Hindawi
Journal of Advanced Transportation
Volume 2018, Article ID 6430950, 21 pages
https://doi.org/10.1155/2018/6430950

Research Article

Ridesharing Problem with Flexible Pickup and Delivery Locations for App-Based Transportation Service: Mathematical Modeling and Decomposition Methods

Meng Zhao,[1] **Jiateng Yin**,[2] **Shi An**,[1] **Jian Wang**,[1] **and Dejian Feng**[1]

[1]*School of Transportation Science and Engineering, Harbin Institute of Technology, Harbin 150090, China*
[2]*State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing 100044, China*

Correspondence should be addressed to Jian Wang; wjhit1974@163.com

Received 14 February 2018; Revised 14 May 2018; Accepted 24 May 2018; Published 5 July 2018

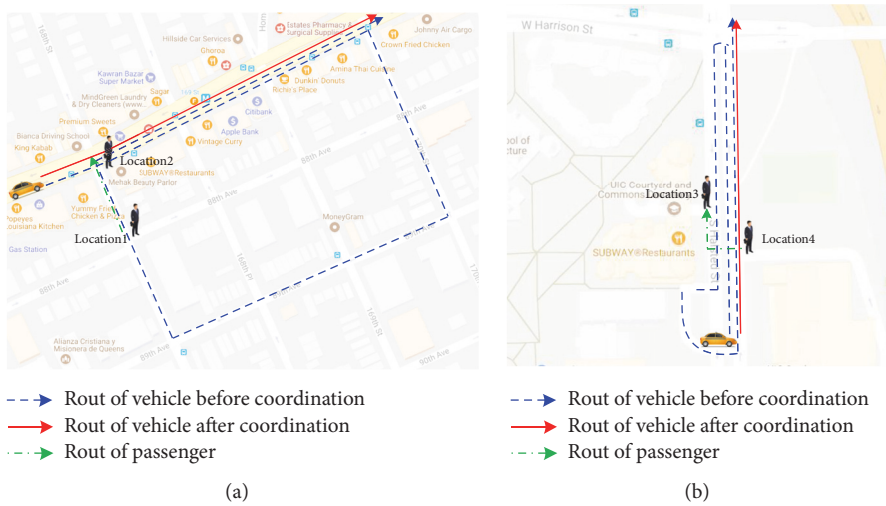
Academic Editor: Wai Yuen Szeto

App-based transportation service system, such as Uber and Didi, has brought a new transportation mode to users, who are able to make reservations using mobile apps conveniently. However, one of the fundamental challenges in app-based transportation system is the inefficiency and unreliability of the vehicle routing plans caused by complex topology of urban road network and unpredictable traffic conditions. A common way to tackle this problem is repositioning pickup or delivery locations via the coordination between drivers and passengers. This paper studies an on-demand ridesharing problem that determines the optimal ride-share matching strategy and vehicle routing plan with respect to flexible pickup and delivery locations. By introducing the concept of space-time windows, the problem is formulated as the pickup and delivery problem with space-time windows (PDPSW) in space-time network. To solve the model efficiently and accurately, we particularly develop a customized solution approach based on Lagrangian relaxation. Numerical examples are conducted to demonstrate the performance of the proposed framework and draw some managerial insights into the optimal system operation. The results indicate that adopting the serving strategy of flexible pickup and delivery locations will evidently reduce the system cost and improve the service quality in app-based transportation service systems.

1. Introduction

With the constant growth of urban size and population, private car use has increased rapidly for its convenience, flexibility, and comfort, which yet causes a series of traffic and environment issues, such as congestion, parking shortage, energy overconsumption, and air pollution. The app-based transportation network and taxi companies (TNC), such as Uber, Lyft, and Didi, have rapidly developed to provide the on-demand ridesharing service that achieves the balance of transport mobility and social benefit. For example, in 2017 the number of Uber user reservations has reached 40 million per month and Uber's share of the United States ride hailing market is 77% (http://www.businessofapps.com/data/uber-statistics/). In China, the number of TNC users is more than 150 million, accounting for 22.3% of the netizens.

Different from the conventional taxi companies, TNC is able to collect passengers' reservations through smartphone apps and quickly extract the detailed travel information, such as the origin and destination locations and the corresponding departure and arrival time windows. The request information is then converted into some task lists involving the specific service schedules (i.e., visiting times and locations) and routing path, which is then performed by a fleet of vehicles. This new technology brings great convenience to passengers but may incur a series of fleet management issues due to the complex topology of urban road network and unpredictable traffic conditions. Specifically, in some cases the pickup or delivery locations of passengers may be spatially nearby but topologically inaccessible or even temporally unreachable for vehicles. This will cause extra detours of vehicles or long-time waiting, which will apparently impact the efficiency of the operation system and reduce the service quality.



(a) (b)

Figure 1: Illustration of pickup and delivery locations repositioning.

In practice, one intuitive way to tackle this problem is repositioning pickup or delivery locations. In practice, the spatial accessibility and flexibility of passengers are much higher than that of vehicles within some local intersections or road segments. Such as shown in Figure 1(a), location 1 (the crossing of 168th St. and 88th Ave.) is the origin of passenger as requested. Since the 168th St. is one-way to north, the vehicle has to take a long detour to pick up the passenger, which is shown as blue dashed line. Alternatively, the driver will first coordinate the detailed pickup location with the passenger after receiving the task schedule and sometimes may reposition it to a new location that is easier to access. For example, the passenger could be asked to walk a short distance (shown as the green chain line) to the major road and then picked up at location 2. Furthermore, to deliver the passenger to the opposite side of S Halsted St., which is shown as location 3 in Figure 1(b), the passenger could be first dropped at location 4 and then walk to location 3 by crossing the crosswalk or overpass. Otherwise, the vehicle has to turn around at the crossing of S Halsted St. and W Harrison St. and then take another U turn back to north. Since mainly based on the drivers' personal experience and not involved in the whole system plan, such rescheduling operations are usually unreliable and may seriously disturb the task schedules. This is also one of the main limitations of the optimized vehicle routing planning in practical applications.

From the system-wide operation perspective, slightly repositioning the pickup and delivery locations of passengers will somehow reduce the vehicle traveling cost and avoid the risks of unexpected transportation delays. Besides, properly adjusting the relative positions of these locations may also increase the matching rate between vehicles and passengers. In the best cases, passengers could be gathered to the same pickup or delivery location and simultaneously served by one vehicle. Therefore, a reliable and flexible vehicle routing strategy for on-demand ridesharing service system is expected to balance the service quality and system cost considering the network complexity and operational dynamics.

Nevertheless, existing studies about the on-demand ridesharing problem with flexible pickup and delivery locations are very few. Different from the ridesharing problem with respect to developing the carpooling matching strategy, the on-demand ridesharing problem is also known in the literature as pickup and delivery problem with time windows (PDPTW), which is a much more complicated problem due to the complex coupling constraints among the vehicle routing, passenger assigning, and vehicle capacity limitations. On the other hand, the concept of flexible serving strategies is mostly introduced to classify and cluster the passengers in matching procedure of carpooling problems. To best of our knowledge, only a few of recent studies have made the attempt to integrate the ride-share matching strategy and vehicle dispatching plan with flexible pickup and delivery locations for the on-demand ridesharing systems.

To bridge the research gap, this paper aims to develop a mathematical model for the on-demand ridesharing operations with flexible pickup and delivery locations. By employing the space-time presentation and the concept of space-time window, the problem is further formulated as the pickup and delivery problem with space-time windows (PDPSW). By this, an integer linear programming (ILP) model is further proposed to simultaneously determine the optimal number of dispatched vehicles, routing plan, and detailed serving strategy, so as to minimize the fixed operation cost, vehicle traveling cost, and passengers' inconvenience cost with respect to the space-time flow balance constraints, passenger serving constraints, and vehicle capacity constraints.

Considering the complexity of the model, we present a customized solution approach based on Lagrangian relaxation (LR) algorithm. Specifically, we dualize two set of coupling constraints to separate two parts of the problem, i.e., the ride-share matching strategy and vehicle routing operation, by introducing different Lagrangian multipliers. The relaxed model is then decomposed into two sets of subproblems that can be seen as knapsack problem and shortest path problem, respectively. To efficiently obtain a feasible solution adapting

from the relaxed solution, a hybrid method is specially developed based on greedy algorithm and dynamic programming (DP). Eventually, a subgradient search is adopted to iteratively update the feasible and relaxed solutions to an acceptable tolerance of optimality gap. The performance of the LR based solution algorithm is then demonstrated through a multiscale experience comparing with that of CPLEX, a widely used commercial solver. Besides, we can also see from the sensitivity result that adopting the operation strategy of flexible pickup and delivery locations significantly reduces the system cost and yet guarantees the service quality.

2. Literature Review

Ridesharing has been widely studied ever since the 1970s, when carpooling attracted more and more attention for its energy efficiency and environmental protection. Particularly in recent years, the considerable improvement of communication capabilities and e-payment allow for a more accessible and secure online ridesharing service provide by TNCs. A systematic review can be seen in Furuhata et al. [1], where a classification is provided to better understand the existing ridesharing systems. Some key challenges are also introduced in this paper to clarify the development of the area of ridesharing problem and indicate the directions of future studies.

Mathematically, the ridesharing problem can be formulated as the PDPTW, or more specifically, the Dial-A-Ride Problem (DARP) when transporting passengers. As a generalized version of the vehicle routing problem (VRP), the PDPTW aims at determining the optimal routs of vehicles to satisfy the requests of pickup and delivery under the limitation of vehicle capacities, corresponding time windows, and coupling constraints (Savelsbergh and Sol [2], Berbeglia et al. [3], and Dumas et al. [4]). Since the VRP is a NP-hard problem, the PDPTW is also NP-hard and more difficult to solve. Therefore, there are number of studies focusing on developing efficient solution algorithms. Studies such as Ropke and Pisinger [5], Li and Lim [6], Baldacci et al. [7], Hosni et al. [8], Küçükoğlu and Öztürk [9], Hu and Chang [10], and Mahmoudi and Zhou [11] all consider the general PDPTW and provide customized heuristic or exact solution approach to efficiently solve certain scales of the problem. Hosni et al. [8] formulated the shared-taxi problem into a mixed interprogram, in which an optimized taxi dispatching strategy is designed to assign the passengers' reservations to a fleet of taxis with optimal routing plans. Furthermore, a LR based solution approach was presented to solve the problem efficiently with relatively small gaps comparing with CPLEX. Despite using the similar algorithmic framework, Mahmoudi and Zhou [11] decomposed the primal problem, which is simplified by adopting the state-space-time network representation, into a series of single PDPTWs. This allows a forward DP solution algorithm to solve the subproblems efficiently and accurately. Furthermore, a serious of numerical studies were conducted to demonstrate the performance of the proposed solution algorithm in solving the on-demand ridesharing problem in large-scales.

On the other hand, many studies have been conducted on ride-share matching problem that specially aims to improve the matching rate between vehicles and passengers or satisfy the passengers' reservations with the minimum fleet size (Brownstone and Golob [12], Ferrari et al. [13], Herbawi and Weber [14], Xu et al. [15], Pelzer et al. [16], Stiglic et al. [17], and Masoud and Jayakrishnan [18]). Herbawi and Weber [14] considered the ride matching problem with time windows, which optimizes the assignments of passengers to vehicles with respect to the serving order and detailed visiting times of passengers' pickup and delivery locations. A genetic and insertion based heuristic algorithm was further proposed and applied in real-time ride matching and iteratively improves the solution quality according to the realistic data. Stiglic et al. [17] introduced the concept of "meeting point" to increase flexibility of pickup and delivery points of passengers. Then a customized algorithm was designed and implemented to optimally match the vehicles and passengers in large-scale ridesharing system. The proposed modeling framework was further performed in a simulation study to show the benefits of adopting meeting point strategy in ridesharing systems. It is worth noting that Tong et al. [19] developed a joint optimization model to integrate the passenger-to-vehicle assignment plan and the bus routing schedule design for the customized bus service system. Based on the space-time network modeling framework, they formulated the proposed problem into a multicommodity network flow-based optimization model. To improve the performance of the proposed algorithm in solving large-scale instances of the problem, an LR based algorithm framework was employed to first reduce the solution space and then decomposed the problem into two sets of subproblems. Besides, two numerical experiences are performed to present the performance of the developed model and algorithm.

The rest of this paper is organized as follows. Section 3 presents the detail of the PDPSW that is further formulated into an integer programming model. Then the LR based solution algorithm is developed in Section 4. Section 5 presents a series of numerical experiments to demonstrate the performance of the proposed solution approach and discuss the experiment results in the system operation strategy perspective. Section 6 concludes this paper and offers some future research directions.

3. Problem Formulation

The framework of the proposed on-demand ridesharing operations is shown in Figure 2. The application of mobile Internet and apps allow the TNCs to collect the travel plans from user reservations, which involve the detailed location and corresponding picking up and delivery time windows. According to these pieces of information, the walk-in catchment area by each timestamp in the relevant time windows can be obtained with respect to the topological structure of urban network. Then, based on the practical traffic conditions and regulations, the potential pickup/delivery locations are further selected within these areas. Here, we define the space-time window as the set that contains all the potential locations at timestamps of the relevant time

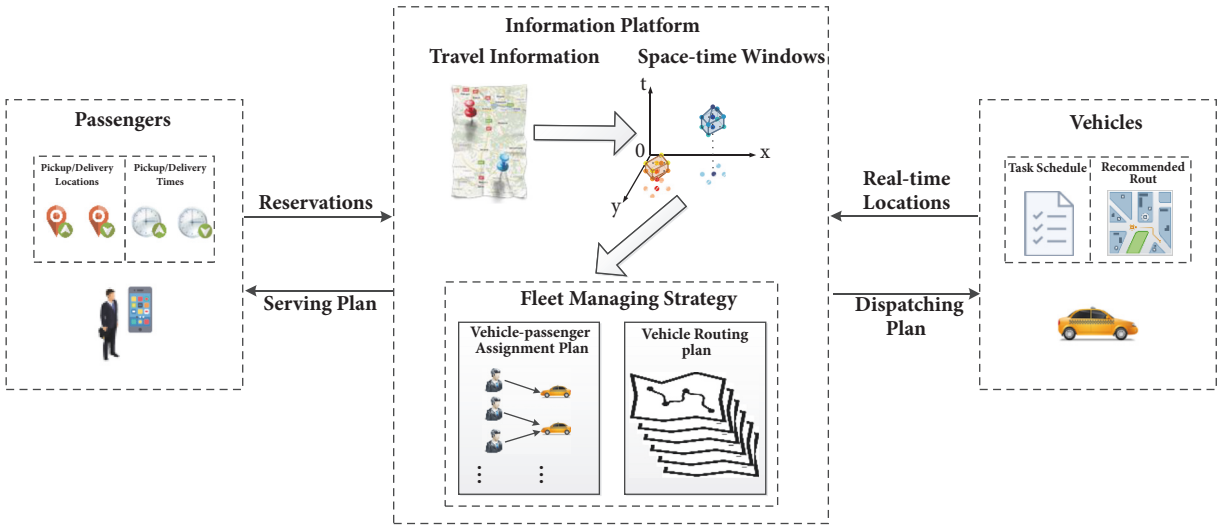


FIGURE 2: The framework of the on-demand ridesharing operations.

window. We consider that all the location-time pairs included in the space-time windows can be potentially selected to serve passengers. Therefore, the detailed vehicle-passenger assignment and vehicle routing plans, which involve the exact pickup and delivery locations and times of each passenger, will be determined by the modeling and decomposition method proposed in our study.

