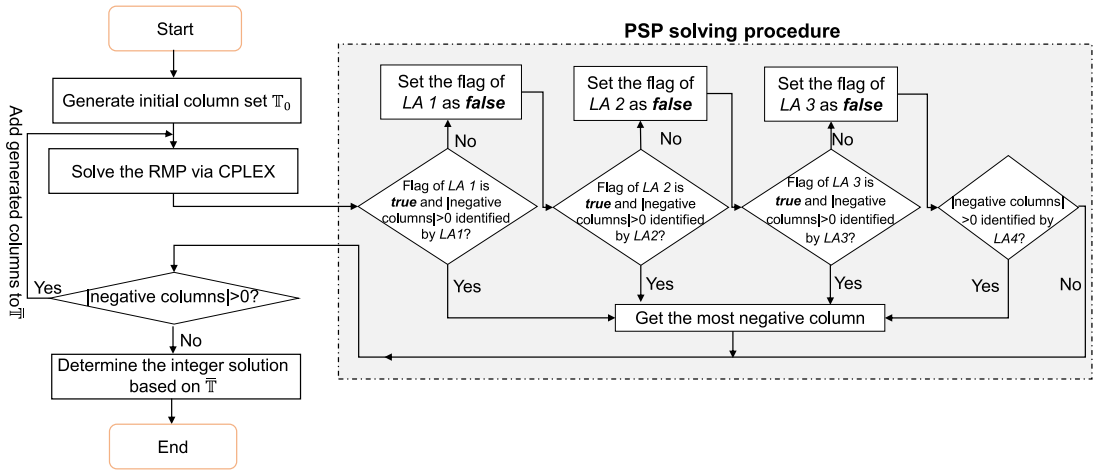



**Fig. 6.** Flowchart of the PSP solving procedure and the column-generation math-heuristic algorithm.

### 4.8. Branch-and-price algorithm

The column-generation procedure can be used to solve the linear programming (LP) relaxation of the restricted master problem, but the solution obtained may not be an integer solution. In these cases, the branch-and-bound algorithm can be utilized to search for an integer solution. The branch-and-bound tree is explored with the best-first search strategy, whereby the node with the lowest lower bound will be explored first. Once the fractional solution is found at a tree node, we will branch on the arc flow variables. First, we need to determine the values of each arc flow variable $x_{ij} = \sum_{k \in \mathcal{K}} \sum_{\tau_k \in \bar{\mathbb{T}}_k} \chi_{ij}^{\tau_k} \delta_{\tau_k}$. Then, we select the $arc\,(i,j) \leftarrow argmin|x_{ij} - 0.5|$ and generate two child nodes with imposed constraints $x_{ij} = 0$ and $x_{ij} = 1$, respectively. Accordingly, the directed graph $G(\mathcal{N}, \mathcal{A})$ and the columns of the restricted master problem will be modified. The details are as follows:

• Impose the constraint $x_{ij} = 0$ to one child node.

At this node, $arc\,(i,j)$ is forbidden in the subproblem and the master problem. Therefore, the distance of $arc\,(i,j)$ can be set to positive infinity, and the columns containing this arc are removed from the master problem.

• Impose the constraint $x_{ij} = 1$ to another child node.

At this node, the successor of node $i$ must be node $j$. Consequently, the distances of $arcs\,\{(i,k)\,|k \neq j\}$ and $arcs\,\{(m,j)\,|m \neq i\}$ can be set to positive infinity, and the columns that contain node $i$ without the successor node $j$ and contain node $j$ without the predecessor node $i$ are deleted from the master problem.

## 5. Numerical experiments

This section aims to evaluate the performance of the proposed column generation math-heuristic (CGM) algorithm and to analyze the impacts of incorporating service quality preferences into the first-mile ridesharing problem. We implement these algorithms in C++ and utilize CPLEX 12.8 to solve the arc-based formulation, the RMP model, and the MP model. All the experiments are conducted on a laptop equipped with a 3.0 GHz AMD Ryzen5 4600H processor and 16 GB of RAM. The maximum running time of the CPLEX solver and the branch-and-price algorithm is set to 7200 s.

### 5.1. Instances generation and parameters setting

We conduct experiments on randomly generated instances based on the road network within a 5 km radius of the Xinzhuang metro station in Shanghai, along with randomly generated requests and vehicles. We set the ratio of general requests and high-priority requests to 2:1, and the probabilities of having 1 or 2 riders per request are 75% and 25%, respectively. We assume that the agency is designing ridesharing schemes for requests whose earliest pickup times are within the time interval [8:00, 9:00]. As a result, the earliest pickup time for each request is randomly generated within this time interval. The latest arrival time and the maximum ride time of each request are also randomly generated based on the direct travel time to the station and the specified earliest pickup time. The maximum number of co-riders for each request is also randomly generated from 2, 3, or 4. Each vehicle has either four or six passenger seats. Other parameters are summarized in Table 5.

**Table 5**
Value of parameters.

| Parameter | $f_k$ | $c$ | $s_i$ | $\varsigma_g$ | $\varsigma_d$ | $\alpha_g$ | $\alpha_d$ | $v$ |
|---|---|---|---|---|---|---|---|---|
| Value | \$3 | 0.5 \$/km | 1 min | \$6 | \$12 | 6 \$/min | 12 \$/min | 25 km/hour |

### 5.2. Computational performance

To evaluate the computational performance of the CGM algorithm, we compare their computational results with those obtained by the CPLEX solver and the branch-and-price exact algorithm on two groups of instances: small-scale and large-scale instances. Each group consists of two sets of instances, with set 1 having relatively insufficient vehicles available and set 2 having sufficient vehicles available. As shown in Tables 6 and 7, each instance is named with the format "R_V", where 'R' represents the number of requests and 'V' denotes the number of available vehicles. For example, the instance named '6_2' in Table 6 indicates that there are six requests requiring first-mile services and two vehicles available to provide first-mile ridesharing services. All the computational results are reported in Tables 6 and 7.

