Public Vehicles for Future Urban Transportation

Ming Zhu, Xiao-Yang Liu, Feilong Tang, *Member, IEEE*, Meikang Qiu, *Senior Member, IEEE*, Ruimin Shen, *Member, IEEE*, Wennie Shu, *Senior Member, IEEE*, and Min-You Wu, *Senior Member, IEEE*

***Abstract*—This paper advocates a new paradigm of transportation systems for future smart cities, namely, public vehicles (PVs), that provides dynamic ridesharing trips at requests. Passengers will enjoy more convenient and flexible transportation services with much less expense. In the PV system, both the number of vehicles and required parking spaces will be significantly reduced. There will be less traffic congestion, less energy consumption, and less pollution. In this paper, the concept, method, and algorithm for the PV system are described. The key issue of effectively implementing the PV system is to design efficient planning and scheduling algorithms. The PV-path problem is formulated, which is NP-complete. Then, a practical approach is proposed, which can serve people anywhere and anytime. The simulation results show that, to achieve the same performance (e.g., total time, waiting time, and travel time), the number of vehicles in the PV system can be reduced by around 90% and 57% compared with the conventional vehicle system and Uber Pool, respectively, and the total traveling distance can be reduced by 34% and 14%.**

***Index Terms*—Public vehicles, future urban transportation, path planning, ridesharing, shared mobility systems.**

I. Introduction

On road networks, non-shared rides, such as single-occupant vehicles, single-passenger vehicles, private cars and taxis, serve the majority of transportation requests for personal travel and daily commuting. These lead to low traffic efficiency due to three factors: low utility, low sharing factor, and low occupancy rate. Some survey shows that, the average occupancy rate of each car is only 1.6 [1]. In recent decades, impacts of non-shared rides on social issues, such as energy consumption, air pollution, and traffic congestion, are paid much attention to.

In order to overcome the inefficiency of non-shared rides, shared rides become a hot topic. Buses can provide shared rides, however their routes are fixed, and the travel time is much longer than non-shared rides such as private cars and taxis, although the price is much lower [2]. Therefore, shared rides with service guarantee such as short trip time are certainly a direction to pursue.

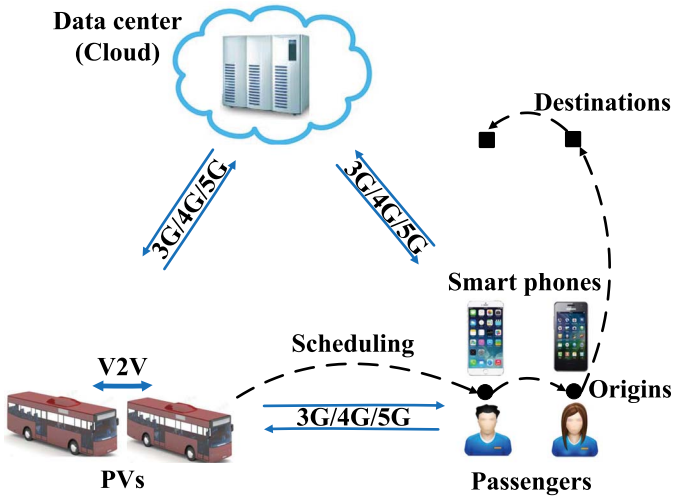


Fig. 1. Architecture of the PV system.

This paper proposes a shared mobility system, public vehicles (PVs), which may be run by a government or a company in the future. PVs are one type of high occupancy vehicles and may be driverless/autonomous [3] electric vehicles, [4] or plug-in hybrid electric vehicles in the future. PVs provide ridesharing trips with service guarantee to replace buses, private cars, and taxis in urban areas. The architecture of the PV system is shown in Fig. 1. It has three parts: data center (cloud), users/passengers, and PVs. If one user/passenger needs service, he/she sends a request through a smart phone to a cloud [5], which includes a pickup point (origin) and a dropoff point (destination). Then the cloud schedules a PV to serve him/her by traversing his/her origin to destination. How to design paths of PVs is referred as PV path (PVP) problem. Similar to existing popular ridesharing systems, users/passengers recognize PVs by license plates or logos.