Furthermore, the detailed vehicle dispatching strategies will be sent to the corresponding vehicles in the form of task schedules and recommended routs, while sent to the passengers in the form of serving plans through the app, which involve the detailed pickup and delivery locations and times.

In contrast to the PDPTW, the extension on spatial dimension is considered in this study to further enhance the efficiency and reliability of the on-demand ridesharing system. To solve the on-demand ridesharing problem with flexible pickup and delivery locations, the specified serving plan is expected with respect to the dynamic spatiotemporal reservations from passengers, limited vehicle capacity, and minimum system cost. Therefore, we can see that the on-demand ridesharing system proposed in our study is essentially generalized to the flexible transit systems with virtual stops since the former one aims to provide a more flexible dispatching strategy to ensure the efficiency of the operation system with respect to the complex topology of urban road network and unpredictable traffic conditions. Nevertheless, in some cases, passengers whose space-time windows (partly) overlap each other could be gathered to the same pickup or delivery location and simultaneously served by one vehicle, which is a more general version comparing with the process of classifying and clustering the passengers in matching procedure of flexible transit systems.

Therefore, a completely new aspect is to simultaneously determine the optimal locations and times for picking up or delivering each passenger, which leads to obviously more complex formulations with respect to the passenger serving and vehicle dispatching plans. The adoption of space-time network enables the physical transportation network to integrate the vehicles' space-time trajectories and passengers' potential space-time distribution. In this regard, comparing with some classical modeling methods for solving Dial-A-Ride Problem (DRAP) (such as Baldacci et al. [7], Hosni et al. [8], and Hu and Chang [10]), our formulation provides a more explicit and compact modeling structure to represent the passengers serving and vehicle dispatching plans without adding the extra constraints, such as the subtour elimination constraints, temporal constraints, and coupling constraints between space and time related variables. This allows the development of a computationally efficient Lagrangian relaxation solution approach that can solve the PDPSW to near-optimum solutions.

In this section, we first introduce the PDPSW in Section 3.1. Then a systematical introduction of the PDPSW modeling framework with respect to the space-time network representation is present in Section 3.2. Eventually, Section 3.3 presents the model formulation of the PDPSW.

3.1. Description of PDPSW. We initially consider a directed transportation network denoted by $\mathbf{S} = (\mathbf{I}, \mathbf{L})$, where $\mathbf{I}$ (indexed by i and j) denotes the set of spatial nodes (such as the intersections in road network) and $\mathbf{L}$ (indexed by (i, j)) is the set of links representing the road segments. Let sequential set $\mathbf{T} = \{t_0, t_0 + \delta, \ldots, t_0 + (|\mathbf{T}| - 1)\delta\}$ denote the time horizon of the considered operation cycle, where t_0 is the initial timestamp and δ stands for the unit intervals between each two adjacent timestamps. Note that the time horizon in this study is assumed to be properly discretized so that all the activities, such as the trips of passengers or vehicles, can be considered starting at any timestamp and complete within integer multiples of δ. Figure 3 illustrates a $20 * 20$ grid network that denotes the transportation network and we assume that the time horizon is $[1, 16]$ and the unit interval $\delta = 1$. Each reservation of passenger is associated with the origin and destination, which are denoted by i_{p1}^{O} and i_{p1}^{D} of passenger $p \in \mathbf{P}$, respectively. We can see in Figure 3 that

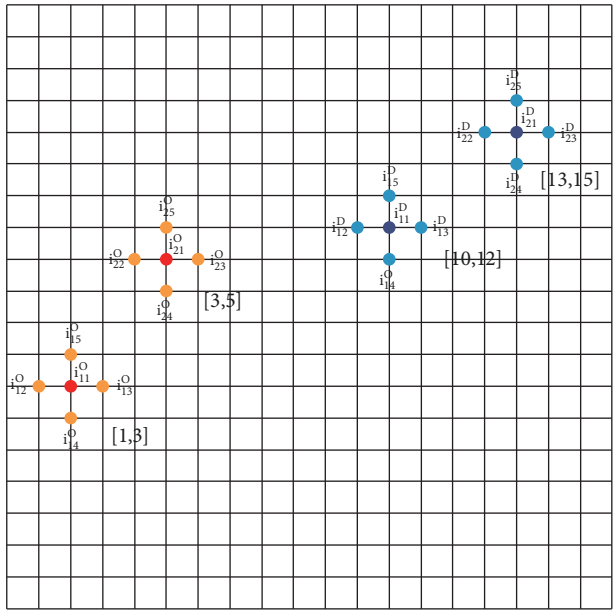


Figure 3: Illustration for PDPSW in spatial transportation network.

nodes i_{11}^O and i_{21}^O (marked as red dots), respectively, denote the origins of passengers 1 and 2, while i_{11}^D and i_{21}^D (marked as dark blue dots), respectively, represent the destination nodes of passenger 1 and 2. As shown in Figure 3, we choose nodes i_{p2}^O-i_{p5}^O that are topologically adjacent to i_{p1}^O as the expanded pickup nodes. Correspondingly, nodes i_{p2}^D-i_{p5}^D are selected as the expanded delivery nodes with respect to i_{p1}^D. Then we define the set that consists nodes i_{p1}^O-i_{p5}^O as the pickup space window of passenger p and denoted by $\mathbf{I}_p^O$, while i_{p1}^D-i_{p5}^D is defined as the delivery space window and denoted by $\mathbf{I}_p^D$. In PDPSW, passengers are supported to be picked up (or delivered) at any nodes within the corresponding pickup (or delivery) space windows. That is, within a predefined maximum walking distance, a passenger can be served at origin nodes or the nodes nearby. Moreover, each reservation of passengers also contains a pair of preferred pickup and delivery time windows with respect to the origin and destination nodes, which are denoted by sequential sets $\mathbf{T}_p^O \subseteq \mathbf{T}$ and $\mathbf{T}_p^D \subseteq \mathbf{T}$. As shown in Figure 3, the pickup time windows of passenger 1 and 2 are, respectively, set as [1, 3] and [3, 5], while the delivery time windows are [10, 12] and [13, 15].

After receiving the passengers' reservations, a certain number of vehicles (denoted by $\mathbf{H}$ and indexed by h) are sent out to pick up the passengers from one of their candidate pickup nodes $i_{pn}^O \in \mathbf{I}_p^O$ within the related pickup time window $\mathbf{T}_p^O$ and then deliver them to one of their candidate delivery nodes $i_{pn}^D \in \mathbf{I}_p^D$ within the related delivery time window $\mathbf{T}_p^D$. Note that, in the PDPSW, passengers are supported to share their ride with the others, which means that vehicles can simultaneously serve multiple passengers during the operation cycle under the limitation of vehicle capacity C_h ($h \in \mathbf{H}$).

3.2. Space-Time Network Representation. Space-time network is able to explicitly depict and rigorously formulate the spatiotemporal movement of commodities with compact model formulations, and it has been widely used in many transportation modeling studies (Kliewer et al. [20], Yang and Zhou [21], Tong et al. [22], Zhen and Jing [23], Li et al. [24], and Zhang et al. [25]). Since the PDPSW is essentially the generalization of PDPTW and aims to simultaneously determine the ride-share matching strategy and routing schedules of vehicles within the space-time dimension, we can formulate the PDPSW by using the space-time network representation.

In particular, a space-time network can be obtained by extending the space network on the time horizon. An illustration example can be seen in Figure 4, where grid network $\mathbf{S}$ in Figure 3 is partly shown as the x-y plane. Time horizon $\mathbf{T}$ is shown as t-axis vertical on the x-y plane. Let set $\mathbf{G} = (\mathbf{V}, \mathbf{A})$ denote the space-time network, where $\mathbf{V}$ (indexed by (i, j)) is the set of space-time vertexes and $\mathbf{A}$ (indexed by (i, t, j, s)) is the set of space-time arcs. Based on the space-time network, we can intuitively present the passengers' reservations by combining the space window and time window. Specifically, we denote the pickup and delivery space-time windows of passenger p by set $\mathbf{V}_p^O$ and $\mathbf{V}_p^D$, which are shown as red and blue polygons in Figure 4. Each space-time window contains all the candidate pickup or delivery space-time vertexes. Considering the duration of time that passengers spend to walk from the origin to the extended pickup nodes, there is only the origin vertex at the earliest timestamp in the pickup time window. As shown in Figure 4, the red dots represent the origin vertexes of passengers. To simplify the illustration, we here assume that it takes passengers one time interval from origin node to all the adjacent pickup nodes. Hence there are five potential pickup vertexes from the second to the last timestamps in the pickup time window (marked as orange and yellow dots). Similarly, there is only the destination vertex (marked as dark blue dotes) at the latest timestamp in delivery time window and five potential pickup vertexes from the earliest to the second last timestamps in the delivery time window.

To serve passengers 1 and 2, vehicle h_1 ($C_{h_1} \geq 2$) is sent out to travel according to a feasible routing plan, which is denoted by the space-time path marked as the green line in Figure 4. We can see that h_1 initially picks up passengers 1 and 2 at space-time vertexes $(i_{13}^O, 2)$ and $(i_{24}^O, 4)$ and then delivers passenger 1 when passing $(i_{13}^D, 10)$ and eventually delivers passenger 2 at $(i_{21}^D, 13)$.

3.3. Model Formulations. Based on the constructed space-time network, an ILP model is roughly formulated in this section. The following assumptions are initially proposed to focus the model on the essence of PDPSW.

Demand. All passengers' reservations are considered to be dynamic and deterministic. That is, all the reservations are given before the operation cycle and cannot be canceled or delayed. In this sense, the detailed serving schedule of vehicles can be obtained based on the serving requests,

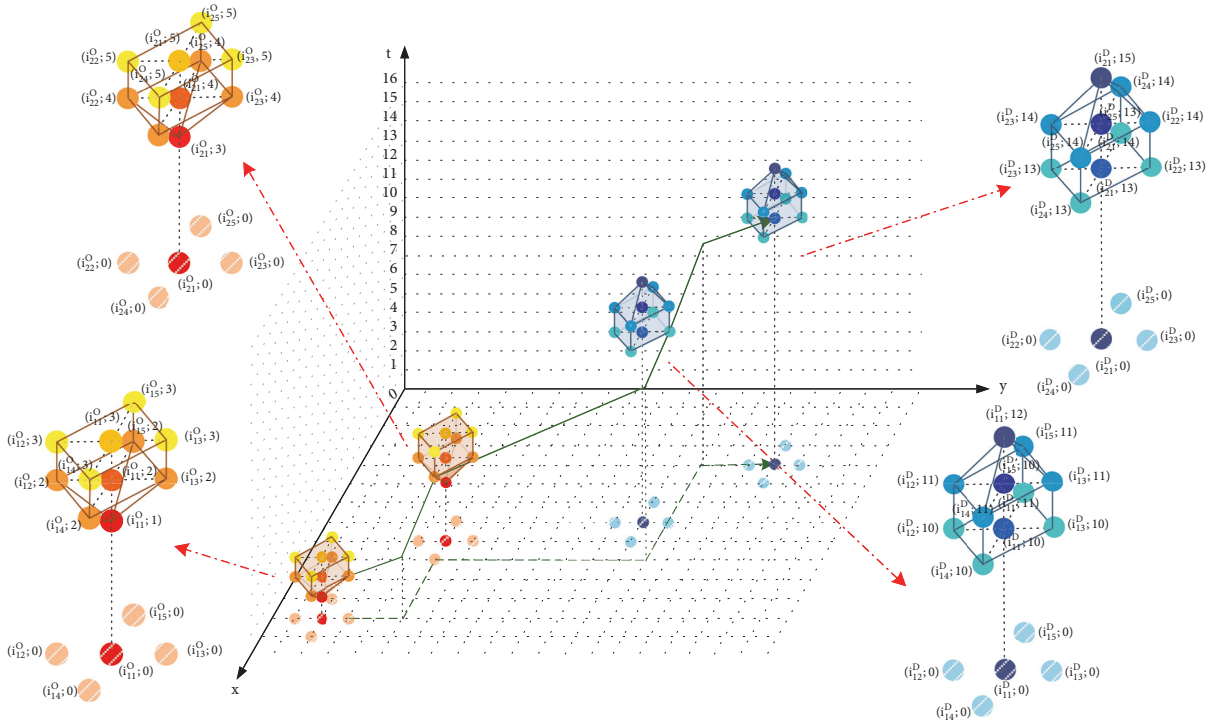


Figure 4: Representation for PDPSW in space-time network.

i.e., the pickup and delivery space-time windows in the formulation.

Space-Time Window. We assume that the pickup and delivery requests of all the passengers are executable. It means that, for each passenger, the time intervals between pickup and delivery space-time windows shall be no less than the vehicle's shortest traveling time along the fastest routing between the closet pickup and delivery nodes.

Vehicle. To simplify the problem without generalization, the vehicles are assumed to be identical; i.e., they have the same capacity and no passengers is onboard at the beginning and ending of the operation cycle.