#### 5.2.1. Small-scale instances

The experiments are conducted on two sets of small-scale instances, and the results are reported in Table 6. The 'Obj' column shows the objective value of the best integer solution identified by the corresponding solution method, while the 'Time' column reports the computation time spent in finding the solution. The 'Gap' column represents the relative difference between the lower bound and the upper bound found by the CPLEX solver. 'Gap1', 'Gap2', and 'Gap3' denote the difference between the objective values of integer solutions obtained by the solver, the branch-and-price algorithm, and the CGM algorithm. As shown in this table, CPLEX cannot obtain the optimal integer solution for some small-scale instances within 7200 s. The branch-and-price and column-generation algorithms are capable of yielding solutions that are equal to or superior to those found by CPLEX within three seconds. Furthermore, the CGM algorithm finds exact solutions for most small-scale instances, with only four exceptions, when compared to the solutions obtained by the solver and the branch-and-price algorithm. With respect to computation time, the CGM algorithm outperforms the other two methods, with an average computation time of less than one second for all instances.

#### 5.2.2. Large-scale instances

To evaluate the computational performance of the CGM algorithm on large-scale instances, we compare its computational results with those obtained by the CGM algorithm with the monodirectional labeling procedure, as well as those generated by the branch-and-price algorithm. The results are reported in Table 7, from which two conclusions can be drawn. In terms of computation time, both CGM algorithms significantly outperform the exact algorithm. Additionally, the CGM algorithm is faster than the CGM algorithm with monodirectional labeling, as it can obtain a near-optimal solution within an average computation time of less than 11 s, whereas the CGM with monodirectional labeling requires 14 s. Regarding solution quality, the CGM algorithm can find a high-quality integer solution for all instances, with an average optimality gap of 0.61% compared to the exact solution. These results demonstrate that the proposed CGM procedure can achieve near-optimal solutions within a short computation time for this problem.

### 5.3. Sensitivity analysis on penalty factors

To investigate the impact of penalty coefficients on service quality, we conduct experiments with multiple penalty coefficient values. Specifically, we take the penalty coefficients for ride time and co-riding between 0 and 6. These penalty factors are for general passengers, and we set the penalty factor for high-priority riders at twice that of general passengers. Then, we record the service quality under different penalty coefficients, and the results are presented in Fig. 7, which shows the number of requests with unmet service quality under different combinations of penalty factors for ride time and co-riding. The results indicate that even a small penalty factor can significantly enhance service quality. For instance, when both penalty factors are increased from 1 to 2, the number of requests whose service quality is not met decreases by 73.08%.

### 5.4. Sensitivity analysis on the ratio of single-passenger to two-passenger requests

To investigate the impact of different ratios of single-passenger to two-passenger requests on the operational costs and computational efficiency of ridesharing services, we select six large-scale instances: 30_10, 40_13, and 50_15 from set 1, and 30_15, 40_20, and 50_25 from set 2. Then, we conduct numerical experiments for each instance, setting the ratios of single-passenger to two-passenger requests at 1:1, 2:1, 3:1, 4:1, and 5:1. The experimental results are shown in Fig. 8, where 'CT' represents the computation time and 'Obj' represents the value of the objective function.

According to Fig. 8, the average computation time for all instances in different ratios is 21.3 s, with the longest being 57.1 s. Additionally, different ratios have minor impacts on computation time for the same instance. Regarding the average operational cost, it decreases from \$210.5 to \$169.04 as the ratio of requests with one passenger increases. This is because a higher ratio of single-passenger requests allows each vehicle to serve more requests, reducing the number of employed vehicles and thus the average operational cost.