The PV system is promising for future urban transportation with several advantages. *First*, it is a centralized system controlled by a cloud and can provide dynamic ridesharing service at requests. *Second*, the quality of trip service can be guaranteed, e.g., limited detour. There is no last mile problem compared with buses. *Third*, the price will be much lower than private cars and taxis due to high traffic sharing factor. *Fourth*, to improve the traffic sharing, the capacity of PVs is larger than taxis, and to preserve the flexibility and dynamic of PVs, the capacity is smaller than buses. The discussions about the capacity of PVs are in Section V-B.

The PV system is different from existing carpool. Carpool is usually a personal decision based on plenty of factors, e.g., trip length, travel time, number of participants. To start a carpool, the owner/driver of a vehicle needs to communicate with other potential participants to reach an agreement on the route. Once

Manuscript received September 10, 2015; revised January 17, 2016; accepted March 4, 2016. Date of publication May 18, 2016; date of current version November 23, 2016. This work was supported in part by the National Natural Science Foundation of China under Project 61373155, Project 91438121, Project 61373156, Project 61303202, Project 91438121, and Project 61373156. The work of M. Qiu was supported by the U.S. National Science Foundation under Grant CNS-1457506. The Associate Editor for this paper was H. A. Rakha.

M. Zhu, X.-Y. Liu, F. Tang, R. Shen, and M.-Y. Wu are with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zhumingpassional@sjtu.edu.cn; yanglet@sjtu.edu.cn; tang-fl@cs.sjtu.edu.cn; rmshen@sjtu.edu.cn; mwu@sjtu.edu.cn).

M. Qiu is with the Department of Computer Science, Pace University, New York City, NY 10038 USA (e-mail: mqiu@pace.edu).

W. Shu is with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87106 USA (e-mail: shu@ece.unm.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2016.2543263

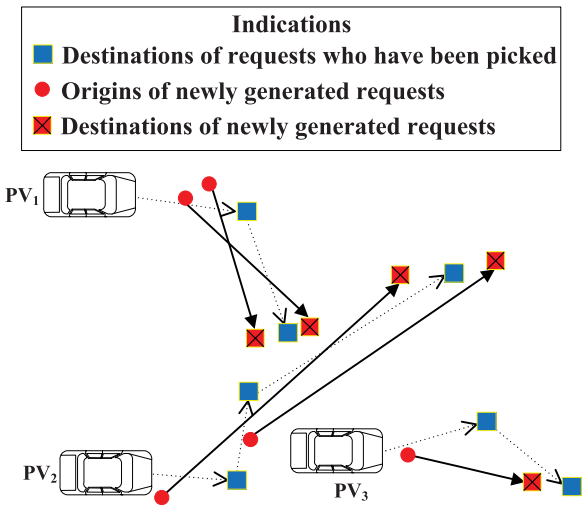


Fig. 2. Scenario of the PV system on a road network.

the carpool starts, no more new participants can join the route even their requests match the current routes perfectly. Fig. 2 shows a scenario: PVs can continuously serve multiple requests along their routes. However, in existing carpool, the newly generated requests may be ignored.

The PV system is also different from existing ridesharing: *First*, the PV system is a centralized system, while the multi-rider ridesharing is a distributed system. *Second*, in the PV system, scheduling strategies are calculated by the cloud, while in multi-rider ridesharing, scheduling strategies are negotiated by drivers and riders, which may be far from optimal solutions. *Third*, PVs cooperate with each other to improve traffic efficiency, e.g., providing multi-hop paths, while in multi-rider ridesharing drivers compete with each other to earn more money. *Fourth*, the PV system is much safer due to larger capacity compared with taxis since vehicles with large capacity will be as safe as buses.

