IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 25, 2017, accepted November 20, 2017, date of publication November 28, 2017,
date of current version February 28, 2018.

*Digital Object Identifier 10.1109/ACCESS.2017.2778221*

# R-Sharing: Rendezvous for Personalized Taxi Sharing

**YAN LYU[1], VICTOR C. S. LEE[2], (Member, IEEE), CHI-YIN CHOW[2], (Senior Member, IEEE), JOSEPH KEE-YIN NG[1], (Senior Member, IEEE), YANHUA LI[3], (Senior Member, IEEE), AND JIA ZENG[4], (Senior Member, IEEE)**

[1]Computer Science Department, Hong Kong Baptist University, Hong Kong
[2]Department of Computer Science, City University of Hong Kong, Hong Kong
[3]Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA 01609, USA
[4]Huawei Noahs Ark Lab, Hong Kong

Corresponding author: Yan Lyu (lvyanly@gmail.com)

This work was supported in part by the HKBU Research Centre for Ubiquitous Computing, in part by the HKBU Institute of Computational and Theoretical Studies, and in part by The Innovation and Technology Commission of the HK SAR Government under the Innovation and Technology Fund under Project ITP/048/14LP. The work of C.-Y. Chow was supported by CityU SRG under Project 7004890. Y. Li was supported in part by NSF CRII under Grant CNS-1657350 and in part a Research Grant from Pitney Bowes Inc.

**ABSTRACT** Dynamic taxi sharing is an effective approach to reducing travel cost and conserving energy resources. Existing taxi sharing frameworks fail to consider personal preferences of passengers on taxi-sharing and unable to group them with compatible preferences for generating the optimal sharing schedule. In this paper, we propose a novel taxi-sharing framework called R-Sharing to provide a personalized rendezvous-sharing service. It enables passengers to set their preferences on four essential sharing experience, i.e., walking distance, waiting time, travel fare, and extra travel time. Given a sharing request, R-Sharing searches the optimal set of nearby companions with compatible personal preferences and similar destination directions, recommends a rendezvous point for them to meet, and plans the shortest sharing route, such that these passengers' probability of accepting the sharing schedule is maximized. Specifically, a companion candidate searching algorithm is proposed for searching nearby potential candidates to sharing a taxi for the request. To select the optimal subset of candidates and generate their optimal sharing schedule, we propose an exact taxi-sharing scheduling algorithm, in which, the rendezvous point is set by considering passengers' personal preference on walking distance and road network factors, and the shortest sharing route is planned by dynamic programming. Further, a heuristic sharing scheduling algorithm is developed to improve the efficiency. Extensive experiments are conducted using a one-month taxi trajectory data set collected in Nanjing, China. Experimental results show that R-Sharing is not only effective in terms of achieving better sharing ratio and reduced total travel distance, but also provides superior sharing experiences.

**INDEX TERMS** Taxi sharing, location-based services, urban computing.

## I. INTRODUCTION

Dynamic taxi sharing has become a popular transport alternative in cities across the world. Such location-based services respond and satisfy passengers' real-time sharing requests by scheduling sharing rides with others going in similar directions. They have become an effective approach to reducing energy consumptions while preserving the benefits of individuals. Let's consider a simplified example: if three nearby individual passengers who plan to take taxis to the downtown area can share the same taxi, they can save two-third travel fare and waiting time for three separate taxis; and more importantly, it can also reduce gas or electricity consumption.

Most existing dynamic taxi sharing services [1]–[4] usually respond a passenger's sharing request by dispatching a taxi that is not fully occupied to pick up the passenger within a certain time period, with the objective of minimizing the overall travel distance or travel time. However, they have two weaknesses.

- Passengers have different preferences on taxi-sharing. For example, some may prefer saving more money by sharing a taxi with more companions while some may

prefer less extra travel time by sharing a taxi with fewer companions. Existing systems fail to take personal preferences of passengers into consideration in planning sharing schedules.
- They also fail to provide the optimal sharing schedule as they usually dispatch a taxi that is not fully occupied to an en-route sharing request. Given a new request, they can only provide a sub-optimal sharing schedule bounded by the current locations and traveling directions of taxis and the destinations of passengers already on board. Nevertheless, a sharing schedule could be changed or even degraded when a taxi needs to pick up a new passenger en route. For instance, the so-called shortest travel time or route could be changed during a ride. Moreover, the on-board passengers may have extended and unpredictable travel distance and time.

To this end, we are motivated to propose a new taxi sharing framework, called **R-Sharing**, which provides a taxi-sharing service from the perspective of passengers. Specifically, given a taxi-sharing request, R-Sharing searches the optimal set of nearby companions with compatible personal preferences and similar destination directions, recommends a rendezvous point which is easy to get a vacant taxi and requires short walking distance for them to meet, and plans the shortest sharing route, such that these passengers' probability of accepting the sharing schedule is maximized.

R-Sharing significantly differs from existing dynamic taxi sharing systems by providing new sharing services with two major unique features. (1) R-Sharing enables passengers to set their preferences on four essential sharing experience, namely, (i) walking distance, (ii) waiting time for a taxi, (iii) travel fare, and (iv) extra travel time, which will be considered in searching companions and planning sharing schedules. (2) R-Sharing groups companions with compatible personal preferences to provide the optimal sharing schedule consisting of the shortest route before getting on a taxi. The passengers will be delivered based on the prearranged sharing schedule, and they will not encounter any unexpected en-route sharing request during their trips. Thus the service quality, especially the expected arrival time at the destination of each passenger, can be guaranteed. To the best of our knowledge, this is the first work to provide a taxi-sharing service that gives passengers the initiative of setting their personal sharing preferences, groups passengers with compatible preferences and generates their optimal sharing schedule.

In general, the main contributions of this work can be summarized as follows.

- We propose a novel dynamic taxi sharing system called R-Sharing with a rendezvous-sharing strategy. R-Sharing searches the optimal nearby companions for a sharing request, recommends the optimal rendezvous point, and plans their shortest sharing route based on their personal preferences, such that their probability of accepting the sharing schedule is maximized.
- A companion candidate searching algorithm is proposed for finding candidates who are likely to satisfy a taxi-sharing request.
- An exact taxi-sharing scheduling algorithm is developed to generate the optimal sharing schedule (comprising of the optimal set of companions, the optimal rendezvous point and the shortest sharing route) for a sharing request, such that the sharing probability is maximized. To improve the efficiency, a heuristic sharing scheduling algorithm is also proposed for generating the optimal sharing scheduling.
- Extensive experiments are conducted based on one-month taxi trajectories collected in Nanjing, China. Experimental results show that R-Sharing achieves higher sharing ratio, longer reduced travel distance, lower average searching and scheduling time compared with other methods, and provides superior sharing experiences with short walking distance, waiting time, extra travel time and low travel fare.

The rest of this paper is organized as follows. Section II reviews the related work. Section III presents the framework of R-Sharing and defines the rendezvous-sharing problem. Section IV describes the companion candidate searching algorithm. Section V elaborates the exact and heuristic sharing scheduling algorithms. The experiment settings are presented in Section VI and experimental results are analyzed in Section VII. Finally, we conclude this paper in Section VIII.

## II. RELATED WORK

In this section, we present background knowledge and the state-of-the-art techniques on dynamic ride sharing systems.

### A. EXISTING DYNAMIC RIDE SHARING SYSTEMS

Dynamic ride sharing systems, such as Uber,[1] Lyft[2] and Didi Chuxing,[3] have become popular transport alternatives in cities across the world. These systems offer real time platforms to both private car owners and the passengers for sharing rides in similar directions. They usually dispatch a taxi (or a private car) that is not fully occupied to a ride request, and schedule specified pick-up and drop-off locations and time sequences for the sharing ride, with the objective of finding the best match between the drivers and passengers. These sharing services, however, only provide sub-optimal sharing schedule due to the unpredictable en-route sharing requests. The on-board passengers may encounter unexpected en-route sharing requests and thus could suffer unpleasant sharing experience with extended travel time during their trips.

By contrast, another type of taxi-sharing systems, such as Bandwagon,[4] also spring up. These systems are designed to shrink taxi lines by instantly matching the passengers waiting in line with similar directions and letting them share rides.

[1] http://www.uber.com
[2] http://www.lyft.com
[3] http://www.didichuxing.com/en/
[4] http://www.bandwagon.io/

The optimal groups of companions can be easily found from the waiting line. However, these systems can only be used at taxi stands where passengers should line up waiting for taxis.

### B. DYNAMIC RIDE SHARING TECHNIQUES

Existing works usually formulate the dynamic taxi-sharing as the dynamic pick-up and delivery problem (DPDP) [5] or dial-a-ride problem (DARP) [6]–[8], with the objective of minimizing overall travel distances, or maximizing the number of ride-matches or benefits for both passengers and drivers. Several factors, including waiting time, travel fare, and detour distance, have been considered as the constraints.

Since the traditional solutions [9]–[12] to DPDP and DARP are not designed to deal with the modern situations where millions of ride requests are generated on the fly, many heuristic algorithms [4], [13]–[16] have been developed for fast matching drivers and passengers and scheduling sharing routes efficiently. For example, Ma *et al.* [7], [8] proposed a taxi-sharing system that schedules proper taxis to pick up real-time sharing requests under time, capacity and monetary constraints. The taxis on road network are searched using an efficient spatio-temporal indexing structure. Ota *et al.* [17] proposed a data-driven and scalable dynamic taxi sharing framework with an efficient optimization algorithm that is linear in the number of trips. Huang *et al.* [18] developed kinetic tree algorithms to schedule the dynamic en-route requests and adjust routes on-the-fly. Pelzer *et al.* [19] designed a dynamic ride sharing framework, in which a partition-based match making algorithm is developed to detect passengers with similar directions. The above algorithms mainly focus on fast matching between drivers and passengers and efficiently scheduling sharing rides. However, they fail to consider the personal preferences of passengers and unable to group them for sharing a taxi with the optimal sharing schedule.