**Table 6**
Computational performances for small-scale instances.

| Instances | | CPLEX solver | | | Branch-and-Price | | CGM | | $Gap^{Algorithm}$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Obj | Time | Gap | Obj | Time | Obj | Time | Gap1 | Gap2 | Gap3 |
| (1) | | (2) | (3) | (4) | (5) | (6) | (7) | (8) | (9) | (10) | (11) |
| Set 1 | 6_2 | 223.40 | 0.1 | 0.00 | 223.40 | 0.1 | 223.40 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 7_2 | 233.40 | 0.1 | 0.00 | 233.40 | 0.1 | 233.40 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 8_2 | 363.85 | 0.1 | 0.00 | 363.85 | 0.1 | 363.85 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 9_3 | 214.13 | 0.3 | 0.00 | 214.13 | 0.1 | 214.13 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 10_3 | 224.13 | 0.4 | 0.00 | 224.13 | 0.1 | 224.13 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 11_3 | 244.13 | 0.7 | 0.00 | 244.13 | 0.1 | 244.13 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 12_4 | 282.35 | 5.4 | 0.00 | 282.35 | 0.2 | 282.35 | 0.2 | 0.00 | 0.00 | 0.00 |
| | 13_4 | 292.35 | 11.0 | 0.00 | 292.35 | 0.2 | 292.35 | 0.3 | 0.00 | 0.00 | 0.00 |
| | 14_4 | 368.79 | 104.9 | 0.00 | 368.79 | 0.3 | 368.79 | 0.4 | 0.00 | 0.00 | 0.00 |
| | 15_5 | 129.44 | 121.9 | 0.00 | 129.44 | 0.3 | 129.44 | 0.3 | 0.00 | 0.00 | 0.00 |
| | 16_5 | 139.44 | 1602.1 | 0.00 | 139.44 | 0.3 | 139.44 | 0.4 | 0.00 | 0.00 | 0.00 |
| | 17_5 | 141.81 | 3531.1 | 0.00 | 141.81 | 0.3 | 141.81 | 0.5 | 0.00 | 0.00 | 0.00 |
| | 18_6 | 164.47 | 7233.7 | 67.34 | 164.47 | 0.4 | 164.47 | 0.5 | 0.00 | 0.00 | 0.00 |
| | 19_6 | 173.44 | 7232.2 | 66.74 | 173.44 | 0.5 | 173.44 | 0.6 | 0.00 | 0.00 | 0.00 |
| | 20_6 | 183.44 | 7235.6 | 60.69 | 183.44 | 0.4 | 183.44 | 0.7 | 0.00 | 0.00 | 0.00 |
| | **Average** | **237.25** | **1805.3** | **12.98** | **225.24** | **0.2** | **225.24** | **0.3** | **0.00** | **0.00** | **0.00** |
| Set 2 | 6_6 | 47.02 | 0.1 | 0.00 | 47.02 | 0.1 | 47.02 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 7_6 | 57.02 | 0.2 | 0.00 | 57.02 | 0.1 | 57.02 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 8_6 | 57.51 | 0.2 | 0.00 | 57.51 | 0.1 | 57.51 | 0.1 | 0.00 | 0.00 | 0.00 |
| | 9_7 | 46.30 | 1.1 | 0.00 | 46.30 | 0.1 | 46.30 | 0.2 | 0.00 | 0.00 | 0.00 |
| | 10_7 | 54.99 | 0.9 | 0.00 | 54.99 | 0.2 | 54.99 | 0.2 | 0.00 | 0.00 | 0.00 |
| | 11_7 | 74.99 | 3.5 | 0.00 | 74.99 | 0.2 | 74.99 | 0.2 | 0.00 | 0.00 | 0.00 |
| | 12_8 | 82.86 | 35.9 | 0.00 | 82.86 | 0.2 | 82.86 | 0.3 | 0.00 | 0.00 | 0.00 |
| | 13_8 | 84.46 | 111.1 | 0.00 | 84.46 | 0.3 | 84.46 | 0.4 | 0.00 | 0.00 | 0.00 |
| | 14_8 | 87.38 | 184.0 | 0.00 | 87.38 | 0.4 | 87.38 | 0.5 | 0.00 | 0.00 | 0.00 |
| | 15_9 | 78.24 | 7201.6 | 21.12 | 78.24 | 0.4 | 78.24 | 0.6 | 0.00 | 0.00 | 0.00 |
| | 16_9 | 88.24 | 3021.4 | 0.00 | 88.24 | 2.4 | 89.08 | 0.8 | 0.00 | 0.95 | 0.95 |
| | 17_9 | 97.99 | 7292.4 | 60.36 | 94.43 | 3.4 | 95.62 | 0.7 | −3.63 | −2.42 | 1.26 |
| | 18_10 | 101.65 | 7232.8 | 61.68 | 101.65 | 14.3 | 102.56 | 0.7 | 0.00 | 0.90 | 0.90 |
| | 19_10 | 109.21 | 7216.5 | 70.23 | 102.74 | 7.5 | 105.90 | 0.8 | −5.92 | −3.03 | 3.08 |
| | 20_10 | 112.74 | 7243.7 | 65.85 | 111.54 | 3.6 | 111.54 | 0.9 | −1.06 | −1.06 | 0.00 |
| | **Average** | **78.71** | **2636.4** | **18.62** | **77.96** | **2.2** | **78.36** | **0.4** | **−0.71** | **−0.31** | **0.41** |

*Gap1 = [(5)–(2)]/(2) × 100%; Gap2 = [(7)–(2)]/(2) × 100%; Gap3 = [(7)–(5)]/(5) × 100%.