This paper focuses on a path planning algorithm for the PV system to minimize vehicles' total travel distance with service guarantee such as low detour. The PVP problem is challenging due to the following reasons. On one hand, PVs provide flexible ridesharing service for both planned trips and unplanned trips, and the paths of PVs dynamically change over time due to new passengers. On the other hand, the comfort of passengers should be guaranteed, e.g., short waiting-travel time.

The contribution of this paper is summarized as follows:

- A type of cloud-based transportation system, public vehicles, is proposed to provide dynamic ridesharing service at requests with service guarantee in urban areas of smart cities.
- To reduce the total travel distance of PVs, preserve comfort (e.g., low detour), and increase the sharing factor of transportation resource, the PVP problem is introduced and formulated. Then its NP-Completeness is derived.
- A practical algorithm is proposed for passengers with service guarantee, e.g., short waiting time and low detour, through which requests can be dealt with promptly. Then a local optimization method is proposed for performance improvement based on the heuristic of the traveling salesman problem.
- Extensive simulations are conducted to evaluate the performance of the PV system compared with two transportation systems: conventional vehicles (consisting of buses, private cars, and taxis) and Uber Pool. The PV system achieves the same performance (total time, waiting plus travel time) with much smaller number of vehicles. Therefore, the traffic congestion is reduced. Moreover, the detour is low.

The rest of this paper is then organized as follows: The backgrounds based on ridesharing and the PV system are described in Section II. In Section III, the system model is described, and the PVP problem is formulated, and then the NP-Completeness is analyzed. Section IV describes the proposed algorithm. Section V shows the performance of the PV system using proposed algorithm and other two transportation systems. Section VI concludes this work.

II. Related Work

Ridesharing [6], [7] is popular recent years since it can reduce the traffic and improve the traffic efficiency. Ridesharing may be widely used in the future with sacrificing a little comfort for passengers such as detour. It is possible that many people approve ridesharing. Some survey results show that, about 45% of people will be interested or potentially will be interested in ridesharing [8]. One study in the city of Madrid [9] presents that the traffic can be reduced by 59% if passengers are willing to share rides with others who work or live within 1 km. From the taxi trips of New York City, Santi *et al.* find that, the trip length can be reduced by 40% [10] with low discomfort and little prolonged travel time. Alexander *et al.* explore the impact of ridesharing services on network-wide traffic congestion using mobile phone records, and find that ridesharing has noticeable impacts on congested travel time [11].

There are still several challenging problems although ridesharing is proposed several decades ago. For example, the method of ride splitting and merging can only be used to cars and not to public transportation systems through the method of examining user-based redistribution in shared mobility systems [12]. Building a ridesharing or carpool model under current traffic patterns is also an important issue. Considering of stochastic disturbances in vehicle travel time, Yan *et al.* develop a stochastic carpool model [13]. However, it is still difficult to build a model [14] to analyze transportation systems with presence of uncertainty.

The most important issue in ridesharing is to determine the paths planning for vehicles [13]. Information communication technology (ICT) such as GPS trajectories mining [15] can help for the routing scheme design. The shared common trip patterns [9] rely on the common subpaths of people, e.g., nearby origins, nearby destinations, and similar traveling routes. Zhang *et al.* propose a carpool solution [16] to reduce the total travel distance of cars. To prompt people to participate carpool a fare model is also introduced. However, the carpool should start only at origins of passengers (e.g., airports or train stations) with their destinations close by. Ma *et al.* propose T-share [17] in taxi sharing system to retrieve candidate taxis that are likely to satisfy a user query (including origin, destination, and time windows). The routing strategy is that, each passenger will be served by a taxi with the minimum detour without considering comfort. Dial-a-ride problem [18] aims at constructing a set of vehicle routes to serve passengers. Meanwhile, some static settings should be satisfied: the given time, capacity and precedence constraints. However, there is no service guarantee for passengers.