To the best of our knowledge, this is the first study that matches passengers with similar directions and schedules their sharing rides from the perspective of passengers. The service quality can be guaranteed because the passengers will be grouped with compatible preferences and recommended with a nearby rendezvous point which is easy to get a taxi. They will be delivered based on the prearranged sharing schedule and will not encounter any unexpected en-route sharing request during their trips.

## III. SYSTEM MODEL

In this section, we first present the framework of R-Sharing, describe the sharing probability model and the pricing strategy, and then formulate the rendezvous sharing problem. Table 1 shows all notations used throughout this paper.

### A. FRAMEWORK OF R-SHARING

We first present the three key concepts in R-Sharing, i.e., sharing request $q$, request queue $Q$ and sharing schedule $\phi$, and then describe the framework and workflow of R-Sharing.

**TABLE 1. Key notations.**

| Symbol | Description |
|---|---|
| $q$ | A taxi-sharing request |
| $T_q$ | The time when the request $q$ is submitted |
| $O_q$ | Origin location of request $q$ |
| $D_q$ | Destination location of request $q$ |
| $\tau_q$ | Maximum waiting time for finding companions |
| $\delta_{walk}$ | Threshold of walking distance to rendezvous point |
| $\delta_{wait}$ | Threshold of waiting time for a vacant taxi |
| $\delta_{fare}$ | Threshold of travel fare ratio |
| $\delta_{extraT}$ | Threshold of extra travel time ratio |
| $w_{walk}$ | A passenger's preference weight on walking distance |
| $w_{wait}$ | A passenger's preference weight on waiting time |
| $w_{fare}$ | A passenger's preference weight on travel fare |
| $w_{extraT}$ | A passenger's preference weight on extra travel time |
| $Q$ | A request queue |
| $R$ | A rendezvous point |
| $k$ | Number of passengers sharing a taxi |
| $\phi$ | A sharing schedule $\phi : R \rightarrow D_{q_1} \cdots \rightarrow D_{q_k}$ |
| $F(i)$ | The taxi fare charged for the $i$-th passenger in a sharing schedule |
| $\gamma$ | The regular taxi fare per kilometer |
| $dist(l_i, l_j)$ | Road network distance from location $l_i$ to $l_j$ |

#### 1) SHARING REQUEST

A *request* $q$ is a passenger's request for finding taxi-sharing companions. Each $q$ is associated with six attributes: $\langle T_q, O_q, D_q, \tau_q, W_q, \Delta_q \rangle$, where $T_q$ is the timestamp of the request time, $O_q$ is the origin location, $D_q$ is the destination location and $\tau_q$ is the maximum waiting time for finding companions. $\Delta_q = \langle \delta_{walk}, \delta_{wait}, \delta_{fare}, \delta_{extraT} \rangle$ and $W_q = \langle w_{walk}, w_{wait}, w_{fare}, w_{extraT} \rangle$ are the sets of user-specified thresholds and weights indicating passengers' personal preference regarding to the following four factors: (1) walking distance to the rendezvous point ($\delta_{walk}$ and $w_{walk}$), (2) waiting time for a vacant taxi at the rendezvous point ($\delta_{wait}$ and $w_{wait}$), (3) the ratio of the travel fare by taxi-sharing to the original travel fare without taxi-sharing ($\delta_{fare}$ and $w_{fare}$), and (4) the ratio of the extra travel time caused by taxi-sharing to the original travel time without taxi-sharing ($\delta_{extraT}$ and $w_{extraT}$). Specifically, each threshold is the maximum value of each factor a passenger can accept, e.g., $\delta_{walk}$ is the maximum walking distance and $\delta_{fare}$ is the maximum fare ratio. Each preference weight indicates the importance of each factor to this passenger. The value of each weight ranges from 0 to 1, and the sum of the four weights equals to 1.

#### 2) REQUEST QUEUE

A *request queue* $Q$ is a list of requests that have not had any companions to share taxis yet. If a new request cannot find any companions from the existing $Q$, it is inserted into $Q$. A request $q \in Q$ is removed from $Q$ if (1) $q$ can share a taxi with a new request, or (2) its waiting time for finding companions exceeds $\tau_q$.

#### 3) SHARING SCHEDULE

A *sharing schedule* $\phi$ comprises of a rendezvous point $R$ for all passengers sharing a taxi, and a sequence of their destination locations, i.e., $R \rightarrow D_{q_1} \rightarrow D_{q_2} \cdots \rightarrow D_{q_k}$,

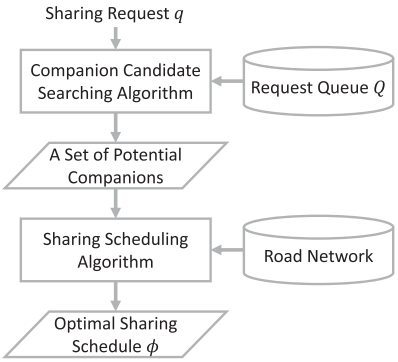



**FIGURE 1.** Framework of R-Sharing.

indicating their order to be delivered. Note that the number of passengers should not exceed the taxi capacity. For the sake of clear exposition, each query here only represents one passenger, and the taxi capacity is set to four, (i.e., $k \leq 4$). Nevertheless, our approach can be applied to a request with multiple passengers.

#### 4) FRAMEWORK AND WORKFLOW

The framework of R-Sharing is illustrated in Fig. 1. R-Sharing takes a passenger's sharing request $q$ as an input and generates the optimal sharing schedule $\phi$ with a companion candidate searching algorithm and a sharing scheduling algorithm. Specifically, R-Sharing processes a sharing request by five steps:

1. A passenger submits a sharing request $q$ to R-Sharing.
2. The *companion candidate searching algorithm* first searches companion candidates that are likely to satisfy $q$ from the *request queue* $Q$.
3. The *sharing scheduling algorithm* generates the optimal schedule $\phi$ with the companions selected from the set of potential candidates.
4. The system informs the optimal sharing schedule to each involved participant.
5. If all participants accepts the optimal sharing schedule, the system removes their requests from the *request queue* $Q$. Otherwise, the request $q$ will be inserted to $Q$. $Q$ also removes the requests whose waiting time for companions exceeds their time window (i.e., $\tau_q$).

### B. SHARING PROBABILITY

There are four factors that affect whether a passenger wants to share a taxi with others, namely, (1) walking distance to the rendezvous point, (2) waiting time for a vacant taxi, (3) fare, and (4) extra travel time. Besides, different passengers weigh these four factors differently. For example, some people may concern more about the fare, but others may prefer short walking distance or waiting time. Therefore, we propose a probabilistic model to estimate the probability of passengers accepting a sharing schedule by considering these four factors with personalized weights.

In the following, given a sharing schedule $\phi$ with $k$ passengers, we first estimate the probability of the $i$-th passenger involved in $\phi$, $i \leq k$, and then calculate the probability of all $k$ passengers accepting $\phi$.

#### 1) $i$-TH SHARING PROBABILITY

To estimate the probability of $i$-th passenger accepting the sharing schedule $\phi$ based on the four factors, we first define a utility function to integrate the four factors with personalized weights. Let $x_{walk}$ be the walking distance to the rendezvous point, $x_{wait}$ be the waiting time for a vacant taxi at the rendezvous point, $x_{fare}$ be the ratio of the travel fare by taxi-sharing to the original travel fare without taxi-sharing, and $x_{extraT}$ be the ratio of the extra travel time caused by taxi-sharing to the original travel time without taxi-sharing. The utility function, denoted by $U_\phi^i$, is defined as:

$$\begin{aligned} U_\phi^i = {} & w_{walk} f(x_{walk}, \delta_{walk}) + w_{wait} f(x_{wait}, \delta_{wait}) \\ & + w_{fare} f(x_{fare}, \delta_{fare}) + w_{extraT} f(x_{extraT}, \delta_{extraT}), \end{aligned} \quad (1)$$

where $f(x, \delta)$ is an evaluation with respect to a single factor. Note that the value of $x$ should not exceed its corresponding threshold $\delta$. Also, a smaller $x$ (namely, walking distance, waiting time, fare ratio and extra travel time ratio) can achieve a higher probability of the passenger accepting the sharing schedule. Therefore, the evaluation $f(x, \delta)$ is defined by measuring how far the value $x$ is from its threshold $\delta$, i.e.,

$$f(x, \delta) = 1 - \frac{x}{\delta}, \quad (2)$$

where $x \leq \delta$ and the value of $f(x, \delta)$ ranges from 0 to 1. Since the sum of the four weights should not exceed 1, the value of the utility function $U_\phi^i$ also ranges from 0 to 1.

We estimate the probability, denoted as $P(i, \phi)$, for the $i$-th passenger accepting the sharing schedule $\phi$ with the Logistic function [20], i.e.,

$$P(i, \phi) = \frac{1}{1 + \exp(-\beta U_\phi^i)}, \quad (3)$$

where $\beta$ is a tunable parameter.

The property of this probability estimator (Eq. 3) is threefold: (1) $P(i, \phi)$ ranges from 0.5 to 1, because $U_\phi^i$ ranges from 0 to 1 for the sharing schedules that can provide sharing services within the threshold of each factor. (2) As the values of the four factors decrease, $P(i, \phi)$ increases. (3) In an extreme case where the sharing schedule provides a perfect service, i.e., no walking distance, no waiting time, extreme low fare and no extra travel time, the utility value $U_\phi^i$ will be close to 1 and the estimator will generate a probability that is close to 1.