**Fig. 7.** Results for sensitivity analysis on penalty cost factors.

**Table 7**
Computational performances for large-scale instances.

| Instances | | Branch-and-Price | | $CGM^{Mono}$ | | CGM | | $Gap^{CGM}$ | |
|---|---|---|---|---|---|---|---|---|---|
| | | Obj | Time | Obj | Time | Obj | Time | Gap1 | Gap2 |
| (1) | | (2) | (3) | (4) | (5) | (6) | (7) | (8) | (9) |
| Set 1 | 20_8 | 102.08 | 11.1 | 102.08 | 0.9 | 102.08 | 1.1 | 0.00 | 0.00 |
| | 22_8 | 117.27 | 16.8 | 120.09 | 1.1 | 121.23 | 1.5 | 2.40 | 3.38 |
| | 24_9 | 113.55 | 1.4 | 113.55 | 1.5 | 113.55 | 1.8 | 0.00 | 0.00 |
| | 26_9 | 142.22 | 248.7 | 142.22 | 2.5 | 142.22 | 1.8 | 0.00 | 0.00 |
| | 28_10 | 154.76 | 320.3 | 158.81 | 3.5 | 154.76 | 2.9 | 2.62 | 0.00 |
| | 30_10 | 167.48 | 182.3 | 169.30 | 3.8 | 169.30 | 2.6 | 1.09 | 1.09 |
| | 32_11 | 163.62 | 1579.3 | 164.18 | 6.2 | 163.86 | 5.3 | 0.34 | 0.15 |
| | 34_11 | 179.51 | 1893.2 | 180.90 | 6.2 | 180.19 | 5.5 | 0.77 | 0.38 |
| | 36_12 | 194.52 | 120.2 | 203.91 | 8.2 | 198.92 | 6.6 | 4.83 | 2.26 |
| | 38_12 | 220.46 | 821.8 | 223.09 | 15.0 | 223.09 | 9.7 | 1.19 | 1.19 |
| | 40_13 | 209.69 | 1236.7 | 209.69 | 12.0 | 209.69 | 12.5 | 0.00 | 0.00 |
| | 42_13 | 232.73 | 54.9 | 232.73 | 18.3 | 232.73 | 14.3 | 0.00 | 0.00 |
| | 44_14 | 247.27 | 139.7 | 247.27 | 21.8 | 247.27 | 20.9 | 0.00 | 0.00 |
| | 46_14 | 273.09 | 1582.4 | 274.00 | 41.1 | 276.24 | 21.3 | 0.33 | 1.15 |
| | 48_15 | 255.04 | 147.1 | 255.04 | 34.0 | 255.04 | 29.1 | 0.00 | 0.00 |
| | 50_15 | 268.47 | 321.2 | 269.08 | 56.9 | 268.47 | 38.2 | 0.23 | 0.00 |
| | **Average** | **190.11** | **542.3** | **191.62** | **14.6** | **191.17** | **10.9** | **0.86** | **0.60** |
| Set 2 | 20_10 | 86.94 | 7.8 | 86.94 | 1.0 | 86.94 | 1.4 | 0.00 | 0.00 |
| | 22_11 | 85.90 | 20.1 | 85.90 | 1.1 | 85.90 | 1.4 | 0.00 | 0.00 |
| | 24_12 | 96.00 | 37.9 | 96.00 | 1.6 | 96.00 | 2.2 | 0.00 | 0.00 |
| | 26_13 | 101.29 | 35.0 | 102.08 | 2.1 | 102.10 | 2.4 | 0.78 | 0.80 |
| | 28_14 | 108.11 | 52.3 | 108.90 | 3.3 | 109.86 | 2.9 | 0.73 | 1.62 |
| | 30_15 | 106.07 | 681.3 | 106.89 | 3.8 | 106.86 | 2.7 | 0.77 | 0.74 |
| | 32_16 | 118.76 | 389.2 | 121.40 | 5.1 | 118.76 | 4.4 | 2.22 | 0.00 |
| | 34_17 | 118.87 | 3323.3 | 118.95 | 6.6 | 120.54 | 4.5 | 0.07 | 1.40 |
| | 36_18 | 123.48 | 126.8 | 126.73 | 9.2 | 123.48 | 7.9 | 2.63 | 0.00 |
| | 38_19 | 120.16 | 6099.7 | 120.16 | 8.0 | 120.89 | 10.1 | 0.00 | 0.61 |
| | 40_20 | 128.05 | 7056.7 | 129.20 | 14.5 | 128.19 | 10.3 | 0.90 | 0.11 |
| | 42_21 | 130.58 | 7207.8 | 133.28 | 23.6 | 133.70 | 20.0 | 2.07 | 2.39 |
| | 44_22 | 138.32 | 1023.2 | 141.98 | 26.5 | 139.89 | 21.9 | 2.65 | 1.14 |
| | 46_23 | 142.34 | 7202.7 | 143.23 | 30.0 | 143.44 | 21.4 | 0.63 | 0.77 |
| | 48_24 | 143.03 | 4131.1 | 143.20 | 34.1 | 143.20 | 26.2 | 0.12 | 0.12 |
| | 50_25 | 143.59 | 7204.0 | 144.57 | 47.8 | 143.82 | 38.7 | 0.68 | 0.16 |
| | **Average** | **118.22** | **2787.4** | **119.34** | **13.6** | **118.97** | **11.2** | **0.89** | **0.62** |

*Gap1 = [(4)–(2)]/(2) × 100%; Gap2 = [(6)–(2)]/(2) × 100%.