All the above solutions have their constraints, which limit the range of use. Some solutions rely on personal decisions, which

TABLE I
DENOTATIONS IN THE PV SYSTEM

R, R_1, R_2	set of requests: $R = R_1 \bigcup R_2$.	$x_{p,i,j}$	is 1, if p traverses edge $(i,j) \in E$, otherwise, 0.
m, m_1, m_2	number of requests. $m = \|R\|$, $m_1 = \|R_1\|$, $m_2 = \|R_2\|$.	$g^+_{p,i}$	is 1 if p picks one request at vertex i, otherwise, 0.
a_p	current location of p.	$g^-_{p,i}$	is 1 if p drops one request at vertex i, otherwise, 0.
P	set of PVs.	c_p	capacity of PVs.
N_p	total number of PVs. $N_p = \|P\|$.	μ_p, e_p	μ_p is set of requests currently being served by p. $e_p = \|\mu_p\|$.
$Y_{p,r}$	is 1, if p serves r, otherwise, 0.	$\tau_{p,i,j}$	travel time from vertex i to j on the path of p.
$f^-_{p,i}$	number of dropped requests by p before i (i is not included).	$d_{i,j}$	distance based on shortest path from vertex i to j.
$f^+_{p,i}$	number of picked requests by p before i (i is not included).	$K_{p,i,j}$	is 1 if both vertices i and j are on the path of p and i precedes (not necessarily immediately) j, otherwise, 0.

usually is not close to an optimal solution. Some strategies do not consider the trip comfort of passengers such as detour distance. The PV system has few constraints compared with existing traditional ridesharing or taxi sharing, and does not depend on personal decisions. An important objective of the PV system is to provide high quality of service for passengers dynamically and efficiently.

III. System Model and Problem Formulation

In this section, the system model of the PVP problem is detailed, and then it is formulated. Finally, the NP-Completeness is discussed.

A. System Model

This paper focuses on the PVP problem with fewer constraints, which includes the ridematching between PVs and passengers, and paths of PVs. Upon a request, a passenger can only be served by a single vehicle from his/her origin to destination, while the multi-hop scheme is not in the scope of this paper.

Assume at some time t_s, there are N_p PVs, denoted by a set P, and total m requests, denoted by a set $R = R_1 \bigcup R_2$. R_1 is a set of requests currently being served by PVs, who have been picked up yet have not arrived at destinations. All the requests except ones in R_1 are put to another set R_2, where some of them may be assigned before, here label them as unassigned. Temporarily, assume each request is corresponding to one passenger. For R_1, since requests are currently being served, their origins are not meaningful anymore but their destinations need to be reached. Thus, let V_d^1 be a set of the destinations of requests existing in R_1. For R_2, both origins and destinations of requests need to be considered. Thus, for requests in R_2, let V_o^2 and V_d^2 be sets of their origins and destinations, respectively. Consider a weighted complete graph, $G = (V, E)$, and $V = V_p \bigcup V_o^2 \bigcup V_d$, and E is a set of edges between two vertices of V. V_p is the set of locations of PVs. Let $V_d = V_d^1 \bigcup V_d^2$, $V' = V_o^2 \bigcup V_d$. The weight of edge between any pair of vertices i and j, is $d_{i,j}$, the traveling distance based on the shortest path between them. Let $r \in R$ denote one request or passenger, and $p \in P$ denote one PV. To describe the problem, variables are summarized in Table I. Two definitions about a schedule for a request and a service list of a PV are presented here.

Definition 3.1: A schedule for any request r, (r_t, r_l, r_o, r_d), denotes the assignment of one PV p to r for trip service. r_t is the request submitting time, r_l is the latest arrival time, r_o is the origin, and r_d is the destination. p will transverse through r_o and r_d with r_o preceding r_d, and meanwhile will pick r at r_o, and drop r at r_d after time r_t before r_l.

Definition 3.2: A service list of any PV p is a set of requests p has to serve, including the requests being served (have been picked up by p, yet have not arrived at their destinations), and the ones will be served (scheduled to p, yet have not been picked up). Let l_p denote the service list, and $|l_p|$ denote the number of requests in l_p.