#### 2) SHARING PROBABILITY

The probability of all passengers accepting sharing schedule $\phi$, referred to as *sharing probability* and denoted by $P(\phi)$, can be estimated by the product of all the probabilities of each

passenger accepting this schedule, i.e.,

$$P(\phi) = \prod_{i=1}^{k} P(i, \phi). \tag{4}$$

### C. PRICING SCHEME

We set the taxi fare for each passenger involved in a sharing schedule to be proportional to distances when they travel alone without sharing a taxi. Specifically, given a sharing schedule from the rendezvous point $R$ to the destination locations of $k$ passengers $R \rightarrow D_{q_1} \cdots \rightarrow D_{q_k}$, let *TotalDist* be the total travel distance of this schedule and $\gamma$ be the regular fare per kilometer, then the total fare of all passengers should pay is $TotalDist \times \gamma$. We assign the total fare to all passengers in proportion to their travel distances if they travel alone, i.e., the fare for sharing a taxi charged to the $i$-th passenger, denoted by $F(i)$, is calculated as

$$F(i) = \frac{dist(R, D_{q_i})}{\sum_{j=1}^{k} dist(R, D_{q_j})} \times TotalDist \times \gamma, \tag{5}$$

where $dist(R, D_{q_i})$ is the road network distance that the $i$-th passenger travel directly from $R$ to $D_{q_i}$ without sharing a taxi with others. For example, for three passengers A, B and C with a schedule of $R \rightarrow D_{q_1} \rightarrow D_{q_2} \rightarrow D_{q_3}$, suppose the total travel distance for this schedule is 3 km and the regular fare is \$10 per km; hence, $TotalDist \times \gamma = 30$. Assume the travel distances when A, B and C travel alone are 1, 1.5, and 2.5 km, respectively. The fare costs for sharing a taxi charged to A, B, and C are $1/(1 + 1.5 + 2.5) \times 30 = 6$, $1.5/(1+1.5+2.5)\times 30 = 9$, and $2.5/(1+1.5+2.5)\times 30 = 15$, respectively. Namely, they can save 80% of travel fare by sharing a taxi compared with traveling alone.

### D. PROBLEM DEFINITION

We formally define our problem as follows: given a sharing request $q$ for finding $k-1$ companions to share a taxi ($k \leq 4$), we aim to search the companions and plan a sharing schedule $\phi$ for them, such that the sharing probability $P(\phi)$ is maximized.

## IV. COMPANION CANDIDATE SEARCHING

In order to find companions with the optimal sharing schedule for a request, we first need to select a small set of candidates which are likely to satisfy the request. Intuitively, a companion candidate should satisfy the following constraints:

- *Time constraint*. The candidate is waiting for someone to share a taxi, i.e., his/her waiting time is within the maximum waiting time for finding companions.
- *Origin constraint*. The origin location of the candidate should be close to the request's origin location, in order to avoid long walking distance.
- *Fare constraint*. The taxi fares charged to both the candidate and the request in a sharing schedule should be lower than the fares when they travel alone.

Since requests whose waiting time for companions exceeds their time windows are removed from the request queue, all pending requests satisfy the time constraint. Then candidates satisfying the *origin constraint* are selected from the waiting queue by checking whether the distance from their origin location to the request's origin location is within a distance threshold $\delta_{walk}^{q_r} + \delta_{walk}^{q_c}$, where $\delta_{walk}^{q_r}$ and $\delta_{walk}^{q_c}$ are the walking distance thresholds of the new request $q_r$ and the candidate $q_c$, respectively. However, it is hard to determine whether these candidates satisfy the *fare constraint* without planning a sharing schedule for them. Here, we first explore which candidates can satisfy the fare constraint, and then present our candidate searching steps.

### A. FARE CONSTRAINT CHECKING STRATEGY

We use an example to explain this strategy. Consider sharing schedule $R \rightarrow D_{q_1} \rightarrow D_{q_2}$, where $R$ is the rendezvous point, $D_{q_1}$ and $D_{q_2}$ are the first and second destinations, respectively. Based on our pricing scheme (Eq. 5), the taxi fare charged to the $i$-th passenger in the sharing schedule $F(i)$ can be calculated as

$$F(i) = \frac{dist(R, D_{q_i}) \times (dist(R, D_{q_1}) + dist(D_{q_1}, D_{q_2})) \times \gamma}{dist(R, D_{q_1}) + dist(R, D_{q_2})},$$

where $i \in \{1, 2\}$, and $dist(R, D_{q_1}) + dist(D_{q_1}, D_{q_2})$ is the total travel distance (i.e., *TotalDist* in Eq. 5) in this schedule. Based on the fare constraint, $F(i)$ should not be higher than the fare without sharing a taxi, i.e., $dist(O_{q_i}, D_{q_i}) \times \gamma$, where $O_{q_i}$ is the origin location of $q_i$. Hence, the fare constraint can be interpreted as $F(i) \leq dist(O_{q_i}, D_{q_i}) \times \gamma$. Note that the rendezvous point $R$ is within a short walking distance to each origin location $O_{q_i}$, we have $dist(R, D_{q_i}) \approx dist(O_{q_i}, D_{q_i})$. Thus, $F(i) \leq dist(O_{q_i}, D_{q_i}) \times \gamma$ can be easily converted to $dist(D_{q_1}, D_{q_2}) \leq dist(O_{q_2}, D_{q_2})$. As a result, this sharing schedule can satisfy the fare constraint if $dist(D_{q_1}, D_{q_2}) \leq dist(O_{q_2}, D_{q_2})$.

To check whether a candidate $q_c$ and a request $q_r$ can form a sharing schedule that satisfies $dist(D_{q_1}, D_{q_2}) \leq dist(O_{q_2}, D_{q_2})$, we can test whether the distances between their destinations $dist(D_{q_c}, D_{q_r})$ and $dist(D_{q_r}, D_{q_c})$ satisfy one of the following conditions,

$$dist(D_{q_c}, D_{q_r}) \leq dist(O_{q_r}, D_{q_r}), \quad \text{or} \tag{6}$$

$$dist(D_{q_r}, D_{q_c}) \leq dist(O_{q_c}, D_{q_c}). \tag{7}$$

If Eq. 6 is satisfied, there exist a schedule $R \rightarrow D_{q_c} \rightarrow D_{q_r}$ that fulfills the *fare constraint*. And if Eq. 7 is satisfied, the schedule $R \rightarrow D_{q_r} \rightarrow D_{q_c}$ also meets the *fare constraint*.

### B. CANDIDATE SEARCHING STEPS

As depicted in Algorithm 1, given a new request $q_r$, we start searching candidates from the request queue $Q$ since $Q$ keeps the requests that satisfy the *time constraint*. For each request $q_c \in Q$, if the origin distance $dist(O_{q_c}, O_{q_r})$ or $dist(O_{q_r}, O_{q_c})$ is smaller than the distance threshold $\delta_{walk}^{q_r} + \delta_{walk}^{q_c}$, then $q_c$ satisfies the *origin constraint* (Line 3). If $q_c$ also satisfies Eq. 6 or Eq. 7, then $q_c$ satisfies the *fare constraint* and will be considered as a companion candidate. (Lines 5 to 7.)

**Algorithm 1** Companion Candidate Searching

**Input:** (1) A new request $q_r$, and (2) request queue $Q$
**Output:** A set of candidate $C$.

```
for each waiting request q_c ∈ Q do
    //Check the origin constraint
    if dist(O_{q_c}, O_{q_r}) or dist(O_{q_r}, O_{q_c}) ≤ δ^{q_r}_{walk}+δ^{q_c}_{walk} then
        //Check the fare constraint
        if dist(D_{q_c}, D_{q_r}) ≤ dist(O_{q_r}, D_{q_r}) or
        dist(D_{q_r}, D_{q_c}) ≤ dist(O_{q_c}, D_{q_c}) then
            C ← q_c
        end if
    end if
end for
```

## V. TAXI SHARING SCHEDULING

Given a new request $q$ and the set of companion candidates, R-Sharing should be able to (1) plan the optimal sharing schedule for a subset of candidates to be sharing a taxi with $q$, and (2) search the best subset of candidates for $q$, such that their optimal sharing schedule achieves the highest sharing probability based on Eq. 4.

In this section, we first elaborate how to plan the optimal sharing schedule comprising of the optimal rendezvous point (Section V-A) and the shortest sharing route (Section V-B), given a subset of companion candidates for $q$. Both exact and heuristic sharing scheduling algorithms are then proposed to generate the optimal subset of companion candidates with the optimal sharing schedule (Sections V-C and V-D).

### A. RENDEZVOUS POINT SETTING

Given a subset of companion candidates (i.e., passengers) to be sharing a taxi with request $q$, we explore the optimal location for their rendezvous point. Intuitively, a rendezvous point should achieve the shortest walking distances from all their origin locations. However, in reality, this point may not be the best one due to the following reasons:

- Different passengers may have different weights on walking distance. A good rendezvous point should be closer to the passengers with larger weights on walking distance, but further to the passengers with smaller weights on walking distance, in order to satisfy the personal preferences of all the passengers.
- Road network factors such as real-time location of vacant taxis and traffic conditions should be considered in choosing a rendezvous point, so that passengers can easily get a vacant taxi at their rendezvous point.

To consider passengers' personalized weights on walking distances, we introduce a measurement on *passengers' satisfaction on a rendezvous point with regard to walking distance* as follows. Given the weight and the threshold on walking distance, denoted by $w^i_{walk}$ and $\delta^i_{walk}$ for passenger $i$, the passenger's satisfaction on a rendezvous point with walking distance $x_{walk}$, denoted by $U^i_{walk}$ can be calculated as

$$U^i_{walk} = w^i_{walk}(1 - \frac{x^i_{walk}}{\delta^i_{walk}}).$$

As the distance $x^i_{walk}$ to the rendezvous point increases, the satisfaction $U^i_{walk}$ decreases.

To incorporate the road network factors in choosing a rendezvous point, we adopt the probability of getting a vacant taxi, denoted by $P_{taxi}(l_j, t)$, with regard to the rendezvous point $l_j$ at time $t$ for measuring how easily the passengers can get a taxi. In our simulation, we utilize the historical trajectory data set to estimate $P_{taxi}(l_j, t)$ [21], and present the details in Section VI-B. Note that more sophisticated estimation models that consider more factors such as real-time locations of taxis and traffic conditions can be implemented, however, this is not the focus of our work.