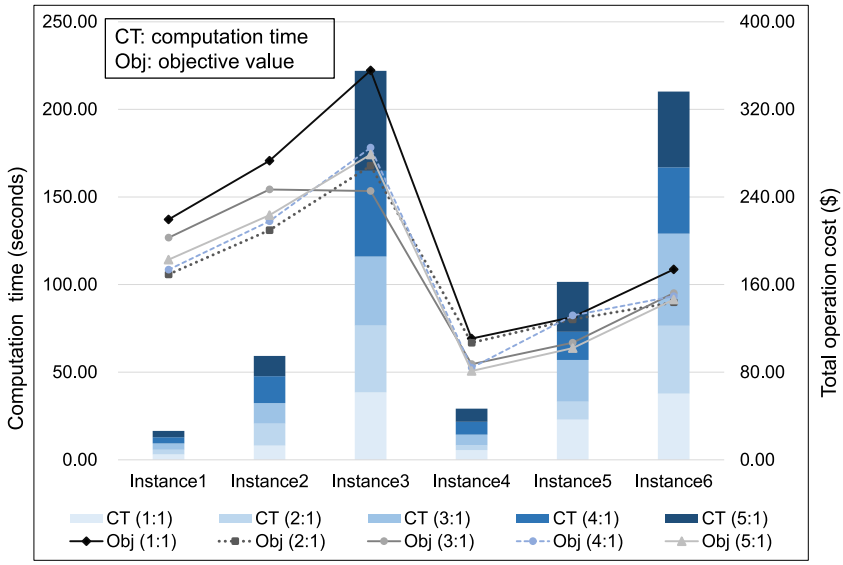



**Fig. 8.** Results for sensitivity analysis on the ratio of single-passenger to two-passenger requests.

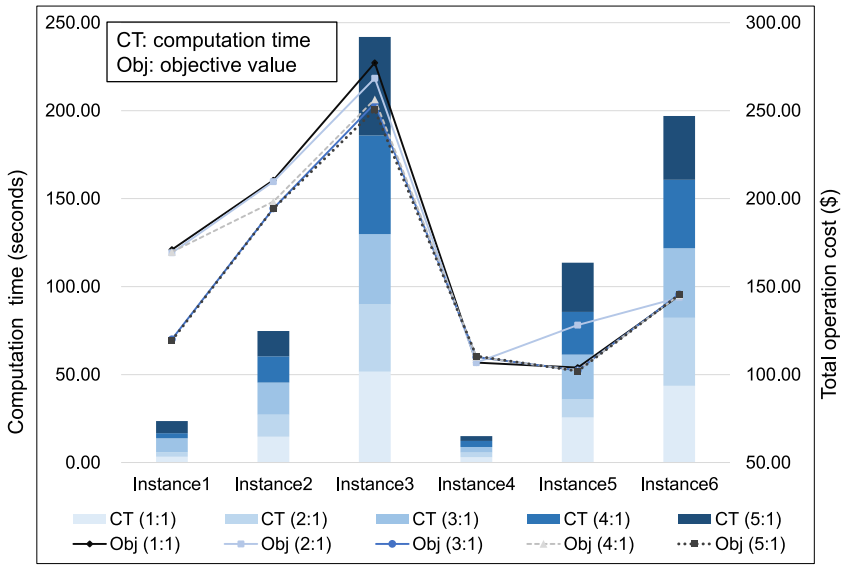



**Fig. 9.** Results for sensitivity analysis on ratios of general to high-priority requests.

### 5.5. Sensitivity analysis on the ratio of general to high-priority requests

To analyze the impact of different proportions of general to high-priority requests on operational costs and algorithmic efficiency, we conduct numerical experiments based on the instances mentioned in the previous section. The proportions of general to high-priority requests in these instances are set at 1:1, 2:1, 3:1, 4:1, and 5:1, respectively. The experimental results are shown in Fig. 9.

As shown in the figure, the average computation time for all instances with different proportions is 22.2 s, with the longest being 56.2 s. Additionally, the different proportions have minor impacts on computation time for the same instance, with a maximum deviation of 18 s. In terms of operational costs, the average operational cost decreases from $169.08 to $153.72 as the proportion of general requests increases. This is because general requests typically have lower service quality requirements, allowing each vehicle to serve more passengers, thus reducing the average operational cost.

### 5.6. Sensitivity analysis on vehicle fixed costs

To explore the impact of varying vehicle fixed costs on operational costs and algorithmic efficiency, we set the vehicle fixed costs at $1, $3, $5, $8, and $10, respectively. Then, we conduct numerical experiments based on the six instances mentioned in Section 5.4. The experimental results are shown in Fig. 10.

As shown in the figure, the average computation time for all instances with different vehicle fixed costs is 21.1 s, with the longest being 63.3 s. Furthermore, the computation time increases as the vehicle fixed cost increases for the same instance, with the average computation time increasing from 17.6 s to 27.0 s. In terms of operational costs, different vehicle fixed costs cause significant changes in total operational costs. The average operational cost increases from $150.56 to $244.68 as the vehicle fixed cost increases.

### 5.7. Sensitivity analysis on vehicle transportation costs

To examine the impact of different vehicle transportation costs on total operating costs and algorithmic efficiency, we perform numerical experiments based on the six instances mentioned in Section 5.4 and set vehicle transportation costs at 0.2 $/km, 0.3 $/km, 0.4 $/km, 0.5 $/km, 0.6 $/km, 0.7 $/km, and 0.8 $/km, respectively.

As shown in Fig. 11, the average computation time for all instances is 20.0 s, with the longest being 54.7 s. Additionally, different vehicle transportation costs have minor impacts on the computation time for the same instance. In terms of operational costs, different vehicle transportation costs cause significant changes in total operational costs. The average operational cost increases from $125.14 to $249.66 as the vehicle transportation cost increases.