Besides, for reader's convenience, the major notations of sets, parameters, and decision variables related to the model formulation are listed below.

Sets and Parameters

A: set of space-time arcs in the space-time network, indexed by (i, t, j, s)

H: set of vehicles, indexed by h

I: set of spatial transportation nodes, indexed by i and j

$\mathbf{I}_p^O$: pickup space window of passenger p

$\mathbf{I}_p^D$: delivery space window of passenger p

P: set of passengers, indexed by passenger p

$\mathbf{P}_p^O$: set of passengers whose departure time window contains timestamp t

$\mathbf{P}_p^D$: set of passengers whose arrival time window contains timestamp t

T: set of timestamps in the considered operation cycle, indexed by t and s

$\mathbf{T}_p^O$: pickup time window of passenger p

$\mathbf{T}_p^D$: delivery time window of passenger p

$\mathbf{V}_p^O$: pickup space-time window of passenger p

$\mathbf{V}_p^D$: delivery space-time window of passenger p

C_h: capacity of a vehicle h

c^f: fixed cost of dispatching a vehicle

c_{ij}^l: vehicle traveling cost through link (i, j)

c_{pi}^O: inconvenience costs for picking up passenger p at pickup node i

c_{pi}^D: inconvenience costs for delivering passenger p at delivery node i

Decision Variables

W_{it}^{hp}: whether vehicle h drops off passenger p at space-time vertex (i, t)

X_{itjs}^h: whether vehicle h travels through space-time arc (i, t, j, s)

Y_h: whether vehicle h is used

Z_{it}^{hp}: whether vehicle h picks up passenger p at space-time vertex (i, t)

Based on the above assumptions, the mathematical model for PDPSW is formulated as follows. We first propose a series of systemic constraints, such as the space-time flow balance constraints, passenger serving constraints, and vehicle capacity constraints. Then, the objective function, which is essentially a summation of all cost components, is further developed.

Space-Time Flow Balance Constraints. To precisely capture the space-time path of each vehicle, we initially introduce the space-time flow balance constraints:

$$\sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} X^h_{itjs} - \sum_{(j,s)\in\mathbf{V}:(j,s,i,t)\in\mathbf{A}} X^h_{jsit} \begin{cases} = Y_h, & (i,t) = (i_O, t_O), \\ = -Y_h, & (i,t) = (i_D, t_D), \\ = 0, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H}, \tag{1}$$

where binary variables $\mathbf{X} \coloneqq \{X^h_{itjs}\}_{h\in\mathbf{H},(i,t,j,s)\in\mathbf{A}}$ and $\mathbf{Y} \coloneqq \{Y_h\}_{h\in\mathbf{H}}$ determine the space-time path of vehicles and the selection of vehicles, respectively. Specifically, $X^h_{itjs} = 1$ if and only if space-time arc (i, t, j, s) is involved in the space-time path of vehicle h and $Y_h = 1$ if and only if vehicle h is dispatched to serve passengers. Note that space-time vertexes (i_O, t_O) and (i_D, t_D) are defined as the dummy origin and destination vertexes of all the vehicles, respectively. We assume that the distance and travel time from dummy origin to any physical space-time vertex are 0 and infinity the other way round, while those from physical space-time vertexes to dummy destination are 0 and also infinity the other way round.

Passenger Serving Constraints. For any TNC, fulfilling all the reservations of the passengers is actually critical for guaranteeing the serving quality. Therefore, we assume that the proposed PDPSW formulation is based on the undersaturated condition. That is, the total fleet size is assumed to be sufficient for serving all the passengers. We define binary variables $\mathbf{Z} \coloneqq \{Z^{hp}_{it}\}_{h\in\mathbf{H},p\in\mathbf{P},(i,t)\in\mathbf{V}^O_p}$ and $\mathbf{W} \coloneqq \{W^{hp}_{it}\}_{h\in\mathbf{H},p\in\mathbf{P},(i,t)\in\mathbf{V}^D_p}$ to denote the passenger serving plan, i.e., $Z^{hp}_{it} = 1$ if and only if vehicle h picks up passenger p at pickup space-time vertex (i, t) and $W^{hp}_{it} = 1$ if and only if vehicle h deliver passenger p at delivery space-time vertex (i, t). Thus, the following constraints are proposed to guarantee that all the passengers' reservations are fulfilled:

$$\sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}^O_p} Z^{hp}_{it} = 1, \quad \forall p \in \mathbf{P}, \tag{2}$$

$$\sum_{(i,t)\in\mathbf{V}^O_p} Z^{hp}_{it} = \sum_{(i,t)\in\mathbf{V}^D_p} W^{hp}_{it}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H} \tag{3}$$

where constraints (2) ensure that each passenger is picked up by a vehicle and constraints (3) impose that passenger p is delivered by vehicle h if p is previously picked up by it.

Moreover, constraints (4) and (5) are proposed to build the relationship between the vehicle routing plan and passenger serving strategy.

$$Z^{hp}_{it} - \sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} X^h_{itjs} \le 0, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}^O_p, \tag{4}$$

$$W^{hp}_{it} - \sum_{(j,s)\in\mathbf{V}:(j,s,i,t)\in\mathbf{A}} X^h_{jsit} \le 0, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}^O_p, \tag{5}$$

In particular, constraints (4) indicate that passenger p is picked up by h from space-time vertex (i, t) if and only if (i, t) is visited by h, while constraints (5) require that vehicle h delivers passenger p to space-time vertex (i, t) if and only if (i, t) is involved in the routing path of h.

Remark 1. Note that to ensure the service quality, which is regarded as the highest priority of most TNCs, all the reservations of passengers are required to be satisfied and thus the fleet size is assumed to be sufficient though only part of them is eventually sent out with respect to the optimized vehicle dispatching strategy. In practice, the fleet size will be set the same as the number of reservations (shown as Section 5). Then during the iteration of our LR algorithm (see Section 4 for details), part of the vehicles will be saved as the optimization result of vehicle-passenger assignment plan and vehicle routing plan. Even though we do acknowledge that, with limited vehicles (especially the total number of vehicles is less than the number of passengers' reservations), it may be difficult to find feasible solutions in PDPSW with respect to the passengers fulfilling constraints.

Vehicle Capacity Constraints. Different from carpooling or customized vanpooling service design, the on-demand ridesharing system allows vehicles to pick up and deliver passengers at any time and the seats are not fixed to any specific passenger during the operation cycle. Therefore, for each vehicle, the capacity constraints are needed at each timestamp; i.e.,

$$\sum_{t'=1}^{t} \sum_{p\in\mathbf{P}^O_{t'}} \sum_{i\in\mathbf{I}:(i,t')\in\mathbf{V}^O_p} Z^{hp}_{it'} - \sum_{t'=1}^{t} \sum_{p\in\mathbf{P}^D_{t'}} \sum_{i\in\mathbf{I}:(i,t')\in\mathbf{V}^D_p} W^{hp}_{it'} \le C_h, \quad \forall h \in \mathbf{H},\ t \in \mathbf{T} \tag{6}$$

where $\mathbf{P}^O_t$ denotes the set of passengers whose departure time window contains timestamp t, while $\mathbf{P}^D_t$ denotes the set of passengers whose arrival time window contains timestamp t. Then we can see that $\sum_{t'=1}^{t} \sum_{p\in\mathbf{P}^O_{t'}} \sum_{i\in\mathbf{I}:(i,t')\in\mathbf{V}^O_p} Z^{hp}_{it'}$ denotes the number of passengers that are picked up by vehicle h by the time of t', while $\sum_{t'=1}^{t} \sum_{p\in\mathbf{P}^D_{t'}} \sum_{i\in\mathbf{I}:(i,t')\in\mathbf{V}^D_p} W^{hp}_{it'}$ is the number of passengers that are delivered by vehicle h by the time of t'. The difference between them is the number of passengers on board by the time of t', which shall be no bigger than C_h.

Besides, the following constraints are further added to postulate binary variable values.

$$X_{itjs}^{h} \in \{0,1\}, \quad \forall h \in \mathbf{H},\ (i,t,j,s) \in \mathbf{A}, \tag{7}$$

$$Y_h \in \{0,1\}, \quad \forall h \in \mathbf{H}, \tag{8}$$

$$Z_{it}^{hp} \in \{0,1\}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}_p^O, \tag{9}$$

$$W_{it}^{hp} \in \{0,1\}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}_p^D, \tag{10}$$

Objective Function. The objective of PDPSW is to determine the optimal ride-share matching strategy and vehicle routing plan to balance the total system cost and serving quality. In our study, the objective function contains three parts. The first part is the fixed cost for dispatching vehicles. It can be considered as a summation of amortized cost of vehicles and hiring cost of drivers per operation cycle; i.e.,

$$CF = \sum_{h \in \mathbf{H}} c^f Y_h, \tag{11}$$

where c^f denotes the fixed cost of dispatching a vehicle. The second part refers to the vehicle routing cost; i.e.,

$$CT = \sum_{h \in \mathbf{H}} \sum_{(i,t,j,s) \in \mathbf{A}} c_{ij}^{l} X_{itjs}^{h}, \tag{12}$$

where c_{ij}^{l} represents the vehicle traveling cost through link (i, j).

The last part is the passenger inconvenience cost; i.e.,

$$CP = \sum_{p \in \mathbf{P}} \sum_{h \in \mathbf{H}} \left(\sum_{(i,t) \in \mathbf{V}_p^O} c_{pi}^{O} Z_{it}^{hp} + \sum_{(i,t) \in \mathbf{V}_p^D} c_{pi}^{D} W_{it}^{hp} \right), \tag{13}$$

where c_{pi}^{O} is the inconvenience cost for picking up passenger p at pickup node i, while c_{pi}^{D} denotes the inconvenience cost for delivering passenger p at delivery node i. Note that CP essentially indicates the cost of poor service quality. In PDPSW, though all the passengers are served within the corresponding space-time windows, it yet causes the decline of service level if passengers are not picked up or delivered passengers at the origin pickup or delivery space-time vertexes. An example can be seen in Figure 4, the service of picking up passengers 1 and 2, and delivering passenger 1 will all incur the passenger inconvenience cost, while delivering passenger 2 will not cause such cost since he/she is delivered right at the origin delivery space-time vertex $(i_{21}^{D}, 13)$.

Note that CF, CT, and CP are all determined by the vehicle-passenger assignment and vehicle routing plans, so these cost components shall generally exhibit the following tradeoffs. Increasing the vehicle dispatching cost shall bring down the number of vehicles dispatched to serve passengers but increase the vehicle routing cost and passenger inconvenience cost. The higher inconvenience cost, which though raises the service quality, will force the vehicles to pick up and deliver the passengers at the origin and destination as early as possible. This will somehow decrease the matching rate between passengers and vehicles. Thus, more vehicles will be required to fulfill the passengers with extra detour, which eventually raises the vehicle dispatching and routing costs. In order to quantitatively solve the PDPSTW, the following integer programming model is formulated:

$$\min_{\mathbf{X},\mathbf{Y},\mathbf{Z}} \quad \sum_{h \in \mathbf{H}} c^f Y_h + \sum_{h \in \mathbf{H}} \sum_{(i,t,j,s) \in \mathbf{A}} c_{ij}^{l} X_{itjs}^{h} + \sum_{p \in \mathbf{P}} \sum_{h \in \mathbf{H}} \left(\sum_{(i,t) \in \mathbf{V}_p^O} c_{pi}^{O} Z_{it}^{hp} + \sum_{(i,t) \in \mathbf{V}_p^D} c_{pi}^{D} W_{it}^{hp} \right) \tag{14}$$

$$\text{s.t.} \quad \sum_{(j,s) \in \mathbf{V}:(i,t,j,s) \in \mathbf{A}} X_{itjs}^{h} - \sum_{(j,s) \in \mathbf{V}:(j,s,i,t) \in \mathbf{A}} X_{jsit}^{h} \begin{cases} = Y_h, & (i,t) = (i_O, t_O), \\ = -Y_h, & (i,t) = (i_D, t_D), \\ = 0, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H}, \tag{15}$$

$$Z_{it}^{hp} - \sum_{(j,s) \in \mathbf{V}:(i,t,j,s) \in \mathbf{A}} X_{itjs}^{h} \le 0, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}_p^O, \tag{16}$$

$$W_{it}^{hp} - \sum_{(j,s) \in \mathbf{V}:(j,s,i,t) \in \mathbf{A}} X_{jsit}^{h} \le 0, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H},\ (i,t) \in \mathbf{V}_p^O, \tag{17}$$

$$\sum_{h \in \mathbf{H}} \sum_{(i,t) \in \mathbf{V}_p^O} Z_{it}^{hp} = 1, \quad \forall p \in \mathbf{P}, \tag{18}$$

$$\sum_{(i,t) \in \mathbf{V}_p^O} Z_{it}^{hp} = \sum_{(i,t) \in \mathbf{V}_p^D} W_{it}^{hp}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H} \tag{19}$$

$$\sum_{t'=1}^{t} \sum_{p \in \mathbf{P}_{t'}^O} \sum_{i \in \mathbf{I}:(i,t') \in \mathbf{V}_p^O} Z_{it'}^{hp} - \sum_{t'=1}^{t} \sum_{p \in \mathbf{P}_{t'}^D} \sum_{i \in \mathbf{I}:(i,t') \in \mathbf{V}_p^D} W_{it'}^{hp} \le C_h, \quad \forall h \in \mathbf{H},\ t \in \boldsymbol{T} \tag{20}$$

$$X_{itjs}^{h} \in \{0,1\}, \quad \forall h \in \mathbf{H},\ (i,t,j,s) \in \mathbf{A}, \tag{21}$$

$$Y_h \in \{0,1\}, \quad \forall h \in \mathbf{H}, \tag{22}$$

$$Z_{it}^{hp} \in \{0,1\}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H}, (i,t) \in \mathbf{V}_p^O, \tag{23}$$

$$W_{it}^{hp} \in \{0,1\}, \quad \forall p \in \mathbf{P},\ h \in \mathbf{H}, (i,t) \in \mathbf{V}_p^D, \tag{24}$$

4. Lagrangian Relaxation-Based Solution Approach

The PDPSW proposed in Section 3.3 is a NP-hard problem since PDPTW is essentially its special case, let alone the two sets of coupling constraints (16) and (17). Therefore, for relatively large-scale instances, it is difficult to solve the model

to optimum using commercial integer programming solvers (e.g., CPLEX) or exact solution approaches (e.g., branch and bound). To tackle this challenge, a customized solution approach based on LR is proposed to solve the problem efficiently with a near-optimum solution.