B. Problem Formulation

$$\text{Objective}: \quad \min \sum_{p \in P} \sum_{\{i,j\} \in E} d_{i,j} * x_{p,i,j} \tag{1}$$

Subject to :

$$\sum_{p \in P} Y_{p,r} = 1,\ r \in R \tag{2}$$

$$f^+_{p,i} + g^+_{p,i} * x_{p,i,j} = f^+_{p,j} \tag{3}$$

$$f^-_{p,i} + g^-_{p,i} * x_{p,i,j} = f^-_{p,j} \tag{4}$$

$$\sum_{j \in V', j \neq i} x_{p,i,j} \leq 1,\ i \in V'' \tag{5}$$

$$\sum_{i \in V'', i \neq j} x_{p,i,j} \leq 1,\ j \in V' \tag{6}$$

$$K_{p,i,j} \geq x_{p,i,j} \tag{7}$$

$$K_{p,i,j} + K_{p,j,i} \leq 1 \tag{8}$$

$$K_{p,i,j} + K_{p,j,j'} + K_{p,j',i} \leq 2 \tag{9}$$

$$e_p + f^+_{p,i} - f^-_{p,i} + g^+_{p,i} - g^-_{p,i} \leq c_p,\ i \in V' \tag{10}$$

$$t_s + \sum_{p \in P} \tau_{p,a_p,r_o} * K_{p,a_p,r_o} \geq r_t,\ r \in R_2 \tag{11}$$

$$t_s + \sum_{p \in P} \tau_{p,a_p,r_d} * K_{p,a_p,r_d} \leq r_l,\ r \in R \tag{12}$$

$$x_{p,i,j}, K_{p,i,j}, K_{p,j,j'}, K_{p,j',i}, g^+_{p,i}, g^-_{p,i} \in \{0,1\}$$

$$f^+_{p,i}, f^-_{p,i} \geq 0.$$

The formulation of PVP using Mixed-Integer Linear Programming (MILP) is shown by Eqns. [(1)–(12)]. Eqn (1), is the objective function, and Eqns. [(2)–(12)] are constraints. Eqn. (1) minimizes the sum of travel distance of each PV. Although trip time is an important issue in the PV system, the optimization of travel distance is chosen as the objective function for several reasons. *First*, if the speed is constant, the optimization of travel distance is equivalent to time. *Second*, the optimization of travel distance can be easily extended to optimize time if taking into account the speed of each road segment. The details are not shown for limited pages. In fact, it is hard to predict speed for traffic congestion. *Third*, the proposed algorithm considers preserving short waiting-travel time and the simulation results show that, the algorithm has good performance for waiting-travel time of passengers.

Eqn. (2) ensures that there are exactly m one-to-one assignments among PVs and requests. Eqns. [(3), (4)] imply the total number of picked or dropped passengers when p traverses the edge $\{i, j\}$. Eqns. [(5)–(9)] are similar to the formulation of the precedence constrained traveling salesman problem. Eqns. (5), (6) mean that, any location from V''/V' at most has one

successor/precursor. Therein, $V'' = V' \bigcup \{a_p\}$. Eqn. (7) implies the relationship between $K_{p,i,j}$ and $x_{p,i,j}$: $K_{p,i,j}$ is not less than $x_{p,i,j}$. This can be inferred from their definitions. Eqn. (8) describes two cases: If neither edge $\{i,j\}$ nor $\{j,i\}$ is not on the path of p, $K_{p,i,j} = K_{p,j,i} = 0$. If edge $\{i,j\}$ or $\{j,i\}$ is on the path of p, only either one of $K_{p,i,j}$ and $K_{p,j,i}$ is 1, and the other is 0. Eqn. (9) prevents the occurrence of subtours. Eqn. (10) is the constraint of capacity of PVs. Eqns. (11), (12) imply the constraint of the request submitting time and latest arrival time.

C. NP-Completeness

The PVP problem is NP-Complete, as it is a dynamic finite capacity dial-a-ride problem [18] with more constraints (e.g., time), which is NP-Hard.