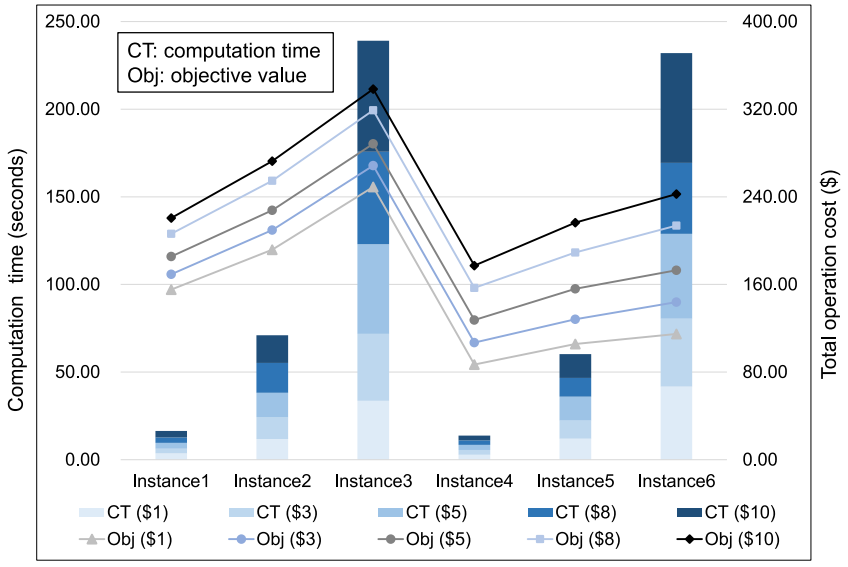



**Fig. 10.** Results for sensitivity analysis on vehicle fixed costs.

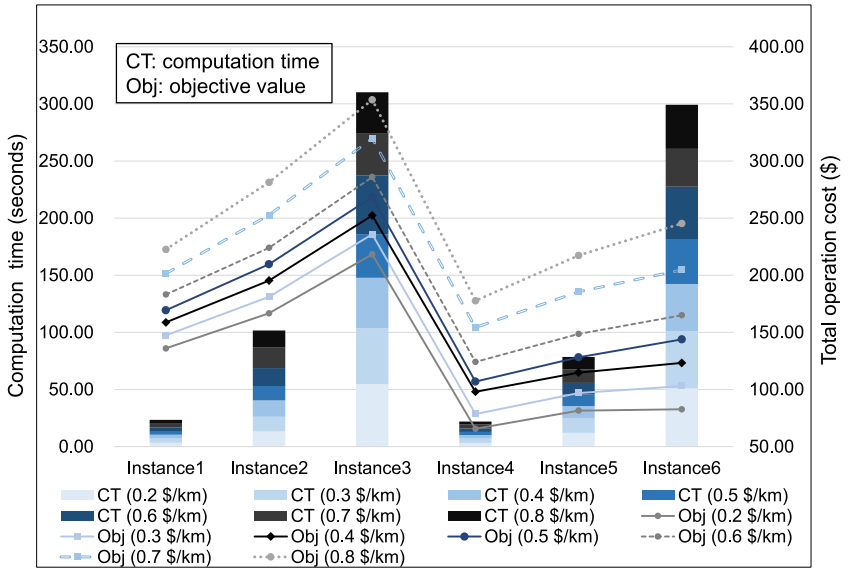



**Fig. 11.** Results for sensitivity analysis on vehicle transportation costs.

### *5.8. Service quality changes caused by considering service heterogeneity*

To explore the impact of service heterogeneity on service quality, we analyze the satisfaction of service quality with and without considering service heterogeneity based on instances with insufficient fleets. Note that service quality can be well satisfied when the fleet is sufficient. Requests with unfulfilled co-riding or detouring requirements are considered unmet requests. Results are reported in Fig. 12, from which two conclusions can be drawn. First, ignoring service heterogeneity leads to a lower satisfaction ratio among high-priority requests as compared to general requests due to the former's lower willingness to share rides and make detours. Second, incorporating service heterogeneity significantly improves service quality for high-priority groups, with the average unsatisfied rate dropping from 36% to 13%. However, taking service heterogeneity into account reduces the service quality for general requests, with their average unmet rate increasing from 25% to 35%. The results suggest that service heterogeneity is vital to improving service quality for high-priority groups when the fleet is insufficient, but it can come at the cost of service quality for general requests.

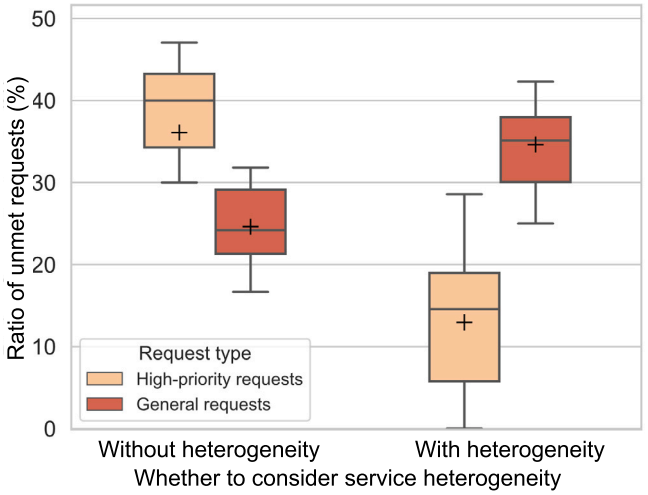



**Fig. 12.** Ratio of unmet request for quality of services.

## 6. Conclusion and future research

This paper investigates the first-mile ridesharing problem with service heterogeneity for different passenger groups. We minimize transit agencies' total operating costs and penalties for causing inconvenience to riders while incorporating service priority by giving higher penalty factors to high-priority groups. The decisions consist of matching requests, routing the heterogeneous fleet, and determining the optimal schedule for each route. We first develop an arc-based mixed integer linear programming model to solve this problem. For obtaining near-optimal solutions within practical computation time requirements, we reformulate the MILP model as a trip-based set-partitioning model and propose a math-heuristic algorithm. To identify trips more efficiently for the pricing subproblem, we introduce a new bidirectional labeling algorithm with novel dominance rules. To obtain exact solutions, we implement a branch-and-price algorithm.

Real-world case studies are conducted based on the road network and randomly generated requests and available vehicles around the Xinzhuang metro station in Shanghai, China. The results suggest that (1) the column-generation math-heuristic procedure is applicable for solving real-world-sized problems and obtaining near-optimal solutions efficiently. For instance, the math-heuristic approach can identify a solution for each case within 40 s. (2) When the fleet is insufficient, the service quality can be significantly improved for high-priority requests by giving them service priority, with the ratio of satisfied requests increasing by 23%, but it may sacrifice the service quality for general requests.

Although the proposed math-heuristic algorithm provides solutions in one minute for all instances, this work does not account for real-time updating of the ridesharing scheme to accommodate new incoming requests and vehicles. Learning-based methods, such as reinforcement learning algorithms, may be employed to update ridesharing schemes in real-time to respond to dynamic incoming requests and vehicles. Moreover, in future work, stochastic or robust approaches can be adopted to model and tackle the uncertainty of travel time.

## CRediT authorship contribution statement

**Ping He:** Writing – original draft, Validation, Software, Methodology, Conceptualization. **Jian Gang Jin:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization. **Martin Trépanier:** Writing – review & editing, Conceptualization. **Frederik Schulte:** Writing – review & editing, Conceptualization.

## 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.

## Acknowledgments

This work is supported by the National Natural Science Foundation of China [Grant 72122014, 72111540273]. The first author very appreciates the support of the China Scholarship Council [Grant 202106230253].