In the following content, Section 4.1 proposes a LR based decomposing approach to obtain the lower bound to the optimal value of primal problem (14)–(24). Basically, by relaxing hard constraints (16) and (17), we decompose the problem into two sets of subproblems that can be easily solved. However, since the relaxed solution is likely infeasible to the primal problem, yet acting as the lower bound, a DP based hybrid method is developed in Section 4.2 to adjust the relaxed solution into a feasible solution to the primal problem. Finally, a subgradient algorithm is employed in Section 4.3 to iteratively update the upper and lower bounds to obtain a near-optimum solution with an applicable optimality gap.

4.1. Model Decomposition. To relax constraints (16) and (17), we add the product of the left-hand side of them into objective function (14) with multipliers $\boldsymbol{\lambda} := \{\lambda_{it}^{hp} \geq 0\}_{h\in\mathbf{H},p\in\mathbf{P},(i,t)\in\mathbf{V}_p^O}$ and $\boldsymbol{\mu} := \{\mu_{it}^{hp} \geq 0\}_{h\in\mathbf{H},p\in\mathbf{P},(i,t)\in\mathbf{V}_p^D}$, respectively. The primal problem (14)–(24) is then relaxed as the following formulation:

$$\begin{aligned}\Delta(\boldsymbol{\lambda},\boldsymbol{\mu}) = \min_{\mathbf{X},\mathbf{Y},\mathbf{Z},\mathbf{W}} \quad & \sum_{h\in\mathbf{H}} c^f Y_h + \sum_{h\in\mathbf{H}} \sum_{(i,t,j,s)\in\mathbf{A}} c_{ij}^l X_{itjs}^h \\ & - \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^O} \lambda_{it}^{hp} \sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} X_{itjs}^h \\ & - \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^D} \mu_{it}^{hp} \sum_{(j,s)\in\mathbf{V}:(j,s,i,t)\in\mathbf{A}} X_{jsit}^h \\ & + \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^O} \left(c_{pi}^O + \lambda_{it}^{hp}\right) Z_{it}^{hp} \\ & + \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^D} \left(c_{pi}^D + \mu_{it}^{hp}\right) W_{it}^{hp},\end{aligned} \tag{25}$$

subject to constraints (15) and (18)–(24).

Note that since the coupling constrains (16) and (17) are relaxed, variables **X** and **Y** are separated from **Z** and **W**, which means that the vehicles' routing plans are released from the passengers serving strategies. Then the relaxed problem is decomposed into two sets of subproblems.

Subproblem 2 (shortest path problem). The first set contains $|\mathbf{H}|$ subproblems associated with variables **X** and **Y**:

$$\begin{aligned}\mathrm{T}_h(\boldsymbol{\lambda},\boldsymbol{\mu}) = \min_{\mathbf{X},\mathbf{Y}} \quad & c^f Y_h + \sum_{(i,t,j,s)\in\mathbf{A}} c_{ij}^l X_{itjs}^h \\ & - \sum_{p\in\mathbf{P}} \sum_{(i,t)\in\mathbf{V}_p^O} \lambda_{it}^{hp} \sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} X_{itjs}^h \\ & - \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^D} \mu_{it}^{hp} \sum_{(j,s)\in\mathbf{V}:(j,s,i,t)\in\mathbf{A}} X_{jsit}^h,\end{aligned} \tag{26}$$

subject to (15), (21) and (22).

We can see that objective function (26) involves two parts with respect to variables **X** and **Y**, respectively. The first part is the vehicle dispatching cost that consists of vehicles amortized cost and driver hiring cost. It is determined by variables **Y** and fixed for each vehicle $h \in \mathbf{H}$. The second part represents the generalized vehicle traveling cost related to the traveling path of vehicles in the space-time network and determined by variables **X**. Since Subproblem 2 only subjects to the space-time flow balance and binary constraints, the second part can be easily regarded as a time-dependent shortest path problem that can be solved to the optimum using some exact solution approach, such as forward DP, label setting, and label correcting (Mahmoudi and Zhou [7], Yin et al. [26], and Yang and Zhou [27]). In specific, we set the generalized vehicle traveling cost as

$$\eta_{itjs}^h = c_{ij}^l - \sum_{p\in\mathbf{P}} \alpha_{itjs}^{hp} - \sum_{p\in\mathbf{P}} \beta_{itjs}^{hp}, \quad \forall (i,t,j,s) \in \mathbf{A} \tag{27}$$

where

$$\alpha_{itjs}^{hp} = \begin{cases} \lambda_{it}^{hp}, & \text{if } (i,t)\in\mathbf{V}_p^O,\ (j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}, \\ 0, & \text{otherwise,} \end{cases} \tag{28}$$

and

$$\beta_{itjs}^{hp} = \begin{cases} \mu_{js}^{hp}, & \text{if } (j,s)\in\mathbf{V}_p^D,\ (i,t)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}, \\ 0, & \text{otherwise,} \end{cases} \tag{29}$$

when vehicle h travels through space-time arc $(i,t,j,s) \in \mathbf{A}$. Then an optimal solution approach based on DP is developed and described in Algorithm 1. Note that the algorithm is very efficient and takes a time complexity of $O(|\mathbf{H}||\mathbf{T}||\mathbf{I}|^2)$.

Subproblem 3 (Knapsack problem). Subproblem 3 only includes one subproblem with respect to variables **Z** and **W**, given as

$$\begin{aligned}\Pi(\boldsymbol{\lambda},\boldsymbol{\mu}) = \min_{\mathbf{Z},\mathbf{W}} \quad & \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^O} \left(c_{pi}^O + \lambda_{it}^{hp}\right) Z_{it}^{hp} \\ & + \sum_{p\in\mathbf{P}} \sum_{h\in\mathbf{H}} \sum_{(i,t)\in\mathbf{V}_p^D} \left(c_{pi}^D + \mu_{it}^{hp}\right) W_{it}^{hp},\end{aligned} \tag{30}$$

subject to (18)–(20), (23) and (24).

Note that Subproblem 3 can be regarded as a knapsack problem with item dependency constraints. That is, if we select an item from a subset, then another item must also be taken from a corresponding subset to the former one. More specifically, in Subproblem 3, one passenger must be delivered within the corresponding delivery space-time window by the vehicle that has already picked him/her up within his/her pickup space-time window. The major objective is to optimize the ride-share matching strategy, which involves the pickup

Step 1. Initialize:
(1) $ST_{it}^{h} = M, \forall (i,t) \in \mathbf{V}, h \in \mathbf{H}$ as the accumulated traveling cost of vehicle h at space-time vertex (i,t);
(2) $PN_{it}^{h} = 0$ and $PT_{it}^{h} = 0, \forall (i,t) \in \mathbf{V}, h \in \mathbf{H}$ to record the previous visiting node and time of each space-time vertex (i,t) within the routing path of vehicle h, respectively;
(3) $\mathbf{R}_h = \varnothing$ to record the space-time arcs within the least cost routing path of vehicle h;
Step 2. Do for each vehicle $h \in \mathbf{H}$
Step 2.1. Set $ST_{it_0}^{h} = 0, \forall i \in \mathbf{I}$;
Step 2.2. Do for each space-time arc $(i,t,j,s) \in \mathbf{A}$;
If $ST_{it}^{h} + \eta_{itjs}^{h} < ST_{js}^{h}$, then update $ST_{js}^{h} = ST_{it}^{h} + \eta_{itjs}^{h}$, and the previous node and time of space-time vertex (j,s) as $PN_{js}^{h} = i$ and $PT_{js}^{h} = t$;
Step 2.3. Select $ST_{i^*(t_0+T\delta)}^{h} = \min ST_{i(t_0+T\delta)}$;
Step 2.4. If $ST_{i^*(t_0+T\delta)}^{h} + c^{f} \leq 0$:
(1) Update the feasible solution of $\mathbf{Y}$ as $\overline{Y}_h = 1$;
(2) Track back from space-time vertex $(i^*, t_0 + T\delta)$ to the dummy origin vertex (i_O, t_O) via the values of PN_{it}^{h} and PT_{it}^{h}, and then record all the relative space-time arcs in set $\mathbf{R}_h$;
Step 3. Return set $\mathbf{R}_h$, then the relaxed solution of $\mathbf{X}$ can be obtained as:

$$\widehat{X}_{itjs}^{h} = \begin{cases} 1, & \text{if } (i,t,j,s) \in \mathbf{R}_h, \\ 0, & \text{otherwise,} \end{cases} \qquad \forall h \in \mathbf{H},\ (i,t,j,s) \in \mathbf{A}$$

Step 4. Return the relaxed solution of $\mathbf{X}$.

Algorithm 1: An optimal solution approach for solving subproblem $\mathrm{T}_h(\boldsymbol{\lambda}, \boldsymbol{\mu})$.

and delivery plans of passengers for each vehicle, to satisfy all the passengers' reservations with the minimum generalized serving cost. Note that the generalized pickup inconvenience cost of passenger p is actually the summation of $\{c_{pi}^{O}\}_{i \in \mathbf{I}_p^O}$ and corresponding multipliers $\{\lambda_{it}^{hp}\}_{h \in \mathbf{H}, (i,t) \in \mathbf{V}_p^O}$, while the generalized delivery inconvenience cost is the summation of $\{c_{pi}^{D}\}_{i \in \mathbf{I}_p^D}$ and multipliers $\{\lambda_{it}^{hp}\}_{h \in \mathbf{H}, (i,t) \in \mathbf{V}_p^D}$. Note that the structure of Subproblem 3 is simple enough with relative fewer variables to be solved efficiently by an integer solver (such as CPLEX) calling the branch-and-bound algorithm.

Furthermore, we can further obtain the optimal objective value of relaxed problem (25) by plugging the relaxed solution of $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, and $\mathbf{W}$ (respectively, denoted by $\widehat{\mathbf{X}} = \{\widehat{\mathbf{X}}_{itjs}^{h}\}_{h \in \mathbf{H}, (i,t,j,s) \in \mathbf{A}}$, $\widehat{\mathbf{Y}} = \{\widehat{\mathbf{Y}}_h\}_{h \in \mathbf{H}}$, $\widehat{\mathbf{Z}} = \{\widehat{\mathbf{Z}}_{it}^{hp}\}_{h \in \mathbf{H}, p \in \mathbf{P}, (i,t) \in \mathbf{V}}$, and $\widehat{\mathbf{W}} = \{\widehat{W}_{it}^{hp}\}_{h \in \mathbf{H}, p \in \mathbf{P}, (i,t) \in \mathbf{V}}$) into the following expression:

$$\Delta(\boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathrm{T}_h(\boldsymbol{\lambda}, \boldsymbol{\mu}) + \Pi(\boldsymbol{\lambda}, \boldsymbol{\mu}). \tag{31}$$

Note that the value of $\Delta(\boldsymbol{\lambda}, \boldsymbol{\mu})$ can serve as the lower bound of the optimal value of the primal problem according to the duality property of LR (see Geoffrion [28]).

4.2. Hybrid Method Based on Greedy Algorithm and Dynamic Programming. For one set of given $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$, if the relaxed solution of variables is found to be feasible to primal problem (14)–(24), then it is also the optimal solution. Otherwise, we have to use certain algorithms to construct a feasible solution based on the current relaxed solution. It is also a typical method adopted in many studies, such as Li and Ouyang [29], Marín [30], An et al. [31], Fu and Diabat [32], and Chen et al. [33].

One intuitive way to modify the relaxed solution is keeping the values of $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$, i.e., the assignments of passengers' reservations to vehicles, and then adjust the values of $\widehat{\mathbf{X}}$ and $\widehat{\mathbf{Y}}$ to reform the space-time trajectories of vehicles and fulfill the reservations by covering their corresponding pickup and delivery space-time vertexes. However, though the values of $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$ are subject to the passenger serving constraints and vehicle capacity constraints, the assignment plans may be still impossible to complete since the space-time flow balance constraints are not taken into consideration. An illustrative example is shown in Figure 5, where three reserved trips are assigned to vehicle h_1 according to the value of $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$. Assuming the capacity of h_1 as $C_{h_1} = 3$, we can see that only part of the reservations can be fulfilled by h_1 itself since it is obviously unrealistic to pick up passengers 1 and 3 from two different location at the same time. That is, we cannot obtain a practical routing strategy for h_1 based on the current assignment plan.

Therefore, to guarantee the efficiency of solution approach for the on-demand ridesharing system, we here particularly develop a three-phase hybrid method that can provide the good-quality feasible solutions of $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, and $\mathbf{W}$ (respectively, denoted by $\overline{\mathbf{X}} = \{\overline{X}_{itjs}^{h}\}_{h \in \mathbf{H}, (i,t,j,s) \in \mathbf{A}}$, $\overline{\mathbf{Y}} = \{\overline{Y}_h\}_{h \in \mathbf{H}}$, $\overline{\mathbf{Z}} = \{\overline{Z}_{it}^{hp}\}_{h \in \mathbf{H}, p \in \mathbf{P}, (i,t) \in \mathbf{V}}$, and $\overline{\mathbf{W}} = \{\overline{W}_{it}^{hp}\}_{h \in \mathbf{H}, p \in \mathbf{P}, (i,t) \in \mathbf{V}}$) adapted from $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$. Specifically, the first step of the hybrid solution (termed as screening procedure) is to screen the task schedule of h_1 and get rid of the vertexes that cannot be practically visited. Here we adopt the greedy algorithm to adjust the previous vehicle routing plan obtained from the values of $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$.