With these two measurements, the rendezvous point setting problem can be formulated as follows. For a set of $k$ passengers, an optimal location is chosen as their rendezvous point, denoted by $R$, such that the average product of the walking distance satisfaction to location $l_j$, (i.e., $U^i_{walk}, 1 \le i \le k$) and the probability of getting a vacant taxi at location $l_j$ at time $t$, (i.e., $P_{taxi}(l_j, t)$) is maximized, i.e.,

$$R = \text{argmax}_{l_j \in \mathcal{L}} \frac{1}{k} \sum_{i=1}^{k} U^i_{walk} \times P_{taxi}(l_j, t), \qquad (8)$$

where $\mathcal{L}$ is the set of locations where passengers can be picked up by a taxi in a city.

To find the optimal $R$ from $\mathcal{L}$, we first search the nearby pick-up points within the threshold $\delta^i_{walk}$ from the origin location for each passenger $i$, and then get the intersection of sets of the nearby pick-up points. For each potential pick-up points in the intersection, we calculate the probability of getting a vacant taxi, as well as the walking distance satisfaction for each passenger $i$, and select the optimal point $R$ with the maximum average product.

### B. SHARING ROUTE PLANNING

Given the subset of companion candidates to be sharing a taxi with request $q$, here, we present how to plan their shortest sharing route that starts from the rendezvous point and visits each destination exactly once to deliver these passengers.

Note that searching the shortest route that visits each destination exactly once is a typical travel salesman problem (TSP), which is NP-hard [22], [23]. We developed a dynamic programming algorithm to generate the shortest sharing route.

Let $\mathcal{D}$ be the set of destinations. We set the rendezvous point $R$ as the origin for all possible routes. For a subset of $\mathcal{S} \subseteq \mathcal{D}$ and $D_i \in \mathcal{S}$, let $(\mathcal{S}, D_i)$ be the set of possible routes that start at $R$, visit each destination in $\mathcal{S}$ exactly once, and finish in $D_i$. Let $Route(\mathcal{S}, D_i)$ be the shortest route, and $Len(\mathcal{S}, D_i)$ be the length of $Route(\mathcal{S}, D_i)$. We obtain the shortest route $Route(\mathcal{S}, D_i)$ by picking the best second-to-last stop $D_k$, such that the route length from $R$ to $D_k$, (i.e.,

$Len(\mathcal{S} - \{D_i\}, D_k))$, plus the road network distance between the last two destinations, i.e., $len(D_k, D_i)$ is minimized:

$$Len(\mathcal{S}, D_i) = \min_{D_k \in \mathcal{S}: k \neq i,} Len(\mathcal{S} - \{D_i\}, D_k) + len(D_k, D_i). \tag{9}$$

$D_k^*$ is denoted as the best second-to-last stop; hence,

$$Route(\mathcal{S}, D_i) = (Route(\mathcal{S} - \{D_i\}, D_k^*), D_i). \tag{10}$$

By iteratively searching the intermediate shortest route for each subset $\mathcal{S} \subseteq \mathcal{D}$ with Eqs. 9 and 10, the shortest route can be generated after the set $\mathcal{D}$ is searched, i.e.,

$$\begin{aligned} Route(\mathcal{D}) &= Route(\mathcal{D}, D_i^*) \text{ and} \\ D_i^* &= \text{argmin}_{D_i \in \mathcal{D}} Len(\mathcal{D}, D_i). \end{aligned} \tag{11}$$

Note that when searching the intermediate shortest route $Route(\mathcal{S}, D_i)$ by Eqs. 9 and 10, the shortest distance $len(D_k, D_i)$ from $D_k$ to $D_i$ need to be computed on the large-scale road network. In practice, such shortest path queries can be answered using some speedup path computation techniques such as contraction hierarchies [24] and path caching [25].

### C. EXACT SHARING SCHEDULING ALGORITHM

For different subsets of companion candidates to share a taxi with $q$, the rendezvous points as well as the sharing routes are different. In order to find the optimal candidate subset with the highest sharing probability, we propose an exact scheduling algorithm, which traverses every possible candidate subset (with the maximum set size of three, due to the taxi capacity limit) for $q$, searches their optimal rendezvous point, schedules their shortest sharing route, and calculates the sharing probability based on the rendezvous point and the sharing route. The optimal candidate subset is generated if their rendezvous point and sharing route can form a sharing schedule with highest sharing probability.

The exact sharing scheduling algorithm is depicted in Algorithm 2. Given a sharing request $q$ and the set of companion candidates, the algorithm traverses every subset $\mathcal{D}$ of the candidates' destinations $\mathcal{T}$. For each subset $\mathcal{D}$, the algorithm first searches the optimal rendezvous point for $q$ and the passengers whose destinations are in $\mathcal{D}$ (Line 3). It then plans the shortest sharing route by iteratively searching the intermediate shortest route $Route(\mathcal{S}, D_i)$ for each $\mathcal{S} \subseteq \mathcal{D}$ with Eqs. 9 and 10 (Lines 4 to 12). The sharing probability is then calculated based on the optimal rendezvous point $R$ and the shortest sharing route $Route(\mathcal{D})$ (Line 13). Finally, the algorithm generates the optimal sharing schedule with the highest sharing probability.

In this solution, for a request with $N$ candidates, there is a total of $\sum_{s=1}^{3} \binom{N}{s}$ possible combinations for sharing a taxi with $q$, (the taxi capacity is set to 4 and $s = |\mathcal{D}|$). For each combination, it takes linear time to search the optimal rendezvous point. The shortest route can be obtained by iteratively searching a total number of $2^{s+1}(s+1)$ states (i.e., $(\mathcal{S}, D_i)$), and each state requires linear time to get the intermediate shortest route $Route(\mathcal{S}, D_i)$. Therefore, the total searching space is $\sum_{s=1}^{3} 2^{s+1}(s+1)^2\binom{N}{s}$.

**Algorithm 2** Exact Sharing Scheduling

**Input:** (1) A new request $q$ with the destination $D_q$, and (2) A set of candidates with their destinations $\mathcal{T}$.
**Output:** Optimal sharing schedule $\phi$.

1: **for** each subset $\mathcal{D} \subseteq \mathcal{T}$, $|\mathcal{D}| \leq 3$ **do**
2: $\quad\mathcal{D} \leftarrow D_q$
3: $\quad$Search the optimal rendezvous point $R$ for the set of passengers whose destinations are in $\mathcal{D}$ by Eq. 8
4: $\quad$**for** $s = 1$ to 3 **do**
5: $\quad\quad$**for** each subset $\mathcal{S}$ of size $s$, $\mathcal{S} \subseteq \mathcal{D}$ **do**
6: $\quad\quad\quad\mathcal{S} \leftarrow R$, $Len(\mathcal{S}, R) = \infty$
7: $\quad\quad\quad$**for** $D_i \in \mathcal{S}$ **do**
8: $\quad\quad\quad\quad$Obtain the shortest route length $Len(\mathcal{S}, D_i)$ and the route $Route(\mathcal{S}, D_i)$ by Eqs. 9-10
9: $\quad\quad\quad$**end for**
10: $\quad\quad$**end for**
11: $\quad$**end for**
12: $\quad$Obtain the shortest route $Route(\mathcal{D})$ by Eq. 11
13: $\quad$Calculate the sharing probability $P(Route(\mathcal{D}))$ by Eq. 4
14: $\quad$**if** $P < P(Route(\mathcal{D}))$ **then**
15: $\quad\quad\phi = Route(\mathcal{D})$, $P = P(Route(\mathcal{D}))$
16: $\quad$**end if**
17: **end for**

### D. HEURISTIC SHARING SCHEDULING ALGORITHM

To speed up the exact scheduling algorithm proposed above, we propose a heuristic sharing-scheduling algorithm, in which, the optimal candidate subset is generated during the process of sharing route planning. Here, we first highlight the basic idea of heuristic scheduling algorithm, and then present the details in Algorithm 3.

#### 1) BASIC IDEA

As depicted in Section V-B, for a set of passengers sharing a taxi, the shortest route from their rendezvous point $R_{\mathcal{D}}$ to each of their destinations can be planned by iteratively searching the shortest route of every subset $\mathcal{S} \subseteq \mathcal{D}$ with Eqs. 9 and 10 until $\mathcal{D}$ is traversed completely, where $\mathcal{D}$ is the set of their destinations. It is important to note that, during each iteration, the intermediate shortest route that starts from $R_{\mathcal{D}}$ and visits each destination in subset $\mathcal{S}$, denoted by $Route^*(\mathcal{S})$, can be easily obtained, namely,

$$\begin{aligned} Route^*(\mathcal{S}) &= Route(\mathcal{S}, D_i^*) \text{ and} \\ D_i^* &= \text{argmin}_{D_i \in \mathcal{S}} Len(\mathcal{S}, D_i). \end{aligned} \tag{12}$$

This intermediate shortest route slightly differs from the actual shortest sharing route for the passengers whose destinations in $\mathcal{S}$, due to the two different rendezvous points. The intermediate shortest route starts from the optimal rendezvous point $R_{\mathcal{D}}$ obtained by considering all passengers whose des-

**Algorithm 3** Heuristic Sharing Scheduling

**Input:** (1) A new request $q$ with the destination $D_q$, and (2) A set of candidates with their destinations $\mathcal{T}$.
**Output:** Optimal sharing schedule $\phi$.