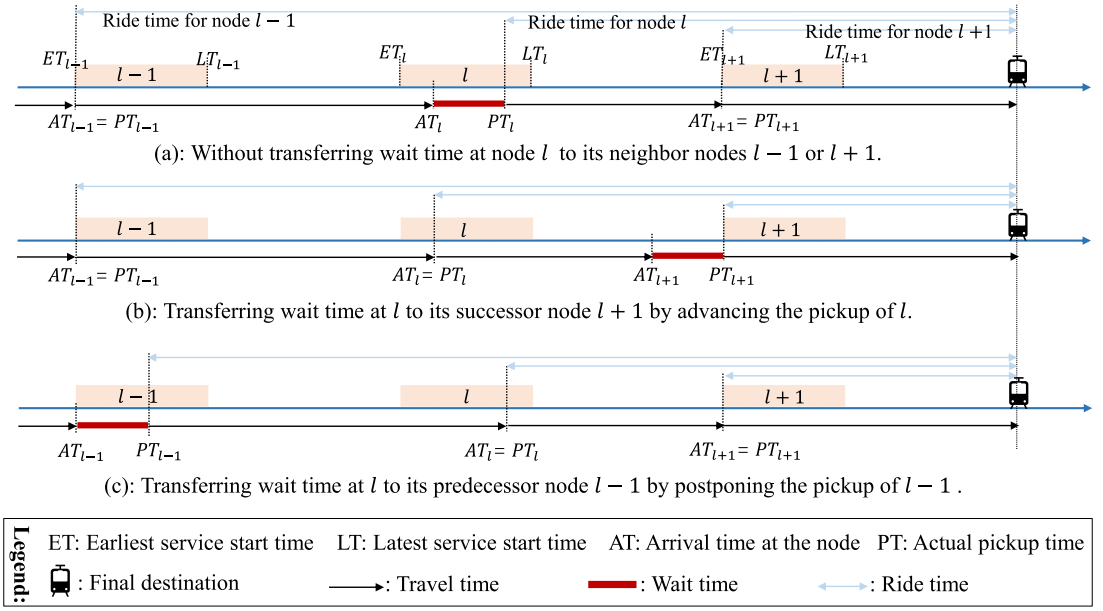



**Fig. A.13.** An illustrative example of shifting the waiting time of node $l$ to its neighborhood nodes, $l-1$ and $l+1$.

## Appendix. Proofs of Proposition 1

Let $t_{p,p+1}$ denote the travel time from node $p$ to its successor node, $p+1$, along route $\mathbb{S}_k$, and $AT_l$, $WT_l$, $PT_l$, as well as $\xi_l$ denote the arrival time, waiting time, pickup time, and ride time for node $l(\forall l \in \mathbb{S}_k)$, respectively. We assume that the service duration is zero for each node, as it can be incorporated into the travel time to simplify.

Then, we can use $\sum_{p\geq l} t_{p,p+1} + \sum_{p>l} WT_p$ to denote $\xi_l$, which consists of total travel time from node $l$ to the last node of $\mathbb{S}_k$ and total waiting times at these successor nodes of $l$. Since travel times between nodes are deterministic, the ride time of each node is only affected by waiting times at its successor nodes. Therefore, minimizing waiting times at its successor nodes allows us to obtain the minimum ride time for node $l$.

For a given node $l$, we can transfer some of its waiting time to neighborhood nodes, $l-1$ or $l+1$, by adjusting its pickup time, as shown in Fig. A.13. Based on $\sum_{p\geq l} t_{p,p+1} + \sum_{p>l} WT_p$, we find that transferring the waiting time from node $l$ to node $l+1$ may increase the ride time for node $l$ without reducing the ride time for other nodes, as shown in Fig. A.13(a) and (b). Accordingly, this type of waiting time transfer process is not helpful in reducing ride times. Instead, we transfer the waiting time at node $l$ to its predecessor node $l-1$ by postponing the pickup of $l-1$, with a maximum transfer limit of $\Delta = min\{LT_{l-1} - PT_{l-1}, PT_l - AT_l\}$. Then, the new ride time $(\xi'_l, \forall l \in \mathbb{S}_k)$ for each node is as follows:

•For node $l$, $\xi'_l = \sum_{p\geq l} t_{p,p+1} + \sum_{p>l} WT_p = \xi_l$;

•For node $l-1$, $\xi'_{l-1} = \sum_{p\geq l-1} t_{p,p+1} + (WT_l - \Delta) + \sum_{p>l} WT_p = \sum_{p\geq l-1} t_{p,p+1} + \sum_{p>l-1} WT_p - \Delta = \xi_{l-1} - \Delta < \xi_{l-1}$;

•For any node $z$ preceding node $l-1$, $\xi'_z = \sum_{p\geq z} t_{p,p+1} + \sum_{l-1>p>z} WT_p + (WT_{l-1}+\Delta) + (WT_l - \Delta) + \sum_{p>l} WT_p = \sum_{p\geq z} t_{p,p+1} + \sum_{p>z} WT_p = \xi_z$; and

•For any nodes following node $l$, their ride time remains the same since the travel times between each arc and wait time at each node are unchanged.

Thus, by transferring the waiting time at node $l$ to its predecessor node $l-1$ (postponing the pickup time of $l-1$), without changing the waiting time at other nodes, we can reduce the ride time for node $l-1$ and without change the ride time of other nodes, as shown in Fig. A.13(a) and (c). By starting the transferring process from the last node and shifting the waiting time of each node to its predecessor node, we can then obtain an optimal schedule with the minimal ride time for each node.

## References

Beirigo, B.A., Negenborn, R.R., Alonso-Mora, J., Schulte, F., 2022a. A business class for autonomous mobility-on-demand: Modeling service quality contracts in dynamic ridesharing systems. Transp. Res. C 136, 103520.

Beirigo, B.A., Schulte, F., Negenborn, R.R., 2022b. A learning-based optimization approach for autonomous ridesharing platforms with service-level contracts and on-demand hiring of idle vehicles. Transp. Sci. 56 (3), 677–703.

Bian, Z., Bai, Y., Liu, X., Wang, B., 2022. An online hybrid mechanism for dynamic first-mile ridesharing service. Transp. Res. C 138, 103585.

Bian, Z., Liu, X., 2019. Mechanism design for first-mile ridesharing based on personalized requirements part I: Theoretical analysis in generalized scenarios. Transp. Res. B 120, 147–171.