Specifically, to gradually capture the iteration process of the screening procedure, we define two lists: the vehicle tasks

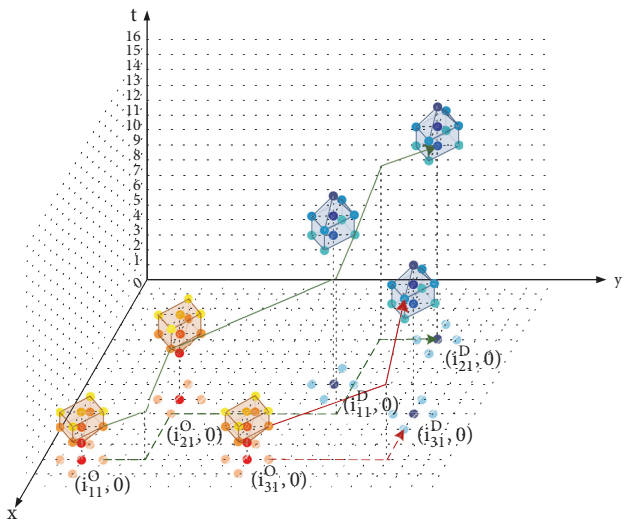


FIGURE 5: Illustration for the hybrid method.

list (denoted as $\mathbf{L}_h$ ($h \in \mathbf{H}$)) and the passengers reassigning list (denoted as $\mathbf{L}_r$). For example, $\mathbf{L}_{h_1}$ in Figure 6 is a list that records the pickup and delivery space-time vertexes to be visited by h_1. The two columns of $\mathbf{L}_{h_1}$, respectively, record the visiting nodes and times of the space-time vertexes and all the items are sorted in ascending order of their corresponding visiting times. Afterwards, we screen list $\mathbf{L}_{h_1}$ to eliminate the space-time vertexes that cannot be visited on time. Specifically, if node i_1 is assumed as the first node visited by vehicle h_1, then the next node i_2 can be visited on time if and only if

$$t_{i_1} + \bar{t}_{i_1 i_2} \leq t_{i_2}, \tag{32}$$

where t_{i_1}, t_{i_2}, and $\bar{t}_{i_1 i_2}$, respectively, represent the visiting time of node i_1 and i_2 and the least traveling time between i_1 and i_2. Note that since the travel time between two adjacent nodes (say i and j) can be easily obtained as $\bar{t}_{ij} = \bar{d}_{ij}/\bar{v}_{ij}$, where $\bar{d}_{ij}$, $\bar{t}_{ij}$, and $\bar{v}_{ij}$, respectively, denote the distance, travel time, and average speed through link (i, j). Then the least travel time between each two nodes in the transportation network can be further obtained by simply adopting a shortest path algorithm, such as Dijkstra's, Floyd-Warshall, or label setting algorithm. Since this area, which has been well studied, is not a focus of this paper, we here simply assume that the travel times between nodes are fixed during the operation cycle and we adopt the label setting algorithm to generate the node travel time matrix. If (32) is not fulfilled, space-time vertex (i_2, t_{i_2}) will not be visited on time and the reservation of corresponding passenger p_1 ($\widehat{Z}^{h_1 p_1}_{i_2 t_{i_2}}$=1) cannot be satisfied. Then we remove both the pickup and delivery space-time vertexes of p_1 from $\mathbf{L}_{h_1}$ to $\mathbf{L}_r$ and the index of passenger p_1 is also added to the third column of $\mathbf{L}_r$. As shown in Figure 6, we assume that t_{26}=3 and $\widehat{Z}^{h_1 3}_{61}$=$\widehat{W}^{h_1 3}_{128}$=1. We can see that space-time vertex (2,1) is the first vertex visited by h_1 and the following vertex (6,1) cannot be visited on time. Then vertexes (6,1) and (12,8) are both removed from $\mathbf{L}_{h_1}$ to $\mathbf{L}_r$ and passenger index 3 is added to the first two cells in third column of $\mathbf{L}_r$.

Afterwards, we turn to the next vertex in $\mathbf{L}_{h_1}$ and repeat the screen process above until all the vertexes are checked. Eventually, the whole screening procedure is also performed for the other occupied vehicles that satisfies $\sum_{p\in\mathbf{P}} \sum_{(i,t)\in\mathbf{V}^{O}_{p}} \widehat{Z}^{hp}_{it} > 0$ until all the tasks lists of these vehicles are screened. The detailed process is shown as Algorithm 2.

To fully satisfy the reservations of the passengers, the remaining passengers that recorded in $\mathbf{L}_r$ shall be reassigned to some of the unoccupied vehicles, i.e., the vehicles satisfying $\sum_{p\in\mathbf{P}} \sum_{(i,t)\in\mathbf{V}^{O}_{p}} \widehat{Z}^{hp}_{it} = 0$ ($h \in \mathbf{H}$). This process is also termed as reassigning procedure and we here propose a greedy based implementing algorithm. Specifically, we initially generate the common tasks list (denoted by $\mathbf{L}_c$) to record the all the pickup and delivery tasks obtained from $\mathbf{L}_r$. As shown in Figure 7, the first two columns contain the node and time to be visited. The third column records the indexes of the corresponding passengers to serve and the fourth one has the property tags representing the purpose of the tasks. Particularly, numbers 1 and 2, respectively, represent the pickup and delivery tasks. Similar to $\mathbf{L}_h$, all the space-time tasks in $\mathbf{L}_c$ are sorted in ascending order of the visiting times Then, we screen list $\mathbf{L}_c$ to select the tasks for h_2 with respect to the flow balance constraints and capacity constraints. For example, we assume that (i_2, t_{i_2}) is the first space-time vertex visited by h_2 and the following one (i_3, t_{i_3}) can also be visited on time if and only if (32) is satisfied. If so, the vehicle will first check if there are passengers to be delivered at this vertex. Note that all the delivered passengers are identified as fully served, so the corresponding tasks will also be removed from $\mathbf{L}_r$ to $\mathbf{L}_{h_2}$. Then, if there are also some passengers to be picked up, it is necessary to first make sure if there are enough spare seats. If so, some of the passengers will be picked up under the capacity limitation. Afterwards, if the vehicle is full of passengers, it will directly head to the closest passenger delivery vertex; otherwise, it will turn to the next vertex and repeat the serving process above. Such as the illustration example in Figure 7, where we set $C_{h_2} = 2$ and vertex (6,1) is the first vertex to visit as shown in $\mathbf{L}_c$. Since we assume that reservations of all the passengers are executable, vertex (12,8) can also be visited by h_2. Then passenger 3 is firstly dropped off at this vertex and the corresponding tasks are removed from $\mathbf{L}_r$ to $\mathbf{L}_{h_2}$. Since there are two spare seats left, only two passengers (say passenger 4 and 5) can be picked up. Afterwards, h_2 can pick up no more passengers and have to directly drop off the passengers onboard. Note that the delivery space-time vertex of passenger 4 (i.e., (21,15)) shall be visited earlier than that of passenger 5 (i.e., (28,20)), so we here choose (21,15) as the next vertex to visit and the serving process are repeated as above. Then turn to the next vertex if h_2 is not full or turn to the earliest vertex to drop off the passengers onboard otherwise. The iteration is carried on until all the tasks in $\mathbf{L}_c$ are screened. Eventually, this resign process will be repeated for some other unoccupied vehicles until all the user trips are fully covered, i.e., $\mathbf{L}_r = \varnothing$. The detailed process of resign procedure is shown as Algorithm 3.

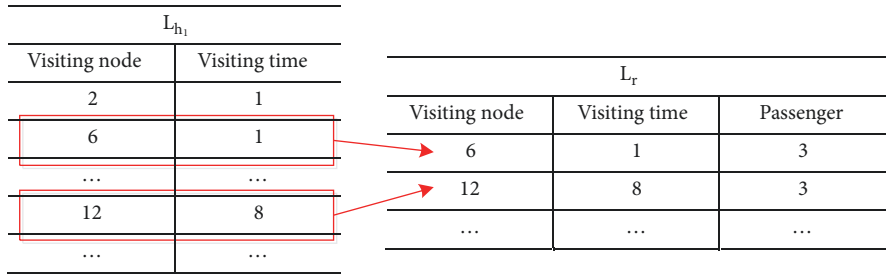


Figure 6: Illustration for the screening procedure.

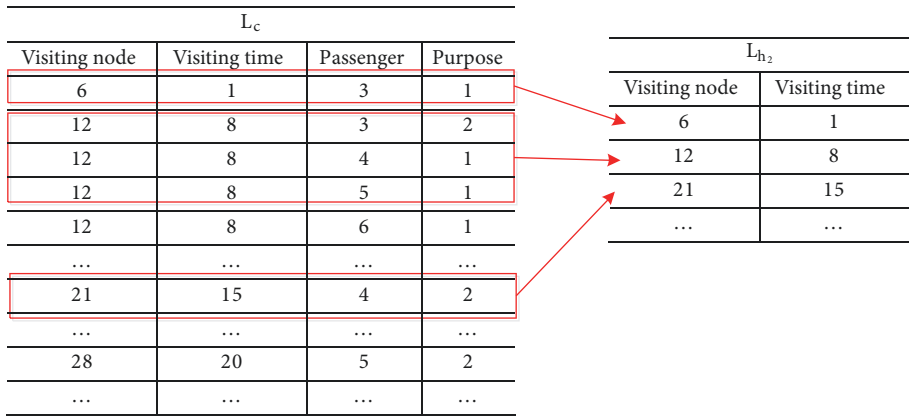


Figure 7: Illustration for the reassigning procedure.

Finally, a DP based algorithm similar to Algorithm 1 is further developed to generate $\overline{\mathbf{X}}$ and $\overline{\mathbf{Y}}$ based on the value of $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$, which is termed as routing procedure. In specific, we first calculate the generalized vehicle traveling cost for $h \in \mathbf{H}$ on space-time arc (i, t, j, s) (denoted by ρ^h_{itjs}) as

$$\rho^h_{itjs} = \begin{cases} -M, & \text{if } \overline{Z}^{hp}_{it} = 1 \vee \overline{W}^{hp}_{js} = 1, \ \forall p \in \mathbf{P}, \\ c^l_{ij}, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H}, \ (i, t, j, s) \in \mathbf{A}, \tag{33}$$

where M is a very large positive value. Then for each vehicle, steps 2.1-2.4 in algorithm 1 are called repeatedly to make the routing schedule that covers all the serving tasks with the least traveling cost. The specific process is presented as Algorithm 4. The DP based implementing algorithm takes a solution time of $O(|\mathbf{H}||\mathbf{T}||\mathbf{I}|^2)$, which is efficient and dominates the total computational time of the hybrid method.

Eventually, by plugging the value of $\overline{\mathbf{X}}$, $\overline{\mathbf{Y}}$, $\overline{\mathbf{Z}}$, and $\overline{\mathbf{W}}$ into primal objective function (14), we obtain a feasible objective value. It also serves as the upper bound of the optimal objective.

4.3. Updating Lagrangian Multipliers. If the upper bound obtained from the hybrid method is equal to the lower bound (31), then the corresponding feasible solution can be returned as the optimal solution according to the dual theory (Fisher [34]). Otherwise, a subgradient algorithm is usually used to update the multipliers $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ based on the optimality gap between the current upper and lower bounds, which can search for a better feasible solution and tighter lower bound, i.e.,

$$\max_{\boldsymbol{\lambda}, \boldsymbol{\mu} \geq 0} \Delta(\boldsymbol{\lambda}, \boldsymbol{\mu}). \tag{34}$$

It is actually a typical procedure that has been widely used in many previous studies for updating Lagrangian multipliers, such as Li [35], Zheng [36], Cui et al. [37], and Yin et al. [38]. By this, the optimality gap can be iteratively narrowed to eventually obtain a near-optimum solution with an acceptable tolerance or even zero in ideal cases, when the optimal solution is also achieved. Its main process is briefly described below.

Firstly, we initialize the iteration index $k = 0$, the step parameter $0 \leq \tau \leq 2$, and Lagrangian multipliers $(\lambda^{hp}_{it})^0 = (\mu^{hp}_{it})^0 = 0$. Then the relaxed solution can be obtained by solving problem (17) as Section 4.1. Accordingly, the feasible solution can be obtained by modifying the relaxed solution with the hybrid method described in Section 4.2. Afterwards, we obtain the step size based on the difference between the current lower and upper bounds, i.e.,

$$t_k = \frac{\tau^k \left(UB^k - LB^k(\boldsymbol{\lambda}, \boldsymbol{\mu})\right)}{\sum_{p \in \mathbf{P}} \sum_{h \in \mathbf{H}} \left(\left(A^p_h\right)^k + \left(B^p_h\right)^k\right)}, \tag{35}$$

Step 1. Initialize:
(1) $\overline{\mathbf{Z}}' = \widehat{\mathbf{Z}}$ and $\overline{\mathbf{W}}' = \widehat{\mathbf{W}}$ to record the phased results of $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$ obtained from screening procedure;
(2) $\mathbf{L}_r$ to record the passengers that need to be resigned;
(3) $\mathbf{L}_h$ to record the serving tasks of vehicle h;
Step 2. Do for each vehicle $h \in \mathbf{H}$
If $\sum_{p\in\mathbf{P}} \sum_{(i,t)\in\mathbf{V}_p^O} \widehat{Z}_{it}^{hp} > 0$, then do from the first vertex $(i_1, h_1) \in \mathbf{L}_h$ to the last one $(i_{|\mathbf{L}_h|}, h_{|\mathbf{L}_h|}) \in \mathbf{L}_h$:
Step 2.1. If $t_{i_k} + \bar{t}_{i_k i_{k+1}} \le t_{i_{k+1}}$ $(k = 1, 2, \ldots, |\mathbf{L}_h|)$, then update $k = k + 1$;
Step 2.2. Otherwise:
(1) Remove the elements that take the value of (i_{k+1}, t_{k+1}) out of list $\mathbf{L}_h$
(2) Do for each $p \in \{p \mid \widehat{Z}_{i_{k+1}t_{k+1}}^{hp} = 1 \vee \widehat{W}_{i_{k+1}t_{k+1}}^{hp} = 1\}$:
Update $\mathbf{L}_r$, $\overline{\mathbf{Z}}'$ and $\overline{\mathbf{W}}'$ as $\mathbf{L}_r = \mathbf{L}_r \cup \{p\}$, $\widehat{Z}_{it}^{\prime hp} = 0, \forall (i,t) \in \mathbf{V}_p^O$ and $\widehat{Z}_{it}^{\prime hp} = 0$, $\forall (i,t) \in \mathbf{V}_p^D$, respectively;
Step 3. Return $\mathbf{L}_h$, $\overline{\mathbf{Z}}'$ and $\overline{\mathbf{W}}'$.