1: Select a dummy rendezvous point $R$ at the location that achieves the minimum total distance from all the origins of $q$ and the candidates.
2: Set sharing probability $P = 0$, and set $Len(\{R\}, R) = 0$
3: **for** $s = 1$ to 3 **do**
4:  **for** all subsets $\mathcal{S}$ of size $s$, $\mathcal{S} \subseteq \mathcal{T}$ **do**
5:  $\mathcal{S} \leftarrow R$, $Len(\mathcal{S}, R) = \infty$
6:  **for** $D_i \in \mathcal{S}$ **do**
7:  Obtain $Len(\mathcal{S}, D_i)$ and $Route(\mathcal{S}, D_i)$ by Eqs. 9-10
8:  **end for**
9:  $\mathcal{S} \leftarrow D_q$
10:  **for** $D_i \in \mathcal{S}$ **do**
11:  Obtain $Len(\mathcal{S}, D_i)$ and $Route(\mathcal{S}, D_i)$ by Eqs. 9-10
12:  **end for**
13:  Get the shortest route $Route^*(\mathcal{S})$ by Eq. 12
14:  Set the optimal rendezvous point $R^*$ for $\mathcal{S}$ by Eq. 8
15:  Replace the dummy point $R$ with $R^*$ in $Route^*(\mathcal{S})$
16:  Calculate sharing probability $P(Route^*(\mathcal{S}))$ by Eq. 4
17:  **if** $P < P(Route^*(\mathcal{S}))$ **then**
18:  $\phi = Route^*(\mathcal{S})$, and $P = P(Route^*(\mathcal{S}))$
19:  **end if**
20:  **end for**
21: **end for**

tination in $\mathcal{D}$, while the other is the rendezvous point $R_{\mathcal{S}}$ for the subset of passengers. In practice, the rendezvous points of all subsets of passengers may not be the same, however, these points are usually close with each other and are basically within a short walking distance. This means that the intermediate shortest route $Route^*(\mathcal{S})$ does not differ a lot from its actual optimal sharing route that starts from its own optimal rendezvous point. In other words, a slightly different starting location hardly affects the delivery sequence for a set of passengers sharing a taxi. Therefore, we can utilize the intermediate shortest route to approximate the actual optimal sharing route for each subset of passengers. By doing this, the approximated shortest sharing route of each subset of passengers can be generated during the process of sharing route planning, from which, we can select the optimal subset of candidates sharing a taxi with $q$.

#### 2) ALGORITHM

The heuristic sharing scheduling algorithm is depicted in Algorithm 3. Given a sharing request $q$ and a set of companion candidates, a dummy rendezvous point is first set as the origin of all possible routes, such that the point achieves the minimum total distance from all the origins of candidates and $q$ (Line 1). Next, the algorithm traverses every subset of candidates with the size of at most three, (the number of passengers sharing a taxi should not exceed four.) For each subset, we calculate the shortest route with different last visited destinations $Route(\mathcal{S}, D_i)$ by Eqs. 9 and 10 (Line 7), and then $D_q$ is added to the subset for continuously searching the shortest routes (Lines 9-12). The shortest route for a subset including $D_q$ is obtained by Eq. 12 (Line 13). The optimal rendezvous point is then searched by Eq. 8, and replaced with the dummy rendezvous point (Lines 14-15). We calculate the sharing probability for this subset by Eq. 4. Finally, the schedule with the highest sharing probability is considered as an output (Lines 16-19).

In this heuristic solution, for a request with $N$ candidates, there is still a total of $\sum_{s=1}^{3}\binom{N}{s}$ possible combinations for sharing a taxi with $q$. For each combination, we only need to search $2s + 1$ times for intermediate shortest route $Route(\mathcal{S}, D_i)$. Therefore, the total searching space is $\sum_{s=1}^{3}(2s + 1)\binom{N}{s}$, which is much smaller than that of the exact sharing scheduling algorithm (i.e., $\sum_{s=1}^{3} 2^{s+1}(s + 1)^2\binom{N}{s}$).

## VI. EXPERIMENT SETTINGS

To evaluate the performance of R-Sharing, we conduct extensive experiments using a large-scale real taxi trajectory data set. After describing the experiment settings in this section, we present experimental results in Section VII.

### A. DATA SETS

#### 1) TAXI TRAJECTORIES

The taxi trajectory data set was collected from Nanjing, China from June 1st to 29th, 2010. There are 7,476 taxis with a total of 540,577,652 GPS records. The average frequency of a GPS record reported from a taxi is about 30 seconds. Each record consists of five useful fields for our study, including the taxi ID, timestamp, latitude, longitude, and working status of the taxi. Due to the different travel patterns in weekdays and weekends & holidays, we split the data set into two subsets, namely, trajectories in the 20 weekdays and trajectories in the 9 weekends and holidays, and conduct separated experiments on these two subsets.

#### 2) ROAD NETWORK

We perform the experiments using the real road network of Nanjing, which contains 246,731 road nodes and 126,714 road segments.

### B. SIMULATION PLATFORM

In order to evaluate the performance of R-Sharing, we build a realistic simulation platform by using the trajectory data set as it provides the real taxi supply and demand in a city [26]. Two realistic sharing request streams and two probability matrices of getting vacant taxis are generated over time of a day for weekdays and weekends & holidays separately. With the generated data, each run of the simulation starts from 0:00 am to 23:59 pm, involving a total number of 7,476 taxis.

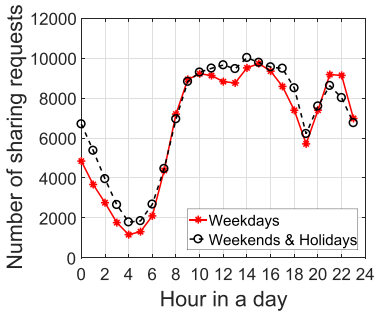



**FIGURE 2.** Generated sharing request number in each hour.

### 1) SHARING REQUEST STREAMS

To generate realistic sharing requests, we first divide the time of the day into non-overlapping equal length time frames and denote the $i$-th time frame by $T_i$. The arrival of requests at each road node is assumed to follow a Poisson distribution during each time frame [27]. To estimate the arrival rate of a new request at road node $n$ during $T_i$, denoted by $\lambda_n^{T_i}$, we assign all ride requests in the data set into the nearest road nodes. For each node, we calculate the average number of ride request per day in each time frame. Let $C_n^{T_i}$ be the average number of ride requests at node $n$ during $T_i$, $\lambda_n^{T_i}$ can be estimated by

$$\lambda_n^{T_i} = C_n^{T_i}/T,$$

where $T$ is the length of each time frame and is set to 30 minutes in the experiments. The generation of taxi-sharing requests at node $n$ during $T_i$ follows a Poisson process with rate of $\lambda_n^{T_i}$. The destination of such a request is selected by the transit probability from origin node $n$ to destination node $m$, denoted by $P_{n,m}^{T_i}$, which is calculated as the average number of requests traveling from $n$ to $m$, denoted by $C_{n,m}^{T_i}$, divided by $C_n^{T_i}$, i.e.,

$$P_{n,m}^{T_i} = C_{n,m}^{T_i} \Big/ C_n^{T_i}.$$

Fig. 2 shows the number of taxi-sharing requests generated during each hour for the weekdays and weekends & holidays, respectively.

Each sharing request is also associated a set of sharing experience thresholds $\langle \delta_{walk}, \delta_{wait}, \delta_{fare}, \delta_{extraT} \rangle$ and weights $\langle w_{walk}, w_{wait}, w_{fare}, w_{extraT} \rangle$ defined by passengers. In the experiments, unless stated otherwise, the default values of the four thresholds are adopted, which are listed in Table 2. Meanwhile, we assume each passenger would assign the same weight on each factor, which is set to 0.25. The default values of other parameters are also listed in Table 2.

### 2) VACANT TAXI PROBABILITY MATRIX

The number of vacant taxis passing by a road node $n$ during time frame $T_i$ is assumed to follow a Poisson distribution $P(X = k) = e^{\lambda} \cdot \lambda^k / k!$, where $k$ is the number of vacant taxis, and $\lambda$ is estimated by the average number of vacant taxis during $T_i$ in different days [21], [27]. The probability of getting a vacant taxi at $n$, denoted by $P_{taxi}(n, t)$, is estimated by the probability of $k \geq 1$, i.e., $P_{taxi}(n, t) = 1 - P(X = 0)$, $(T_i \leq t < T_{i+1})$.

**TABLE 2.** Default parameter setting.

| Parameter | Symbol | Value |
|---|---|---|
| Threshold of walking distance | $\delta_{walk}$ | 300 meters |
| Threshold of waiting time for taxi | $\delta_{wait}$ | 15 minutes |
| Threshold of fare ratio | $\delta_{fare}$ | 0.7 |
| Threshold of extra travel time ratio | $\delta_{extraT}$ | 0.3 |
| Time window for finding companions | $\tau_q$ | 15 minutes |
| Length of a time frame | $T$ | 30 minutes |
| Tunable parameter for sharing probability | $\beta$ | 8 |
| Regular taxi fare per kilometer | $\gamma$ | 3 RMB/km |

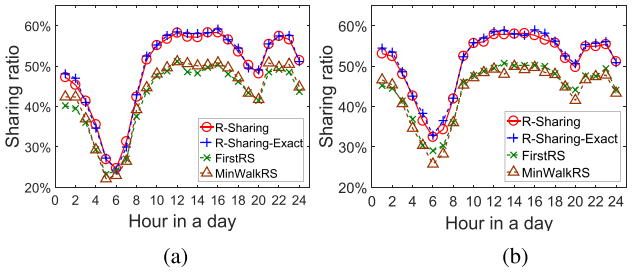



**FIGURE 3.** Percentage of satisfied sharing requests in each hour. (a) Weekday. (b) Weekends & Holidays.

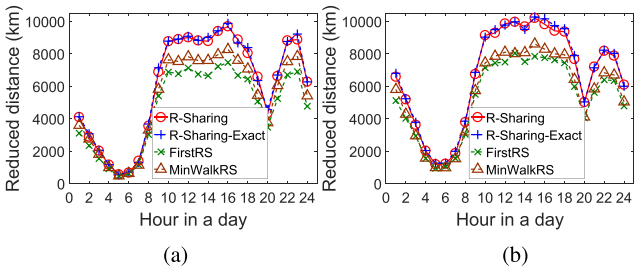



**FIGURE 4.** Reduced total distance from satisfied sharing requests in each hour. (a) Weekday. (b) Weekends & Holidays.

### 3) ESTIMATED WAITING TIME FOR VACANT TAXI

This waiting time is estimated based on the vacant taxi probability matrix. Specifically, we generate the arrival time of vacant taxi at a node $n$ during a time frame $T_i$ based on a Poisson process. The arrival rate of vacant taxis is estimated by using historical taxi trajectories, i.e., the average number of vacant taxis passing by the node $n$ during $T_i$ divided by the length of the time frame. The waiting time can be estimated by the arrival time of vacant taxi and the time of the passenger arriving at a rendezvous point.