Consequently, using MILP for the optimal solution of the PVP problem becomes impractical, especially when the numbers of PVs and requests increase in a large scale road network. In the next section, the construction of paths for PVs and the schedule of requests heuristically about this problem are discussed.

IV. Heuristics

In this section, one insertion method for path planning is introduced, and then an algorithm is detailed, and finally a local optimization method is introduced.

A. A Key Routine

The origin-destination insertion method is a key routine used in the proposed algorithm. In general, for the requests in μ_p (by definition, they are in R_1), there is no precedence constraint among their destinations. On the other hand, for a new request $r \in R_2$, its origin r_o has to be visited before its destination r_d. Therefore, if p decides to take a request $r \in R_2$, both r_o and r_d need to be inserted into its current path with the precedence constraint being satisfied. With respect to r and p, the insertion cost $\pi_{r,p,i,j}$ is the additional cost of servicing r by p at locations i and j for r_o and r_d. Assume that r is taken by p.

Definition 4.1: The insertion cost of r at locations (i,j) on the current path of p $\pi_{r,p,i,j}$ is the additional cost of picking up r at i and dropping off r at j. If we insert one origin-destination (OD) pair (r_o, r_d) of r, where r_o precedes r_d, a new path will be obtained. The cost $\pi_{r,p,i,j}$ is the difference between the distance of the two paths.

Here, the current path of p $\{\theta_0, \theta_1, \ldots, \theta_{L_p}\}$, includes the location of p $\theta_0 = a_p$. In order to insert a new request r, select one location θ_i $(0 \le i \le L_p)$ on the path of p to insert r_o after θ_i. Then from the locations after r_o select another location $\theta_j (i+1 \le j \le L_p + 1)$ to insert r_d. Thus, there are $(L_p - i + 1)$ locations to select r_d for every selected r_o. The insertion complexity for one PV is $O(L_p^2)$. Let a function $q_p = \text{INSERT}(r, p, i, j)$ return q_p, the path of p, by inserting origin r_o and destination r_d at the locations of i and j on the path of p.

Let $d\{\theta_{i-1}, \theta_i\}$ denote the shortest distance between θ_{i-1} and θ_i on the road network. Let $d\{\theta_i, \ldots, \theta_j\} = d\{\theta_i, \theta_{i+1}\} + \ldots, + d\{\theta_{j-1}, \theta_j\}$ $(i < j)$ denote the shortest distance from θ_i to $\theta_{i+1}, \ldots$ and to θ_j. Eqn (13) describes four cases of inserting one OD pair. The *first* case means that, r_d is not the last point of the path, and r_o immediately precedes r_d. The *second* case means that, r_d is the last point, and r_o immediately precedes r_d. The *third* case means that, r_d is the last point, and r_o does not immediately precede r_d. The *fourth* case means that, r_d is not the last point, and r_o does not immediately precede r_d.

$$\pi_{r,p,i,j} = \begin{cases} d\{\theta_i, r_o, r_d, \theta_{i+1}\} - d\{\theta_i, \theta_{i+1}\}, \\ \quad \text{if } 0 \le i \le L_p - 1,\ j = i+1 \\ d\left\{\theta_{L_p}, r_o, r_d\right\}, \\ \quad \text{if } i = L_p,\ j = i+1 \\ d\{\theta_i, r_o, \theta_{i+1}\} + d\left\{\theta_{L_p}, r_d\right\} - d\{\theta_i, \theta_{i+1}\}, \\ \quad \text{if } 0 \le i \le L_p - 1,\ j = L_p + 1 \\ d\{\theta_i, r_o, \theta_{i+1}\} + d\{\theta_j, r_d, \theta_{j+1}\} \\ -d\{\theta_i, \theta_{i+1}\} - d\{\theta_j, \theta_{j+1}\}, \\ \quad \text{if } 0 \le i \le L_p - 1, j \le L_p, j \ne i+1. \end{cases} \quad (13)$$

Definition 4.2: The least insertion cost (LIC) $\pi_{r,p} = \pi_{r,p,i',j'}$ will be the least cost for all possible insertion locations of i' and j' of p for the request r.