Algorithm 2: Screening procedure for modifying $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$ to $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$.

Step 1. Initialize:
(1) $\mathbf{L}_p$ to record the pickup and delivery space-time vertexes of the passengers in $\mathbf{L}_r$;
(2) $\mathbf{L}_O^h$ to record the passengers that are picked up by vehicle h;
(3) The feasible solution $\overline{\mathbf{Z}} = \overline{\mathbf{Z}}'$ and $\overline{\mathbf{W}} = \overline{\mathbf{W}}'$;
Step 2. Do for each vehicle $h \in \mathbf{H}$
If $\sum_{p\in\mathbf{P}} \sum_{(i,t)\in\mathbf{V}_p^O} \widehat{Z}_{it}^{hp} = 0$:
Step 2.1. Update list $\mathbf{L}_O^h$ by adding passengers $p \in \{p \mid \sum_{h\in\mathbf{H}} \widehat{Z}_{i_1t_1}^{hp} = 1 \wedge p \in \mathbf{L}_r\}_{(i_1,t_1)\in\mathbf{L}_c}$ under the vehicle capacity limit C;
Step 2.2. Do from the first vertex $(i_1, t_1) \in \mathbf{L}_c$ to the last one $(i_{|\mathbf{L}_c|}, h_{|\mathbf{L}_c|}) \in \mathbf{L}_c$:
Step 2.2.1. If $|\mathbf{L}_O^h| < C$, then turn to the space-time vertex $(i_l, t_l) \in \mathbf{L}_c$, $(l = \min\{l \mid t_l > t_k\})$;
Step 2.2.2. Otherwise, turn to the space-time vertex $(i_l, t_l) \in \mathbf{L}_c$ $(l \in \{l \mid \sum_{h\in\mathbf{H}} \widehat{W}_{i_lt_l}^{hp} = 1, \forall p \in \mathbf{L}_O^h\})$;
Step 2.2.3. If $t_{i_k} + \bar{t}_{i_k i_l} \le t_{i_l}$ $(k = 1, 2, \ldots, |\mathbf{L}_h|)$:
If the set of passengers to be delivered at space-time vertex (i_l, t_l) is not empty, i.e., $\mathbf{P}_l^d = \{p \mid \sum_{h\in\mathbf{H}} \widehat{W}_{i_1t_1}^{hp} = 1 \wedge p \in \mathbf{L}_O^h\} \neq \varnothing$, then update $\mathbf{L}_O^h = \mathbf{L}_O^h - \mathbf{P}_l^d$ and $\mathbf{L}_h = \mathbf{L}_h \cup \{\mathbf{L}_O^h \cap \mathbf{P}_l^d\}$, and execute step 2.1 otherwise;
Step 2.2.4. Otherwise, if $\mathbf{P}_l^d = \{p \mid \sum_{h\in\mathbf{H}} \widehat{W}_{i_1t_1}^{hp} = 1 \wedge p \in \mathbf{L}_O^h\} \neq \varnothing$, then update $\mathsf{L}_\mathsf{h}^\mathsf{O} = \mathsf{L}_\mathsf{h}^\mathsf{O} \text{¡ } \mathsf{P}_\mathsf{l}^\mathsf{d}$;
Step 2.2.5. Update $k = l$, repeat Step 2.2.1 and 2.2.2 to locate the next space-time vertex for continuing reassigning process;
Step 2.3. Update $\mathbf{L}_r$, $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$ as $\mathbf{L}_r = \mathbf{L}_r - \mathbf{L}_h$, $\overline{Z}_{it}^{hp} = 1$ $(p \in \mathbf{L}_h$ and $\sum_{h\in\mathbf{H}} \widehat{Z}_{it}^{hp} = 1)$ and $\overline{W}_{it}^{hp} = 1$ $(p \in \mathbf{L}_h$ and $\sum_{h\in\mathbf{H}} \widehat{W}_{it}^{hp} = 1)$, respectively;
Step 2.4. If $\mathbf{L}_r = \varnothing$, terminate the iteration; otherwise, update $h = h + 1$ and continue the iteration;
Step 3. Return the feasible solution $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$.

Algorithm 3: Reassigning procedure for modifying $\widehat{\mathbf{Z}}$ and $\widehat{\mathbf{W}}$ to $\overline{\mathbf{Z}}$ and $\overline{\mathbf{W}}$.

where

$$\left(A_h^p\right)^k = \sum_{(i,t)\in\mathbf{V}_p^O} \left(\left(\widehat{Z}_{it}^{hp}\right)^k - \sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} \left(\widehat{X}_{itjs}^h\right)^k \right)^2,$$
$$\left(B_h^p\right)^k = \sum_{(i,t)\in\mathbf{V}_p^D} \left(\left(\widehat{W}_{it}^{hp}\right)^k - \sum_{(j,s)\in\mathbf{V}:(j,s,i,t)\in\mathbf{A}} \left(\widehat{X}_{jsit}^h\right)^k \right)^2, \quad (36)$$

and UB^k, LB^k, and τ^k are the best upper bound, the current lower bound, and the step parameter in iteration k, respectively. Note that if the lower bound does not improve within a certain number (say 5) of iterations, then we can slightly adjust the step size by updating the step parameter as $\tau^{k+1} = \tau^k/\theta$, where θ is a contraction ratio greater than 1.

Then the multipliers in iteration $k + 1$ can be obtained by

$$\left(\lambda_{it}^{hp}\right)^{k+1} = \max\left\{0, \left(\lambda_{it}^{hp}\right)^k + t_k\left(\left(\widehat{Z}_{it}^{hp}\right)^k - \sum_{(j,s)\in\mathbf{V}:(i,t,j,s)\in\mathbf{A}} \left(\widehat{X}_{itjs}^h\right)^k \right)\right\}, \quad (37)$$

Step 1. Initialize:

(1) $ST_{it}^{h} = M, \forall (i,t) \in \mathbf{V}, h \in \mathbf{H}$ as the accumulated traveling cost of vehicle h at space-time vertex (i,t);

(2) $PN_{it}^{h} = 0$ and $PT_{it}^{h} = 0, \forall (i,t) \in \mathbf{V}, h \in \mathbf{H}$ to record the previous visiting node and time of each space-time vertex (i,t) within the routing path of vehicle h, respectively;

(3) $\mathbf{R}_h = \varnothing$ to record the space-time arcs within the least cost routing path of vehicle h;

(4) $$\rho_{itjs}^{h} = \begin{cases} -M, & \text{if } \overline{Z}_{it}^{hp} = 1 \vee \overline{W}_{js}^{hp} = 1, \ \forall p \in \mathbf{P}, \\ c_{ij}^{l}, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H}, \ (i,t,j,s) \in \mathbf{A},$$
to present the generalized vehicle traveling cost for each vehicle on each space-time arc;

Step 2. Do for each vehicle $h \in \mathbf{H}$

Step 2.1 Set $ST_{it_0}^{h} = 0, \forall i \in \mathbf{I}$;

Step 2.2 Do for each space-time arc $(i,t,j,s) \in \mathbf{A}$;

If $ST_{it}^{h} + \rho_{itjs}^{h} < ST_{js}^{h}$, then update $ST_{js}^{h} = ST_{it}^{h} + \rho_{itjs}^{h}$, $PN_{js}^{h} = i$ and $PT_{js}^{h} = t$;

Step 2.3 Select $ST_{i^*(t_0+T\delta)}^{h} = \min ST_{i(t_0+T\delta)}$;

Step 2.4 Track back from space-time vertex $(i^*, t_0 + T\delta)$ to the dummy origin vertexes (i_O, t_O) via PN_{it}^{h} and PT_{it}^{h}, and then record all the relative space-time arcs in set $\mathbf{R}_h$;

Step 3 Return set $\mathbf{R}_h$, then the feasible solution of $\mathbf{X}$ and $\mathbf{Y}$ can be obtained as:

$$\overline{X}_{itjs}^{h} = \begin{cases} 1, & \text{if } (i,t,j,s) \in \mathbf{R}_h, \\ 0, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H}, \ (i,t,j,s) \in \mathbf{A}$$

$$\text{and } \overline{Y}_h = \begin{cases} 1, & \text{if } \sum_{p \in \mathbf{P}} \sum_{(i,t) \in \mathbf{V}_p^O} \widehat{Z}_{it}^{hp} > 0, \\ 0, & \text{otherwise}, \end{cases} \quad \forall h \in \mathbf{H};$$

Step 4 Return the feasible solution $\overline{\mathbf{X}}$ and $\overline{\mathbf{Y}}$.

Algorithm 4: DP based implementing algorithm for constructing feasible solutions of $\mathbf{X}$ and $\mathbf{Y}$.

$$\left(\mu_{it}^{hp}\right)^{k+1} = \max \left\{ 0, \left(\mu_{it}^{hp}\right)^{k} + t_k \left(\left(\widehat{W}_{it}^{hp}\right)^{k} - \sum_{(j,s) \in \mathbf{V}:(j,s,i,t) \in \mathbf{A}} \left(\widehat{X}_{jsit}^{h}\right)^{k} \right) \right\}, \tag{38}$$

The main LR procedures are summarized in Algorithm 5.

5. Numerical Examples

In Section 5, a series of numerical studies are conducted to test the performance of the proposed modeling framework and draw some managerial insight about the on-demand ridesharing operation strategies. In Section 5.1, we first demonstrate the effectiveness of our solution approach by comparing with that of CPLEX. Afterwards, multiscale experiments are performed to further illustrate the impact of different scales on the efficiency and accuracy of our solution approach. Then, we explore how the optimal vehicle operation strategies change over the key parameters in Section 5.2.

5.1. Experimental Results Comparing with CPLEX. To better demonstrate the performance of the model and solution approach proposed in our study, a set of illustrative numerical examples are conducted comparing with CPLEX. The option parameters of CPLEX are set to default value. All the experiments are implemented on a PC with 3.4 GHz CPU and 8GB RAM. The proposed solution algorithm and CPLEX are all performed on the MATLAB platform. Specifically, CPLEX is called by Yalmip (https://yalmip.github.io/), a free MATLAB toolbox for optimization modeling by interfacing external solvers such as CPLEX, Gurobi, and XPRESS. Since we use 5000 seconds as the solution threshold, the solution process will be terminated after that and return the final feasible solution and relevant gap. If no feasible solution is obtained within the time limit, we report the gap as "NA". Note that even though the PDPSW is rather complicated for adopting the concept of pickup and delivery space-time windows, our proposed ILP model is formulated compactly enough that CPLEX is able to solve medium-scale cases. The LR parameters are set as follows: τ=2, $\overline{\tau}$=10^{-3}, and θ=2. $\overline{K}$=200.

Initially, the proposed solution approach will be, respectively, examined in the 100-node grid, 400-node grid, and 900-node grid as shown in Figure 3. To simplify the experience without generality, we set $\overline{d}_{ij}$=$\overline{v}_{ij}$=1, $\forall (i,j) \in \mathbf{L}$. The fixed cost is set as c^f=\$30 within the operation cycle. We unify the capacity of each vehicle as $C_h = C = 2$ $(h \in \mathbf{H})$ and the traveling cost is assumed as $c_{ij}^{l} = c_l \times \overline{d}_{ij}$, $\forall (i,j) \in \mathbf{L}$, where c_l is constant coefficient and here we set c_l=\$0.1. The pickup and delivery space-time windows of each passenger's reservation are given as randomly distributing within the space-time network. In special, we assume that 2 timestamps and 5 nodes are contained in each pickup window, while 3 timestamps and 5 nodes are contained in each delivery window. The distribution pattern of space-time vertexes is

Step 1. Initialize:
(1) Lagrangian multipliers $(\lambda_{it}^{hp})^0 = (\mu_{it}^{hp})^0 = 0, \forall p \in \mathbf{P}, h \in \mathbf{H}, (i,t) \in \mathbf{V}$;
(2) Step parameters $0 \le \tau \le 2$;
(3) Iteration index $k := 0$;
(4) Best upper bound $UB^0 = +\infty$;
Step 2. Solve relaxed problem (25) with the solution approach described in Section 4.1, and then obtain the relaxed solution and the lower bound;
Step 3. If the lower bound does not improve in certain numbers of iterations, then update $\tau^{k+1} = \tau^k/\theta$ $(\theta > 1)$;
Step 4. Adapt the relaxed solution obtained above to the relative feasible solution by using the implementing algorithm proposed in Section 4.2, and then update UB^k to the relative feasible objective if UB^k is greater than it;
Step 5. Calculate t_k as equation (35);
Step 6. Update Lagrangian multipliers as equations (37) and (38);
Step 7. Terminate this algorithm if
(1) Optimality gap $(UB^k - LB^k(\boldsymbol{\lambda}, \boldsymbol{\mu}))/UB^k \le \varepsilon$, where ε is a predefined error tolerance;
(2) Step parameters $\tau^k < \overline{\tau}$, where $\overline{\tau}$ is the predefined lower threshold;
(3) Iteration index $k = \overline{K}$, where $\overline{K}$ is the predefined upper threshold;
Otherwise, update $k \longleftarrow k + 1$ and turn to Step 2;
Step 8. Return the current best upper bound as the final objective value and corresponding feasible solution and optimality gap.

Algorithm 5: LR algorithm for VRPSTW.

Table 1: Comparison between LR and CPLEX.

Instance	C_{total}		Solution time (sec)		Gap (%)	
	CPLEX	LR	CPLEX	LR	CPLEX	LR
100-20-20	548.4	548.4	418.6	775.4	<0.1	0.59
400-30-30	915.2	895.6	4629.3	1876.5	22.60	3.13
900-40-40	NA	1225.7	>5000	4817.4	NA	7.22

the same as the illustration example in Section 3.2. Besides, we uniformly set the inconvenience costs for picking up and delivering a passenger as $c_{pi}^O = c_p \times \overline{d}_{i_{p0}^O j}$, $i \in \mathbf{I}_p^O$, and $c_{pi}^D = c_p \times \overline{d}_{j i_{p0}^D}$, $i \in \mathbf{I}_p^D$, respectively, where c_p is constant coefficient and we set c_p=\$0.1.