## C. IMPLEMENTED METHODS

In all experiments, we compare the performance of the following four methods.

1) R-Sharing. This is our proposed framework. Given a sharing request, it selects the potential companion

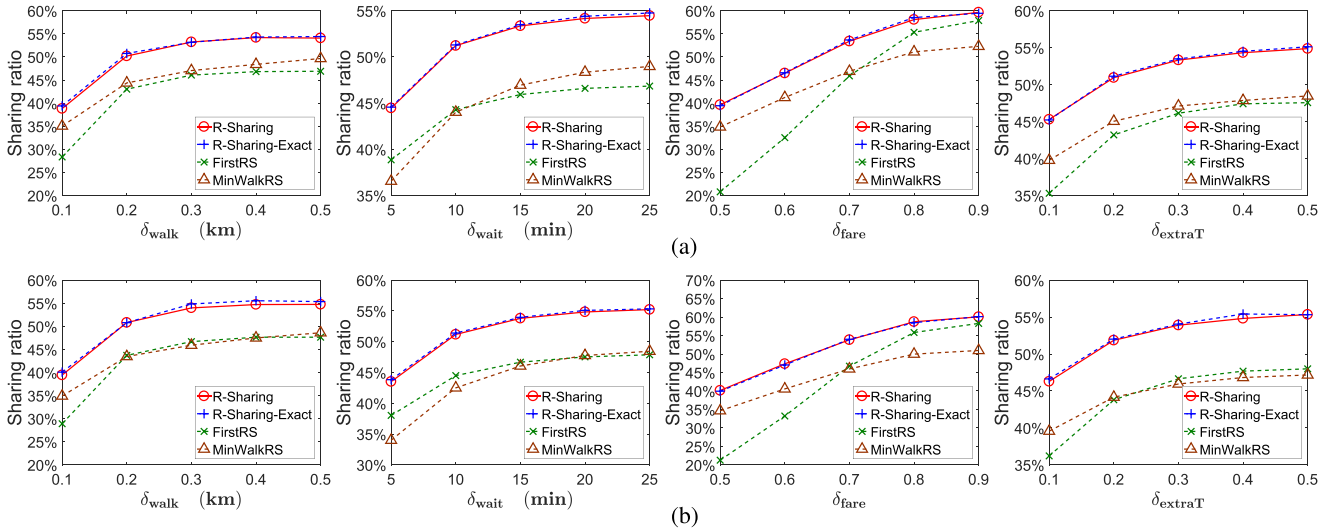



**FIGURE 5.** Sharing ratio with different $\delta$. (a) Weekday. (b) Weekends & Holidays.

candidates using the proposed Algorithm 1 and generates the sharing schedule using the proposed Algorithm 3.

2) R-Sharing-Exact. This is our exact solution for the rendezvous sharing problem, in which, given a sharing request, the potential companion candidates are searched by Algorithm 1, and the optimal sharing schedule is generated by the proposed Algorithm 2.
3) First candidate rendezvous sharing (FirstRS). This method is to minimize the searching and scheduling time. It terminates the searching for candidates once it finds a candidate that satisfies the three candidate constraints. The optimal rendezvous point and the shortest route for this candidate and the new request are scheduled as in R-Sharing.
4) Minimum walking distance rendezvous sharing (MinWalkRS). To study the effectiveness of rendezvous point setting, this method sets the rendezvous point for a set of passengers at the location that achieves the minimum total walking distance. The companion candidate searching and sharing route planning remain the same as in R-Sharing.

Note that it is not suitable to transform existing taxi-sharing techniques into our framework for comparison due to two main reasons. (1) R-Sharing is the first framework that aims to find nearby sharing companions from the perspective of passengers. (2) Existing taxi sharing frameworks dispatch taxis to pick up passengers, and they are designed to find the best match between taxis and passengers.

### D. METRICS

The performance of the implemented methods is evaluated in terms of effectiveness, efficiency, and sharing experience of passengers.

The **effectiveness** is measured by two metrics:

- *Sharing ratio* is the ratio of the number of requests that successfully find companions sharing a taxi to the total number of requests in an hour.
- *Reduced travel distance* measures the total travel distance reduced by taxi-sharing in an hour, i.e., the reduced travel distance is equal to the total travel distance shared by passengers subtracted from their total individual travel distance. This measurement shows how environmental resources, like gas and electricity, can be conserved by the method.

The **efficiency** of taxi-sharing methods are measured by two measurements:

- *Average searching and scheduling time* is the average time for searching companion candidates and scheduling a rendezvous point and a route for a request. It measures the efficiency of the searching and scheduling algorithms.
- *Average waiting time for getting companions* is the average time for a request to wait for getting companions, i.e., from the time when a request is submitted to the time when it gets the response of suggested companions. This metric measures how much time the taxi-sharing method is need to respond to a request.

We measure the **sharing experience** of passengers based on the following four factors:

- *Walking distance* is the road network distance from the origin location of a request to a rendezvous point.
- *Waiting time for a vacant taxi* is the time for a group of passengers waiting for a vacant taxi at a rendezvous point.
- *Fare ratio* is the ratio of the taxi fare of a passenger sharing a taxi with others to the fare without taxi-sharing.

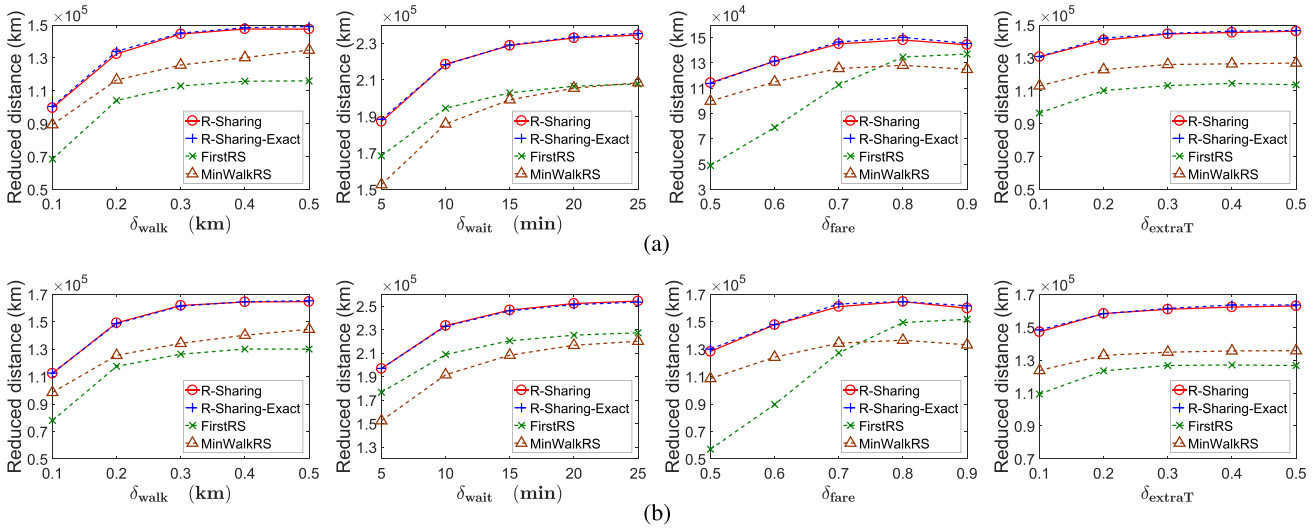



**FIGURE 6.** Reduced travel distance from taxi sharing with different $\delta$. (a) Weekday. (b) Weekends & Holidays.

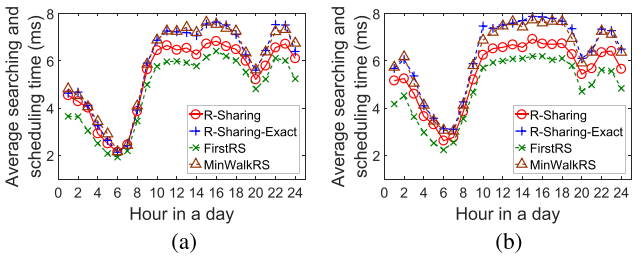



**FIGURE 7.** Average searching and scheduling time in each hour. (a) Weekday. (b) Weekends & Holidays.

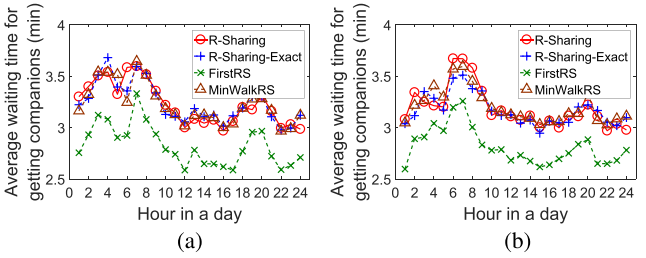



**FIGURE 8.** Average waiting time for getting companions in each hour. (a) Weekday. (b) Weekends & Holidays.

- *Extra travel time ratio* is the ratio of the extra travel time caused by taxi-sharing to the total travel time without taxi-sharing.