Bian, Z., Liu, X., Bai, Y., 2020. Mechanism design for on-demand first-mile ridesharing. Transp. Res. B 138, 77–117.
Bongiovanni, C., Geroliminis, N., Kaspi, M., 2024. A ride time-oriented scheduling algorithm for dial-a-ride problems. Comput. Oper. Res. 165, 106588.
Brown, A., Manville, M., Weber, A., 2021. Can mobility on demand bridge the first-last mile transit gap? Equity implications of Los Angeles' pilot program. Transp. Res. Interdiscip. Perspect. 10, 100396.
Bulhoes, T., Ha, M.H., Martinelli, R., Vidal, T., 2018. The vehicle routing problem with service level constraints. European J. Oper. Res. 265 (2), 544–558.
Chen, S., Wang, H., Meng, Q., 2020. Solving the first-mile ridesharing problem using autonomous vehicles. Comput.-Aided Civ. Infrastruct. Eng. 35 (1), 45–60.
Cordeau, J.-F., Laporte, G., 2003. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. Transp. Res. B 37 (6), 579–594.
Costa, L., Contardo, C., Desaulniers, G., 2019. Exact branch-price-and-cut algorithms for vehicle routing. Transp. Sci. 53 (4), 946–985.
Currie, G., Fournier, N., 2020. Why most DRT/Micro-transits fail – What the survivors tell us about progress. Res. Transp. Econom. 83, 100895.
Dabia, S., Ropke, S., Van Woensel, T., De Kok, T., 2013. Branch and price for the time-dependent vehicle routing problem with time windows. Transp. Sci. 47 (3), 380–396.
Desaulniers, G., Desrosiers, J., Solomon, M.M., 2006. Column Generation, vol. 5, Springer Science & Business Media.
Doan, T.T., Bostel, N., Hà, M.H., 2021. The vehicle routing problem with relaxed priority rules. EURO J. Transp. Logist. 10, 100039.
Dror, M., 1994. Note on the complexity of the shortest path models for column generation in VRPTW. Oper. Res. 42 (5), 977–978.
Grahn, R., Qian, S., Hendrickson, C., 2022. Optimizing first-and last-mile public transit services leveraging transportation network companies (TNC). Transportation 1–28.
He, P., Jin, J.G., Schulte, F., 2024. The flexible airport bus and last-mile ride-sharing problem: Math-heuristic and metaheuristic approaches. Transp. Res. E 184, 103489.
He, P., Jin, J.G., Schulte, F., Trépanier, M., 2023. Optimizing first-mile ridesharing services to intercity transit hubs. Transp. Res. C 150, 104082.
Huang, Y., Kockelman, K.M., Garikapati, V., 2022. Shared automated vehicle fleet operations for first-mile last-mile transit connections with dynamic pooling. Comput. Environ. Urban Syst. 92, 101730.
Joncour, C., Michel, S., Sadykov, R., Sverdlov, D., Vanderbeck, F., 2010. Column generation based primal heuristics. Electron. Notes Discrete Math. 36, 695–702, ISCO 2010 - International Symposium on Combinatorial Optimization.
Kumar, P., Khani, A., 2021. An algorithm for integrating peer-to-peer ridesharing and schedule-based transit system for first mile/last mile access. Transp. Res. C 122, 102891.
Lin, N., Akkerman, R., Kanellopoulos, A., Hu, X., Wang, X., Ruan, J., 2023. Vehicle routing with heterogeneous service types: Optimizing post-harvest preprocessing operations for fruits and vegetables in short food supply chains. Transp. Res. E 172, 103084.
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. Transp. Res. E 157, 102546.
Ma, T.-Y., Rasulkhani, S., Chow, J.Y., Klein, S., 2019. A dynamic ridesharing dispatch and idle vehicle repositioning strategy with integrated transit transfers. Transp. Res. E 128, 417–442.
Masoud, N., Nam, D., Yu, J., Jayakrishnan, R., 2017. Promoting peer-to-peer ridesharing services as transit system feeders. Transp. Res. Rec. 2650 (1), 74–83.
Righini, G., Salani, M., 2006. Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints. Discrete Optim. 3 (3), 255–273.
Shen, Y., Zhang, H., Zhao, J., 2018. Integrating shared autonomous vehicle in public transportation system: A supply-side simulation of the first-mile service in Singapore. Transp. Res. A 113, 125–136.
Si, H., Shi, J., Hua, W., Cheng, L., De Vos, J., Li, W., 2023. What influences people to choose ridesharing? An overview of the literature. Transp. Rev. 1–26.
Smith, S.L., Pavone, M., Bullo, F., Frazzoli, E., 2010. Dynamic vehicle routing with priority classes of stochastic demands. SIAM J. Control Optim. 48 (5), 3224–3245.
Sohrabi, S., Ziarati, K., Keshtkaran, M., 2024. Revised eight-step feasibility checking procedure with linear time complexity for the dial-a-ride problem (DARP). Comput. Oper. Res. 164, 106530.
Stiglic, M., Agatz, N., Savelsbergh, M., Gradisar, M., 2018. Enhancing urban mobility: Integrating ride-sharing and public transit. Comput. Oper. Res. 90, 12–21.
Wang, S., Baldacci, R., Yu, Y., Zhang, Y., Tang, J., Luo, X., Sun, W., 2023. An exact method for a first-mile ridesharing problem. Transp. Sci. 57 (6).
Wong, K.-I., Wong, S.C., Yang, H., Wu, J., 2008. Modeling urban taxi services with multiple user classes and vehicle modes. Transp. Res. B 42 (10), 985–1007.
Ye, J., Pantuso, G., Pisinger, D., 2022. Online order dispatching and vacant vehicles rebalancing for the first-mile ride-sharing problem. Available at SSRN 4265368.
Zhen, L., He, X., Zhuge, D., Wang, S., 2024. Primal decomposition for berth planning under uncertainty. Transp. Res. B 183, 102929.
Zhen, L., Xu, Z., Wang, K., Ding, Y., 2016. Multi-period yard template planning in container terminals. Transp. Res. B 93, 700–719.
Zhen, L., Zhuge, D., Wang, S., Wang, K., 2022. Integrated berth and yard space allocation under uncertainty. Transp. Res. B 162, 1–27.
Zheng, M., Pantuso, G., 2023. Trading off costs and service rates in a first-mile ride-sharing service. Transp. Res. C 150, 104099.
Zuo, T., Wei, H., Chen, N., Zhang, C., 2020. First-and-last mile solution via bicycling to improving transit accessibility and advancing transportation equity. Cities 99, 102614.