The numerical results are shown in Table 1. In the instance column, the codes in each character string, respectively, denote the number of spatial nodes, the number of timestamps, and the number of passenger reservations. For example, instance 100-20-20 contains 100 nodes, 20 timestamps, and 20 passenger reservations. Note that, to ensure that there are always enough vehicles for dispatching during the iteration process of algorithm, we set the initial fleet size the same as the number of passengers, i.e., $|\mathbf{H}| = 30$. However, the number of vehicles that are eventually sent out for serving passengers (denoted as H_u) is determined as $H_u = \sum_{h \in \mathbf{H}} \overline{Y}_h$. In the result columns, the optimality gap of LR is obtained by $(UB^k - LB^k(\boldsymbol{\lambda}, \boldsymbol{\mu}))/UB^k \times 100\%$, where UB and LB denote the best upper and lower bound. Besides, we also present the average values of total system cost (denoted by C_{total}) obtained by these two solution approaches as a comparison.

We can see from Table 1 that both of the two solution approaches take longer solution time with the increase of instance scale. Nevertheless, the performance of CPLEX degenerates much faster than LR. For example, in solving the smaller scale instances like 100-20-20, CPLEX takes shorter solution time to obtain the same objective value and a smaller gap. Nevertheless, for the medium scale ones, we can see that LR can still solve the instances to solutions with smaller objective value (895.6) and gap (<4%) in a relatively short solution time (<1900s), while the performance of CPLEX turns to be inefficient and unreliable. Moreover, for the larger scale instances, CPLEX cannot obtain any feasible solution within 5000 seconds, yet LR can still obtain a good-quality solution (1225.7) with relatively small gap (<8%) in a reasonable solution time (<4900).

To further test how the performance of model (14)–(24) changes over the different problem scales, we perform a total of 27 instances in different scales, which is shown as Table 2. Particularly, we repeat each instance for 10 times with randomly generated passenger reservations involving the pickup and delivery space-time windows. Then the average values (short for Ave) and standard deviations (short for SD) of total system cost (denoted by C_{total}), vehicles dispatching cost (denoted by CF), vehicle routing cost (denoted by CT), passenger inconvenience cost (denoted by CP), solution time, and optimality gap of each instance are all present in Table 2. Besides, to show more details about the solutions, we specially select a case in instance 100-40-40 as an illustrative example

Table 2: Numerical results for multiscale instances.

Instance	Value	CF	CT	CP	C_{total}	Solution Time (sec)	Gap (%)
100-20-20	Ave	534.0	12.1	2.3	548.4	775.4	0.59
	SD	12.0	1.3	1.1	12.0	117.1	0.30
100-20-30	Ave	834.0	17.1	4.3	855.4	804.4	2.63
	SD	12.0	1.6	0.6	13.5	169.8	0.64
100-20-40	Ave	1086.0	22.3	4.3	1112.6	1264.5	3.05
	SD	22.4	1.2	1.2	22.2	120.1	0.21
100-30-20	Ave	504.0	13.6	2.1	519.7	671.6	2.82
	SD	12.0	0.4	1.2	11.7	218.8	0.12
100-30-30	Ave	750.0	18.7	3.5	772.2	1079.7	3.28
	SD	32.9	1.1	0.6	33.9	159.1	1.35
100-30-40	Ave	1002.0	25.1	5.3	1032.4	1768.7	5.66
	SD	24.0	0.7	1.1	23.1	170.0	1.53
100-40-20	Ave	480.0	14.1	2.7	496.9	1215.7	2.18
	SD	0.0	0.6	0.5	0.8	590.6	0.51
100-40-30	Ave	726.0	20.1	4.2	750.4	1370.2	3.20
	SD	39.8	0.9	0.6	36.4	273.3	0.67
100-40-40	Ave	996.0	27.0	5.5	1028.5	1433.8	3.97
	SD	29.4	1.0	1.5	28.2	217.1	0.62
400-20-20	Ave	580.0	15.0	2.1	597.0	1151.8	2.05
	SD	14.1	1.0	1.2	13.5	356.9	0.36
400-20-30	Ave	870.0	23.2	3.3	896.5	1279.0	2.91
	SD	0.0	1.2	1.5	0.9	332.9	0.07
400-20-40	Ave	1162.5	29.2	3.8	1195.5	1336.0	4.22
	SD	32.7	1.2	2.4	31.5	459.5	0.81
400-30-20	Ave	565.0	20.8	2.3	588.1	1552.4	2.83
	SD	20.6	1.6	1.2	19.1	408.4	0.63
400-30-30	Ave	858.0	33.0	4.7	895.6	1876.5	3.13
	SD	14.7	2.9	0.2	16.7	628.9	0.73
400-30-40	Ave	1140.0	42.0	4.7	1186.7	1963.0	4.18
	SD	19.0	1.8	1.3	19.5	399.8	0.52
400-40-20	Ave	540.0	25.2	2.1	567.3	1981.3	2.45
	SD	19.0	3.3	0.7	21.8	293.8	0.45
400-40-30	Ave	840.0	35.5	2.6	878.1	2415.5	3.83
	SD	26.8	2.9	1.4	26.5	867.6	0.83
400-40-40	Ave	1152.0	49.4	2.7	1204.0	4223.5	4.24
	SD	40.7	3.1	2.3	42.1	392.9	1.34
900-20-20	Ave	588.0	15.4	1.4	604.8	1018.5	2.98
	SD	14.7	1.5	1.4	14.5	373.9	0.24
900-20-30	Ave	882.0	24.2	2.1	908.3	1980.1	3.91
	SD	14.7	2.0	1.6	14.4	860.4	0.02
900-20-40	Ave	1188.0	33.3	2.2	1223.6	2234.3	4.80
	SD	14.7	1.6	2.0	14.1	731.8	0.37
900-30-20	Ave	588.0	24.7	1.2	613.8	1883.7	4.50
	SD	14.7	3.0	1.2	15.4	713.3	0.30
900-30-30	Ave	882.0	37.3	1.3	920.6	2153.5	5.67
	SD	14.7	2.5	1.4	13.5	541.2	0.17
900-30-40	Ave	1170.0	48.7	4.4	1223.0	3080.2	6.31
	SD	19.0	1.6	2.2	17.0	1348.9	0.45
900-40-20	Ave	588.0	31.6	0.4	619.9	2484.6	4.83
	SD	14.7	4.1	0.5	15.3	1096.1	0.07
900-40-30	Ave	888.0	47.9	1.2	937.1	4222.2	6.00
	SD	14.7	3.3	1.7	11.6	975.3	0.75
900-40-40	Ave	1158.0	63.8	3.9	1225.7	4817.4	7.22
	SD	14.7	4.5	2.1	15.2	1901.5	0.67

Table 3: Computational results of a case in instance 100-40-40.

# of Vehicle	# of Passenger	Pickup Location/Time	Delivery Location/Time	Travel Time
1	16	24/7	78/18	9
2	30	93/7	3/17	8
3	32	29/5	22/14	7
4	40	47/14	87/22	4
5	9	54/10	23/14	4
6	8	29/5	67/16	4
7	25	4/11	34/19	2
8	11	94/12	39/24	12
9	4	33/12	59/20	8
10	33	58/3	64/8	5
11	34	54/15	43/18	2
12	39	13/5	48/17	8
13	28	61/8	30/24	15
	38	71/5	47/16	
14	14	67/7	26/16	5
15	6	44/17	64/23	2
16	24	78/8	58/15	2
17	3	87/16	37/25	5
18	18	77/9	21/24	11
19	35	65/10	76/17	2
20	23	77/11	95/17	4
21	10	87/11	16/22	8
22	20	38/13	48/19	1
23	12	95/14	89/23	4
24	36	67/14	91/27	9
25	2	83/10	27/21	11
26	5	51/4	33/12	4
27	7	27/3	56/9	3
28	37	95/14	70/24	8
29	15	9/15	85/27	12
30	17	75/15	42/25	6
31	21	11/19	41/24	3
32	26	82/5	94/11	3
33	1	67/12	48/18	3
34	13	68/6	12/19	11
35	22	33/11	29/19	8
36	19	2/15	26/22	6
37	29	64/6	49/15	7
38	31	76/7	84/12	5
39	27	26/11	50/20	6

to present the number of dispatched vehicles, the number of passengers they serve, the specified pickup and delivery locations and times, and the travel time of each vehicle, which is shown as Table 3 in Appendix.

We can see that as the number of reservations increases, the solution time and the optimality gap strictly increase, yet such tendency is not so obvious when the scale of space-time network grows. For example, when the scale of instance increases from instance 100-20-20 to 100-30-20, the solution time even decreases from 775.4 to 671.6, and the similar case can also be seen from 100-30-40 to 400-30-40 that the optimality gap decreases from 5.66 to 4.18. Besides, we can also find that the dispersion of solution time in each instance is much more evident with respect to the randomly distributed passenger reservations, while the optimality gap is barely affected. This demonstrate that the SD of solution time is sensitive to the space-time distribution of passenger reservations and shall be paid more attention in setting solution threshold for real world cases.

Furthermore, we can also find how the variations of instance scale affect the optimal vehicles operation strategies. For example, as the number of passengers' reservations grows, *CF, CT,* and C_{total} significantly increase and *CP* slightly increases as well. This can be easily explained that when the number of reservations increases, more vehicles will be needed with a longer total routing distance. And it also incurs more inconvenience cost. Nevertheless, an interesting phenomenon is found that for the same transportation network, when the time horizon becomes longer, *CT* keeps increasing while *CF* and C_{total} dramatically decrease. This is probably because that shorter time horizon will aggregate the distribution of passenger reservations in time. That is, for the same number of reservations, the space-time windows will become denser in time as the time horizon reduces. Then it may be more difficult to compete a ridesharing since too little time left for vehicles to pick up (or deliver) another passenger after picking up (or delivering) one. Conversely, as the time horizon increases, it may be easier for passengers to stagger their travel times and share rides. Thus, by properly designing the vehicle routing plan, part of the vehicles could be saved. That is, a small increase for vehicle routing cost can dramatically save a relatively larger amount of fixed investment. This can also well explain the reason why larger transportation network causes the growth of *CF* and *CT*. For the same number of reservations, larger network will apparently decentralize the distribution of passengers' space-time windows in space. Then it may be more difficult to compete a ridesharing since too long distance for vehicles to reach another pickup (or delivery) node after picking up (or delivering) a passenger. Thus, more vehicles are needed to serve the same number of passengers, which also incurs longer total traveling distance in total. Nevertheless, we also find that the increasing rate of *CF* generally slows down with the growth of transportation network, while that of *CT* presents the opposite trend. This implies that although longer traveling distance is needed to serve passengers in a larger transportation network, a properly designed vehicle routing plan may partly offset the growth trend of vehicle dispatching cost.

Besides, we can see that *CF* dominates the variation of C_{total} since the fixed cost of dispatching vehicles occupies a larger proportion of the total system cost. Note that it is practically reasonable to set the unit initial fixed cost (i.e., \$30 within the operation cycle) larger than the unit vehicle routing cost (i.e., \$0.1 for a vehicle traveling unit distance) and passenger inconvenience cost (i.e., \$0.1 for walking unit distance or waiting unit time). For the SD of cost components, we can see that *CF* is more affected by the distribution of passenger reservations and apparently dominate the SD of C_{total}. Nevertheless, it can also be seen that when the scale of transportation network raises to 900, the SD of *CF* is generally kept to 14.7. It is probably because that when the scale of transportation network is big enough, the dispersion of the number of dispatched vehicles is stable and predictable, despite the variations of time horizon and number of passenger reservations.

5.2. Model Sensitive Analysis. In this section, we show a set of more detailed sensitivity results on how the user demand and key parameter c_p affect the optimal vehicle operation strategies. Particularly, the experiments in this section are based on the 100-node network with 40 timestamps. The values of benchmark parameters are set the same as Section 5.1.

To demonstrate the efficiency of the ridesharing system with flexible pickup and delivery locations (termed as space-time windows pattern and short for STW) in comparison to that of the conventional one (termed as time windows pattern and short for TW), we simultaneously present the optimal objectives of STW and TW, which is shown as Figures 8(a) and 8(b). In particular, C_{total}/*CF*/*CT*/*CP*-STW, respectively, denote the total cost and all cost components in STW, while C_{total}/*CF*/*CT*-TW, respectively, denote that in TW (except *CP* for constantly being zero in TW). Note that since the proposed STW is essentially the generalization of TW, we can simply convert STW to TW by reducing the space window to only the origin or destination node. Then we see that when the number of passengers' reservations grows from 0 to 140, all the cost components of both STW and TW significantly increase (which has been well explained in Section 5.1), while *CP* keeps a relatively slower increasing rate. Besides, we also notice that the values of C_{total}, *CF*, and *CT* in TW are all bigger than that in STW. This implies that even though there is additional cost *CP*, replacing TW with STW, which actually does not reduce the service quality too much, will obviously improve the system economy. Furthermore, as the number of reservations continues to grow from 140 to 200, the difference between *CF*-PDPTW and *CF*-PDPSW keeps increasing, which dominates that of the total system cost. And this trend can be found more obvious between *CT*-TW and *CT*-STW. This suggests that when there are more reservations during the operation cycle, adopting STW is a more efficiency and environmentally friendly solution.

Furthermore, an important issue that evaluates the service quality is the inconvenience cost of passengers, which also affects the optimal vehicle operation strategies. To this end, we specially verify the variations of total cost and all the cost components with the magnitudes of coefficient c_p. The results are shown in Figures 8(c) and 8(d). As c_p grows from 0 to 1, *CP* increases dramatically and dominates the growth of C_{total}. Nevertheless, *CF* and *CT* do not change too much and hence the increasing rate of C_{total} is much smaller than that of *CP*. This implies that properly increasing the inconvenience costs, which does not significantly raises the total system cost, may be a compromise solution to balance the economy of system operation and the service quality. In practice, the operator can make some rebates to the passengers to recover the loose of passengers for the extra walking distance or waiting time.