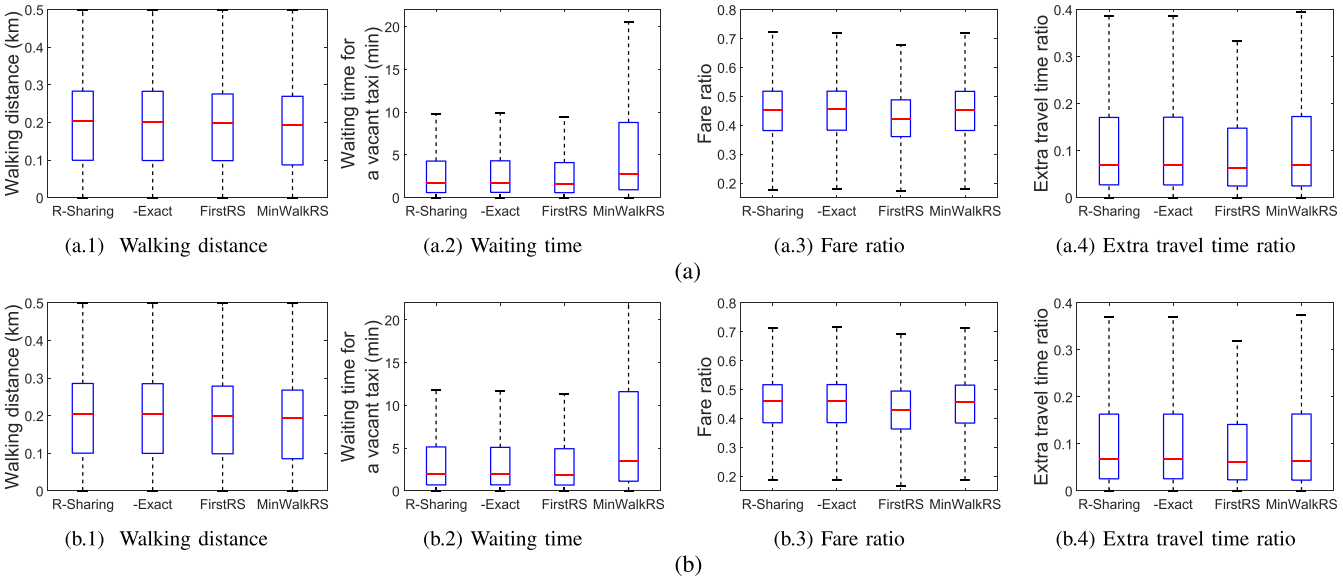
## VII. EXPERIMENTAL RESULTS

This section presents experimental results from perspectives of effectiveness, efficiency, and the sharing experience of passengers.

### A. EFFECTIVENESS

We compare R-Sharing with the R-Sharing-Exact, FirstRS and MinWalkRS methods to study the effectiveness of candidate searching, rendezvous point setting and sharing scheduling in our R-Sharing framework, in terms of sharing ratio and reduced travel distance. In Figs. 3 to 6, where R-Sharing is comparable to R-Sharing-Exact, and performs better than FirstRS and MinWalkRS, with respect to different hours in a day and different $\delta_{walk}$, $\delta_{wait}$, $\delta_{fare}$ and $\delta_{extraT}$ for both weekday and weekend & holiday data sets. The detailed observations are listed as follows.

- Both R-Sharing and R-Sharing-Exact always achieve the best sharing ratio and reduced travel distance. Although R-Sharing adopts the heuristic algorithm which plans a route with an approximate rendezvous point to reduce the computations, its schedules are very comparable to the optimal ones.
- FirstRS performs worse than R-Sharing and R-Sharing-Exact. This is because FirstRS stops searching other companion candidates once it finds one, ignoring other possible candidates that can achieve higher sharing probability. In contrast, both R-Sharing and R-Sharing-Exact search the optimal set of companions who can form the optimal sharing schedule. This strategy could find more passengers sharing a taxi while achieving the highest sharing probability. The significant sharing ratio and reduced distance gaps between FirstRS and R-Sharing demonstrate the necessity and importance of finding the optimal set of companions.
- MinWalkRS also performs worse than R-Sharing and R-Sharing-Exact. The main reason is that MinWalkRS always sets the rendezvous point that achieves the minimum total walking distance without considering real-world road network factors such as the probability of getting a vacant taxi. The waiting time for a taxi at its rendezvous point could be very long, resulting in lower sharing probability at this location. In other words, it is




(a.1) Walking distance (a.2) Waiting time (a.3) Fare ratio (a.4) Extra travel time ratio

(a)

(b.1) Walking distance (b.2) Waiting time (b.3) Fare ratio (b.4) Extra travel time ratio

(b)

**FIGURE 9.** Sharing experience comparison. (a) Weekday. (b) Weekends & Holidays.

harder to find companions due to such long waiting time for a vacant taxi at this rendezvous point.

- As shown in Figs. 3 and 4, all the methods achieve the highest sharing ratio and highest reduced travel distance during 10:00 am to 16:00 pm and 21:00 pm to 23:00 pm. This is because these two time intervals are the peak hours in a day, the more passengers sending sharing requests, the higher probability for them to find sharing companions. It is interesting to find that during early morning (2:00 am to 6:00 am), the sharing ratio and reduced travel distance in weekends and holidays are higher than that of weekdays. This can be explained by the higher travel demand (i.e., the number of requests) during early morning in weekends and holidays than that of weekdays.
- In Figs. 5 and 6, the sharing ratio and reduced travel distance increase as the thresholds $\delta_{walk}$, $\delta_{wait}$, $\delta_{fare}$ and $\delta_{extraT}$ increase. Based on Eq. 3, the sharing probability increases as each of the threshold increases, leading to more passengers sharing a taxi with others. We can observe R-Sharing and R-Sharing-Exact always achieve the highest sharing ratio and reduced travel distance for different thresholds.

### B. EFFICIENCY

We evaluate the efficiency of the four taxi-sharing methods on a computer with 3.4 GHz CPU and 16 GB RAM. Fig. 7 shows the average searching and scheduling time per request for each method in each hour of a day. The average searching and scheduling time is around 5 to 8 ms during peak hours, and around 2 to 4 ms in early morning. Since the number of requests submitted during the peak hours is very large, there is a large candidate set for each request; thus, resulting in larger amount of computations for searching and scheduling. During early morning, as the number of requests decreases, the computation workload for searching and scheduling also decreases. FirstRS always achieves the lowest searching and scheduling time because this method stops searching candidates once it finds one and schedules rendezvous point and route for the only two passengers. It is important to note that R-Sharing incurs less computations than R-Sharing-Exact and MinWalkRS, especially during peak hours.

Fig. 8 shows the average waiting time for getting companions sharing a taxi. The average waiting time is around 3 to 4 minutes for R-Sharing. The waiting time is lower during peak hours than that during early morning. This is because it is easier to find a companion during peak hours, but may need to wait longer for a companion in early morning. FirstRS also achieves the lowest waiting time, as its companions searching process returns the first closest candidate in term of request time, however, such high efficiency is achieved by sacrificing effectiveness. To sum up, R-Sharing strikes a balance between effectiveness and efficiency compared with the other methods.

### C. SHARING EXPERIENCE VALIDATION

In order to evaluate the sharing experience of passengers, we analyze the walking distance to rendezvous points, waiting time for a vacant taxi, fare ratio and extra travel time ratio for the passengers who successfully find companions sharing a taxi.

Fig. 9 shows the distributions of each experience metric for the four methods (''-Exact'' stands ''R-Sharing-Exact'') using boxplot [28]. The band inside each box is the median value of a measurement (such as median walking distance and median waiting time), and each box shows the interquar-

tile range (IQR) of the gaps (i.e., the bottom and top of the box are the 25th and 75th percentiles). Over 75% passengers have the sharing experiences in which the walking distances to rendezvous point are within 0.3 km, the waiting time for a vacant taxi at the rendezvous point are within 5 minutes, the fare ratio is lower than 0.55 (i.e., the fare incurred by taxi-sharing is only 55% of the fare without taxi-sharing), and the extra travel time ratio is within 0.2 (i.e., the travel time by taxi-sharing is only 1.2 times of that without taxi-sharing.)

We can also observe that the sharing experiences of passengers who accept the sharing schedules generated by the four implemented methods have no significant difference. This is because we use the same metric to measure the probability of passengers accepting a sharing schedule in these methods, and passengers accept a sharing schedule only if the schedule can achieve the same level of service quality, i.e., short walking distance, waiting time, extra travel time and low fare. However, it is important to note that although MinWalkRS sets the rendezvous point at the location with the minimum total walking distance, as depicted in Figs. 9(a.1) and 9(b.1), the walking distances generated by MinWalkRS have no significant difference with that by R-Sharing. On the contrary, the waiting time incurred by MinWalkRS are significantly longer than that by R-Sharing (Figs. 9(a.2) and 9(b.2)), because MinWalkRS does not consider the probability of getting a vacant taxi. These results demonstrate that our rendezvous point setting strategy in R-Sharing strikes a balance between the walking distance and the waiting time.

## VIII. CONCLUSION

In this paper, we proposed a new taxi-sharing framework, called R-Sharing, to provide a personalized rendezvous-sharing service. It enables passengers to set their preferences on four essential sharing experience, i.e., walking distance, waiting time, travel fare, and extra travel time. Given a taxi-sharing request, R-Sharing searches the optimal set of nearby companions, recommends a rendezvous point for them to meet, and plans their shortest sharing routes based on their personal preferences, such that the probability for them to accept a sharing schedule is maximized. A probabilistic model was proposed to estimate the sharing probability by considering potential passengers' personal preferences on sharing experience. We then proposed a companion candidate searching algorithm to search nearby potential companion candidates. To select the optimal subset of candidates sharing a taxi with the request, an exact taxi-sharing scheduling algorithm was developed to generate the optimal sharing schedule (comprising of the optimal set of companions, the optimal rendezvous point and the shortest sharing route) for a sharing request, such that the sharing probability is maximized. To improve the efficiency, a heuristic sharing scheduling algorithm was also proposed for generating the optimal sharing scheduling Extensive experiments are conducted using a one-month taxi trajectory data set collected in Nanjing, China. Experimental results confirmed that R-Sharing is effective and efficient, and provides excellent sharing experiences for its passengers.