Definition 4.3: The minimum insertion cost (MIC) $\pi_r = \pi_{r,p'} = \pi_{r,p',i',j'}$ will be the minimum insertion cost of the request r for all possible PVs in P.

B. Algorithm

The general idea of the algorithm is that, to reduce the waiting time, put the requests with the longest waiting time as the highest scheduling rank. Then use the origin-destination insertion method to obtain multiple paths of PVs. To guarantee the *QoS* (quality of service) of passengers, the detour ratio should not exceed its threshold. To reduce the travel distance of PVs, choose the path with the minimum distance under the above constraints.

Latest arrival time is not considered here for several reasons. On one hand, it is still hard to accurately predict the travel time, although some researchers have proposed new solutions [19] for it. On the other hand, some passengers may be upset about waiting for others even several minutes raised by the inaccuracy of speed prediction.

Travel distance, equivalent waiting distance and detour ratio are introduced here since they constitute comfort of passengers. Assume p serves r. Let r_p and r_a denote the pickup and arrival (dropoff) time. The travel distance of p from the time r_t to r_p is named as *equivalent waiting distance*, which is denoted by d_r^w (Unit: km). Each passenger has a equivalent waiting distance threshold W_r (Unit: km). The travel distance of p from the time r_p to r_a is named as *travel distance* of r, which is denoted by d_r^t (Unit: km) and it is not shorter than d_r^s, the shortest distance from its origin to destination. The detour ratio of r is $\delta(r) = d_r^t - d_r^s / d_r^s$, which is the percentage of additional distance compared with d_r^s. Let $\bar{\delta} = (\sum_r d_r^t - \sum_r d_r^s)/(\sum_r d_r^s)$ denote the average detour ratio of all requests. Make sure that the detour ratio of any request does not exceed a threshold Δ: $\delta(r) \le \Delta$, which aims at not losing too much comfort.

QoS can be formulated by the equivalent waiting distance and detour ratio. It is detailed by Eqn (14). Therein α_1 and α_2 are weights, which can promote passenger experiences in the future, and are obtained from a survey of passengers. They actually reflect the preference of passengers. If passengers prefer shorter waiting time to travel time, α_1 should be larger than α_2, and vice versa. Generally, their values are 1, since most passengers focus on the total trip time. After trip service finishes, passengers can rate the service. QoS(r) would be higher if the passenger thinks the service is better. The PV system will try to improve QoS according to the ratings. Let $s(p,t)$ denote the speed of p, where t denotes time. Let D_r (Unit: km) denote the threshold of the total travel distance of

the PV which serves r from the time r_t to r_a, which is shown by Eqn. (15). The detail of travel constraint is presented by Eqn. (16). We wish that both waiting distance and detour ratio do not exceed their thresholds. Considering of a limited number of PVs, and in order to serve all the requests, equivalent waiting distance is not necessarily strictly guaranteed, but the detour ratio should be guaranteed.

$$\mathrm{QoS}(r) = \alpha_1 * \frac{d_r^s}{d_r^w} + \alpha_2 * \frac{1}{\delta(r)} \tag{14}$$

$$D_r = W_r + d_r^s * (1 + \Delta) \tag{15}$$

$$\int_{r_t}^{r_a} s(p,t)\,dt = \int_{r_t}^{r_p} s(p,t)\,dt + \int_{r_p}^{r_a} s(p,t)\,dt = d_r^w + d_r^t. \tag{16}$$

PCI (Precedence Constrained origin-destination Insertion) algorithm is presented by Algorithm 1. At some time t_s, the PV system selects the unassigned requests R', whose request submitting time is not later than t_s. Here, R' is different from R_2 in Section III. Then sort requests R' by the ascending order of their request submitting time. Line (1) can balance the waiting time of corresponding requests. To reduce the complexity, do not change the assignment of the requests which have been scheduled, yet have not been picked up by PVs. Line (4) evaluates capacity constraint. Lines (7–8) calculate and evaluate QoS constraints: if any request's detour ratio is larger than Δ, the cost of serving this request is infinity. Line (9) calculates the least insertion cost of r on the path of p. Line (12) gets the minimum insertion cost of r among all the PVs. Lines (13–18) mean that, if one PV can serve r, and the QoS of all requests (including r) is guaranteed, the path and service list of this PV should be both updated. Line (17) implies that, if the appropriate PV to serve r is not found, r has to wait until some PV can serve him/her under QoS constraints.