6. Conclusion

This paper studied the on-demand ridesharing problem with flexible pickup and delivery locations for the TNC systems. By employing a space-time network representation, the considered problem is formulated as a PDPSW that integrates the ride-share matching strategy and vehicle routing plan. In

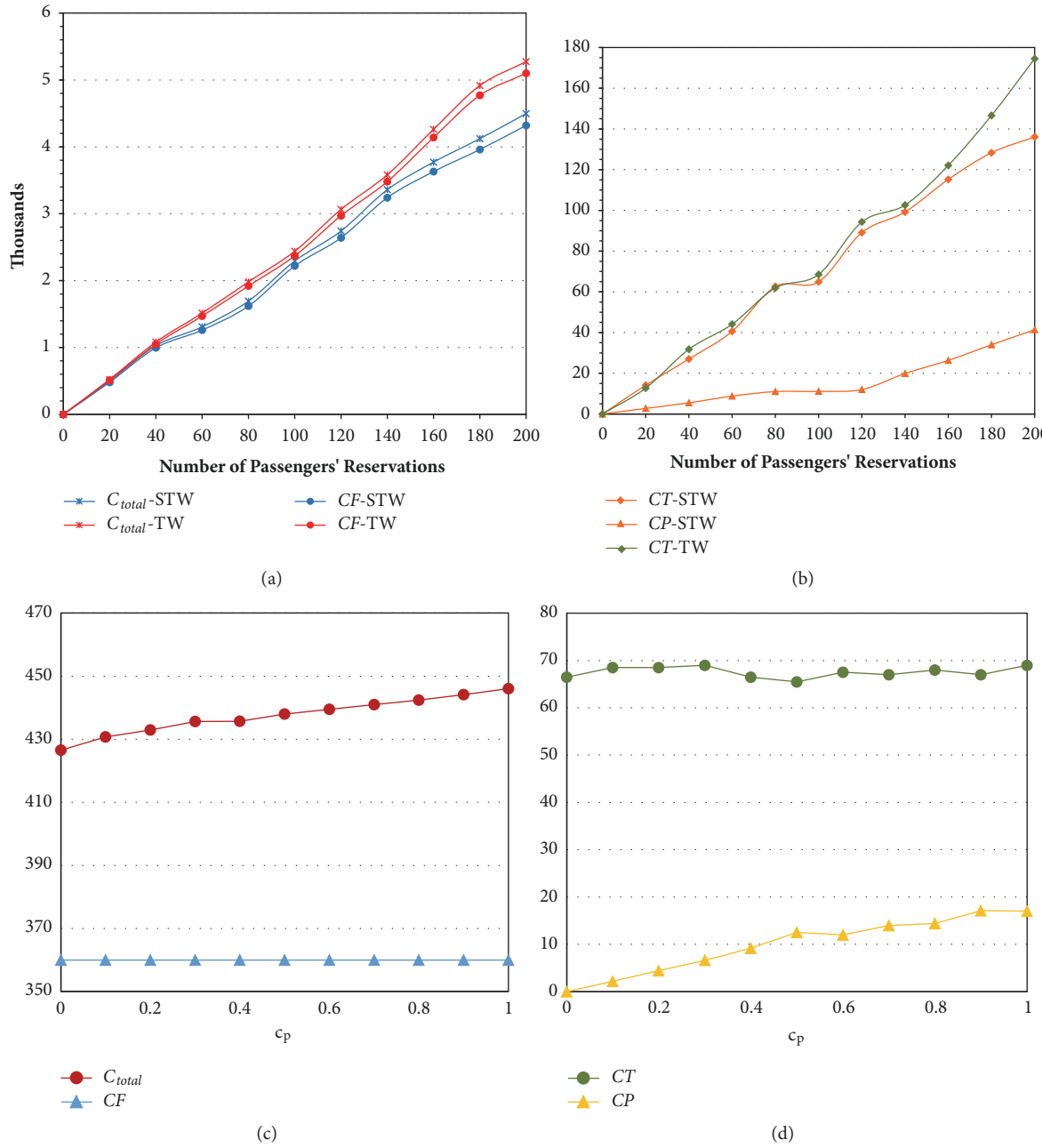


Figure 8: Sensitivity analyses on number of passengers' reservations and c_p.

particular, the concept of space-time window is introduced to expand the time windows of passengers in the spatial dimension. To balance the system operation cost and service quality, we specially took the inconvenience cost of passengers into consideration in the mathematical model. Since the proposed model is difficult to solve, a customized solution approach was proposed based on the LR algorithm framework. This solution algorithm has been evidenced to be efficient and accurate by a set of numerical examples comparing with CPLEX. Moreover, the numerical results also revealed some managerial insights on how the key parameters affect the optimal operation design results. For example, we found that adopting the flexible serving strategy, i.e., properly adjusting the pickup and delivery locations, can evidently reduce the system cost and improve the passenger-to-vehicle matching rate without incurring too much extra inconvenience cost. This validates the importance and necessity of flexible serving strategy in balancing the system cost and service quality. Besides, we also saw that increasing the inconvenience costs will not affect the total system cost. This implies that offering the passengers with proper rebates in practice will somehow improve the service quality without bring up the system cost too much.

The presented model can be further extended in the following directions: Initially, this study assumes that the passengers' reservations are predetermined, and this can

be extended for some cases where the reservations are temporally raised or canceled. Thus, an interesting extension of this study is investigating the PDPSW under the stochastic or dynamic conditions. Moreover, time-dependent travel time shall be considered since the link travel time within urban area could be changeable and valuable for only short-term prediction. It will be interesting to verify the benefits of adopting flexible serving strategy under more complex transportation conditions.

Appendix

To show more details about the computational results of multiscale instances, we specially select a case in instance 100-40-40 to present the numbers of dispatched vehicles, the numbers of passengers they serve, the specified pickup and delivery locations and times, and the travel time of each vehicle in Table 3. We can see that only one vehicle (# 13) completes a ridesharing service. That is because the number of passengers is relatively small with respect to the scale of the space-time network. This can be demonstrated in Figures 8(a) and 8(b) that when the number of passengers is small (especially no more than 40), the number of dispatched vehicles is generally proportional to the number of passengers with relatively fewer ridesharing services.

Data Availability

The datasets used to support the findings of this study are available upon request from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. U1734210, 51578199, 51478151, and 61790573), Beijing Municipal Natural Science Foundation (no. L171007), State Key Laboratory of Rail Traffic Control and Safety (no. RCS2018ZT011), and Fundamental Research Funds for the Central Universities under Grant 2017RC004.

References

[1] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transportation Research Part B: Methodological*, vol. 57, pp. 28–46, 2013.

[2] M. W. Savelsbergh and M. Sol, "The general pickup and delivery problem," *Transportation Science*, vol. 29, no. 1, pp. 17–29, 1995.

[3] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, "Static pickup and delivery problems: a classification scheme and survey," *TOP*, vol. 15, no. 1, pp. 1–31, 2007.

[4] Y. Dumas, J. Desrosiers, and F. Soumis, "The pickup and delivery problem with time windows," *European Journal of Operational Research*, vol. 54, no. 1, pp. 7–22, 1991.

[5] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

[6] H. Li and A. Lim, "A Metaheuristic for the Pickup and Delivery Problem with Time Windows," *International Journal on Artificial Intelligence Tools*, vol. 12, no. 02, pp. 173–186, 2003.

[7] R. Baldacci, E. Bartolini, and A. Mingozzi, "An exact algorithm for the pickup and delivery problem with time windows," *Operations Research*, vol. 59, no. 2, pp. 414–426, 2011.

[8] H. Hosni, J. Naoum-Sawaya, and H. Artail, "The shared-taxi problem: Formulation and solution methods," *Transportation Research Part B: Methodological*, vol. 70, pp. 303–318, 2014.

[9] I. Küçükoğlu and N. Öztürk, "A differential evolution approach for the vehicle routing problem with backhauls and time windows," *Journal of Advanced Transportation*, vol. 48, no. 8, pp. 942–956, 2014.

[10] T.-Y. Hu and C.-P. Chang, "A revised branch-and-price algorithm for dial-a-ride problems with the consideration of time-dependent travel cost," *Journal of Advanced Transportation*, vol. 49, no. 6, pp. 700–723, 2015.

[11] M. Mahmoudi and X. Zhou, "Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state-space-time network representations," *Transportation Research Part B: Methodological*, vol. 89, pp. 19–42, 2016.

[12] D. Brownstone and T. F. Golob, "The effectiveness of ridesharing incentives. Discrete-choice models of commuting in Southern California," *Regional Science & Urban Economics*, vol. 22, no. 1, pp. 5–24, 1992.

[13] E. Ferrari, R. Manzini, A. Pareschi, A. Persona, and A. Regattieri, "The Car Pooling Problem: Heuristic Algorithms Based on Savings Functions," *Journal of Advanced Transportation*, vol. 37, no. 3, pp. 243–272, 2003.

[14] W. Herbawi and M. Weber, "A genetic and insertion heuristic algorithm for solving the dynamic ridematching problem with time windows," in *GECCO'12—Proceedings of the Fourteenth International Conference on Genetic and Evolutionary COMputation*, pp. 385–392, ACM, New York, 2012.

[15] H. Xu, F. Ordõñez, and M. Dessouky, "A traffic assignment model for a ridesharing transportation market," *Journal of Advanced Transportation*, vol. 49, no. 7, pp. 793–816, 2015.

[16] D. Pelzer, J. Xiao, D. Zehe, M. H. Lees, A. C. Knoll, and H. Aydt, "A Partition-Based Match Making Algorithm for Dynamic Ridesharing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2587–2598, 2015.

[17] M. Stiglic, N. Agatz, M. Savelsbergh, and M. Gradisar, "The benefits of meeting points in ride-sharing systems," *Transportation Research Part B: Methodological*, vol. 82, pp. 36–53, 2015.

[18] N. Masoud and R. Jayakrishnan, "A decomposition algorithm to solve the multi-hop Peer-to-Peer ride-matching problem," *Transportation Research Part B: Methodological*, vol. 99, pp. 1–29, 2017.

[19] L. C. Tong, L. Zhou, J. Liu, and X. Zhou, "Customized bus service design for jointly optimizing passenger-to-vehicle assignment and vehicle routing," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 451–475, 2017.

[20] N. Kliewer, T. Mellouli, and L. Suhl, "A time-space network based exact optimization model for multi-depot bus scheduling," *European Journal of Operational Research*, vol. 175, no. 3, pp. 1616–1627, 2006.

[21] L. Yang and X. Zhou, "Constraint reformulation and a Lagrangian relaxation-based solution algorithm for a least expected time path problem," *Transportation Research Part B: Methodological*, vol. 59, pp. 22–44, 2014.

[22] L. Tong, X. Zhou, and H. J. Miller, "Transportation network design for maximizing space-time accessibility," *Transportation Research Part B: Methodological*, vol. 81, pp. 555–576, 2015.

[23] Q. Zhen and S. Jing, "Train rescheduling model with train delay and passenger impatience time in urban subway network," *Journal of Advanced Transportation*, vol. 50, no. 8, pp. 1990–2014, 2016.

[24] P. Li, P. Mirchandani, and X. Zhou, "Solving simultaneous route guidance and traffic signal optimization problem using space-phase-time hypernetwork," *Transportation Research Part B: Methodological*, vol. 81, no. 1, pp. 103–130, 2015.

[25] L. Zhang, Q. Meng, and T. Fang Fwa, "Big AIS data based spatial-temporal analyses of ship traffic in Singapore port waters," *Transportation Research Part E: Logistics and Transportation Review*, 2017.

[26] J. Yin, T. Tang, L. Yang, Z. Gao, and B. Ran, "Energy-efficient metro train rescheduling with uncertain time-variant passenger demands: an approximate dynamic programming approach," *Transportation Research Part B: Methodological*, vol. 91, pp. 178–210, 2016.

[27] L. Yang and X. Zhou, "Optimizing on-time arrival probability and percentile travel time for elementary path finding in time-dependent transportation networks: Linear mixed integer programming reformulations," *Transportation Research Part B: Methodological*, vol. 96, pp. 68–91, 2017.

[28] A. M. Geoffrion, "Lagrangean relaxation for integer programming," *Mathematical Programming*, no. 2, pp. 82–114, 1974.

[29] X. Li and Y. Ouyang, "Reliable traffic sensor deployment under probabilistic disruptions and generalized surveillance effectiveness measures," *Operations Research*, vol. 60, no. 5, pp. 1183–1198, 2012.

[30] Á. G. Marín, "Airport taxi planning: Lagrangian decomposition," *Journal of Advanced Transportation*, vol. 47, no. 4, pp. 461–474, 2013.

[31] S. An, N. Cui, Y. Bai, W. Xie, M. Chen, and Y. Ouyang, "Reliable emergency service facility location under facility disruption, en-route congestion and in-facility queuing," *Transportation Research Part E: Logistics and Transportation Review*, vol. 82, pp. 199–216, 2015.

[32] Y.-M. Fu and A. Diabat, "A Lagrangian relaxation approach for solving the integrated quay crane assignment and scheduling problem," *Applied Mathematical Modelling: Simulation and Computation for Engineering and Environmental Systems*, vol. 39, no. 3-4, pp. 1194–1201, 2015.

[33] B. Y. Chen, W. H. Lam, and Q. Li, "Efficient solution algorithm for finding spatially dependent reliable shortest path in road networks," *Journal of Advanced Transportation*, vol. 50, no. 7, pp. 1413–1431, 2016.

[34] M. L. Fisher, "The Lagrangian relaxation method for solving integer programming problems," *Management Science*, vol. 27, no. 1, pp. 1–18, 1981.

[35] X. Li, "An integrated modeling framework for design of logistics networks with expedited shipment services," *Transportation Research Part E: Logistics and Transportation Review*, vol. 56, pp. 46–63, 2013.

[36] H. Zheng, "Optimization of bus routing strategies for evacuation," *Journal of Advanced Transportation*, vol. 48, no. 7, pp. 734–749, 2014.

[37] J. Cui, M. Zhao, X. Li, M. Parsafard, and S. An, "Reliable design of an integrated supply chain with expedited shipments under disruption risks," *Transportation Research Part E: Logistics and Transportation Review*, vol. 95, pp. 143–163, 2016.

[38] J. Yin, L. Yang, T. Tang, Z. Gao, and B. Ran, "Dynamic passenger demand oriented metro train scheduling with energy-efficiency and waiting time minimization: Mixed-integer linear programming approaches," *Transportation Research Part B: Methodological*, vol. 97, pp. 182–213, 2017.