## REFERENCES

[1] A. Di Febbraro, E. Gattorna, and N. Sacco, "Optimization of dynamic ridesharing systems," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2359, pp. 44–50, Oct. 2013.

[2] K. Ghoseiri, A. E. Haghani, M. Hamedi, and M. Center, "Real-time rideshare matching problem," Dept. Civil Environ. Eng., Univ. Maryland, College Park, MD, USA, Tech. Rep. UMD-2009-05 and DTRT07-G-0003, 2011.

[3] E. Kamar and E. Horvitz, "Collaboration and shared plans in the open world: Studies of ridesharing," in *Proc. IJCAI*, 2009, pp. 1–8.

[4] Z. Siddiqi and R. Buliung, "Dynamic ridesharing and information and communications technology: Past, present and future prospects," *Transp. Planning Technol.*, vol. 36, no. 6, pp. 479–498, 2013.

[5] M. Furuhata, M. Dessouky, F. Ordóñez, M.-E. Brunet, X. Wang, and S. Koenig, "Ridesharing: The state-of-the-art and future directions," *Transp. Res. B, Methodol.*, vol. 57, pp. 28–46, Nov. 2013.

[6] D. O. Santos and E. C. Xavier, "Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive," *Expert Syst. Appl.*, vol. 42, no. 19, pp. 6728–6737, 2015.

[7] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *Proc. IEEE ICDE*, Apr. 2013, pp. 410–421.

[8] S. Ma, Y. Zheng, and O. Wolfson, "Real-time city-scale taxi ridesharing," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 7, pp. 1782–1795, Jul. 2015.

[9] G. Berbeglia, J.-F. Cordeau, and G. Laporte, "Dynamic pickup and delivery problems," *Eur. J. Oper. Res.*, vol. 202, no. 1, pp. 8–15, 2010.

[10] M. Diana and M. M. Dessouky, "A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows," *Transp. Res. B, Methodol.*, vol. 38, no. 6, pp. 539–557, 2004.

[11] R. M. Jorgensen, J. Larsen, and K. B. Bergvinsdottir, "Solving the dial-a-ride problem using genetic algorithms," *J. Oper. Res. Soc.*, vol. 58, no. 10, pp. 1321–1331, 2007.

[12] B. Hunsaker and M. Savelsbergh, "Efficient feasibility testing for dial-a-ride problems," *Oper. Res. Lett.*, vol. 30, no. 3, pp. 169–173, 2002.

[13] A. Kleiner, B. Nebel, and V. A. Ziparo, "A mechanism for dynamic ride sharing based on parallel auctions," in *Proc. IJCAI*, vol. 11. 2011, pp. 266–272.

[14] C. Tian, Y. Huang, Z. Liu, F. Bastani, and R. Jin, "Noah: A dynamic ridesharing system," in *Proc. ACM SIGMOD*, 2013, pp. 985–988.

[15] D. Mukherjee, S. Banerjee, and P. Misra, "Ad-hoc ride sharing application using continuous sparql queries," in *Proc. WWW*, 2012, pp. 579–580.

[16] D. Zhang, T. He, Y. Liu, and J. A. Stankovic, "CallCab: A unified recommendation system for carpooling and regular taxicab services," in *Proc. IEEE Big Data*, Oct. 2013, pp. 439–447.

[17] M. Ota, H. Vo, C. Silva, and J. Freire, "A scalable approach for data-driven taxi ride-sharing simulation," in *Proc. IEEE Big Data*, Oct. 2015, pp. 888–897.

[18] Y. Huang, F. Bastani, R. Jin, and X. S. Wang, "Large scale real-time ridesharing with service guarantee on road networks," *Proc. VLDB Endowment*, vol. 7, no. 14, pp. 2017–2028, 2014.

[19] 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 Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2587–2598, Oct. 2015.

[20] M. Taylor and C. D. Creelman, "PEST: Efficient estimates on probability functions," *J. Acoust. Soc. Amer.*, vol. 41, no. 4A, pp. 782–787, 1967.

[21] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, Oct. 2013.

[22] A. Mingozzi, L. Bianco, and S. Ricciardelli, "Dynamic programming strategies for the traveling salesman problem with time window and precedence constraints," *Oper. Res.*, vol. 45, no. 3, pp. 365–377, Jun. 1997.

[23] C. Malandraki and R. B. Dial, "A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem," *Eur. J. Oper. Res.*, vol. 90, no. 1, pp. 45–55, Apr. 1996.

[24] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter, "Exact routing in large road networks using contraction hierarchies," *Transp. Sci.*, vol. 46, no. 3, pp. 388–404, 2012.

[25] D. Zhang, Y. Liu, A. Liu, X. Mao, and Q. Li, "Efficient path query processing through cloud-based mapping services," *IEEE Access*, vol. 5, pp. 12963–12973, 2017.

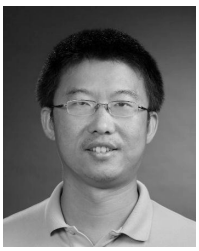
[26] Z. Feng and Y. Zhu, "A survey on trajectory data mining: Techniques and applications," *IEEE Access*, vol. 4, pp. 2056–2067, 2016.
[27] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.
[28] M. Frigge, D. C. Hoaglin, and B. Iglewicz, "Some implementations of the boxplot," *Amer. Stat.*, vol. 43, no. 1, pp. 50–54, 1989.

**YAN LYU** received the M.S. degree in pattern recognition and intelligent systems from the University of Science and Technology of China, China, in 2013, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2016. She was a Post-Doctoral Research Fellow with Hong Kong Baptist University, Hong Kong, in 2017. She is currently a Post-Doctoral Research Fellow with the National University of Singapore, Singapore. Her research interests include intelligent transportation systems, spatiotemporal data analytics, mobile computing, and location-based services.

**VICTOR C. S. LEE** (M'92) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 1997. He is currently an Assistant Professor with the Department of Computer Science, City University of Hong Kong. His research interests include intelligent transportation systems, data dissemination in vehicular networks, real-time databases, and performance evaluation. He is a member of the Association for Computing Machinery and the IEEE Computer Society. He has been the Chairman of the IEEE Hong Kong Section Computer Chapter from 2006 to 2007.

**CHI-YIN CHOW** received the M.S. and Ph.D. degrees from the University of Minnesota-Twin Cities in 2008 and 2010, respectively. He is currently an Assistant Professor with the Department of Computer Science, City University of Hong Kong. His research interests include machine learning, spatio-temporal data management and analysis, GIS, mobile computing, and location-based services. He was the Co-Founder and Co-Organizer of ACM SIGSPATIAL MobiGIS 2012–2016. He received the 10-Year Best Paper Award at VLDB 2016, and the Best Paper Awards at ICA3PP 2015 and MDM 2009.

**JOSEPH KEE-YIN NG** received the B.Sc., M.Sc., and Ph.D. degrees in computer science all from the University of Illinois at Urbana–Champaign. He joined Hong Kong Baptist University (HKBU) in 1993, and is currently a Full Professor and the Director with the Research Centre for Ubiquitous Computing, Department of Computer Science. He is also the Programme Director of the computer science degree programme and introduced health information technology and health informatics into the undergraduate and the graduate programmes in HKBU. His current research interests include real-time and embedded systems, cyber physical systems, Internet of Things, wireless communications, and ubiquitous/pervasive computing, location-aware computing, mobile and ubiquitous healthcare. He received two Patents and published over 160 technical papers in journals and conferences. He has also served as the Steering Chairs, the Program Chairs, and the General Chairs for numerous International Conferences and the Associate Editors and members of the Editorial Board of International Journals. Prof. Ng had also served as the Region ten Coordinator for the Chapter Activities Board of the IEEE Computer Society, and was the Coordinator of the IEEE Computer Society Distinguished Visitors Program (Asia/Pacific). He is a Senior Member of the IEEE and has been a member of the IEEE Computer Society since 1991. He has been an Exco-member, General Secretary, Vice-Chair, Chair and is currently Past Chair and Exco-member for the IEEE, Hong Kong Section, Computer Chapter. He received numerous Awards and Certificate of Appreciation from the IEEE, the IEEE Region 10, the IEEE Computer Society, and from the IEEE Hong Kong Section, for his Leadership and Services to the ICT Industry. He is also a member of the IEEE Communication Society, ACM, a fellow of the Hong Kong Computer Society, and the Founding Member and Exco-Member, Treasurer, and Vice-Chair for the Internet Society-Hong Kong Chapter. He had also been a member of the Board of Directors of the Hong Kong Internet Registration Corporation Limited and within his team has received The Directors of the Year 2012 Award organized by The Hong Kong Institute of Directors.



**YANHUA LI** (S'09–M'13–SM'16) received double Ph.D. degrees in electrical engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2009, and in computer science from the University of Minnesota at Twin Cities in 2013, respectively. He was a Researcher with the Huawei Noah's Ark Lab, Hong Kong from 2013 to 2014, and has interned with Bell Labs, NJ, USA, Microsoft Research Asia, and Huawei Research Laboratory of America from 2011 to 2013. He is currently an Assistant Professor with the Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA, USA. His research interests are urban data analytics, data-driven cyber-physical systems, and smart cities.

**JIA ZENG** (S'05–M'07–SM'13) received the B.Eng. degree from the Wuhan University of Technology, Wuhan, China, in 2002, and the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2007. He is currently a Principal Researcher with Huawei Noah's Ark Laboratory, Hong Kong. His research interests are machine learning and big data applications. He is a member of the CCF and the ACM.