Algorithm 1: PCI Algorithm

Input : R', set of unscheduled requests
Output : $\{q_p\}, p \in P$, paths of PVs
$\{l_p\}, p \in P$, service lists of PVs

1: Put the requests with the longest waiting time to the highest scheduling rank;
2: **for** $r \in R'$ **do**
3: **for** $p \in P$ **do**
4: **if** $|l_p| \geq c_p$ **then**
5: $\pi_{r,p} \leftarrow \infty$;
6: **else**
7: Calculate $\pi_{r,p,i,j}$ using Eqn (13) for all possible insertion locations;
8: If there exists a request $r' \in \{r \bigcup l_p\}$, $\delta(r') > \Delta$, $\pi_{r,p,i,j} \leftarrow \infty$;
9: $\pi_{r,p} \leftarrow \min\{\pi_{r,p,i',j'}\}$;
10: **end**
11: **end**
12: $\pi_r \leftarrow \pi_{r,p',i'',j''} = \min\{\pi_{r,1}, \ldots, \pi_{r,N_p}\}$;
13: **if** $\pi_r \neq \infty$ **then**
14: $q_{p'} \leftarrow \mathrm{INSERT}(r, p', i'', j'')$; Update path
15: $l_{p'} \leftarrow l_{p'} \bigcup \{r\}$; Update service list
16: **else**
17: Put r to a waiting queue;
18: **end**
19: **end**

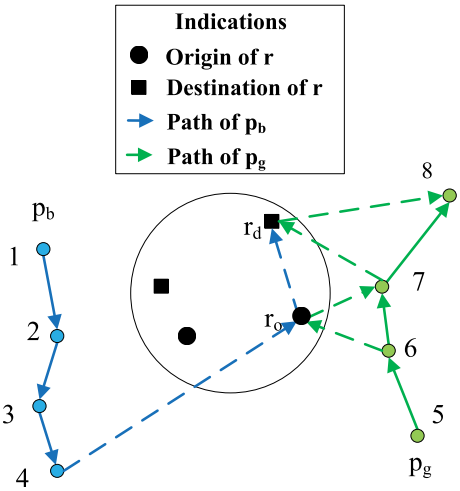


Fig. 3. Example of PCI.

The complexity of inserting one OD pair on the path of p is $O(L_p^2) = O(c_p^2)$. It is carried out by line (14) and the corresponding path and service list are updated afterwards. The complexity of PCI is $O(mN_pc_p^2)$.

Fig. 3 shows an example of PCI. The arrows denote the current paths of two PVs, p_b (the blue one) and p_g (the green one). One a new request r needs service (r_o is the origin, and r_d is the destination). The least insertion cost for (r_o, r_d) on the path of p_b is $\pi_{r,p_b} = d\{4, r_o, r_d\}$. The least insertion cost for (r_o, r_d) on the path of p_g is $\pi_{r,p_g} = d\{6, r_o, 7, r_d, 8\} - d\{6, 7, 8\}$. Assume QoS of all requests are guaranteed if the above operations are executed. If $\pi_{r,p_g} < \pi_{r,p_b}$, this request will be served by p_g, and the new path of p_g will be $\{5 \rightarrow 6 \rightarrow r_o \rightarrow 7 \rightarrow r_d \rightarrow 8\}$.

C. Local Optimization

After requests are assigned to PVs and paths are updated, how to locally optimize the path of each PV is discussed here. Traveling salesman problem (TSP) improvement based on k-change neighborhoods methods such as 2-OPT and 3-OPT [20] for the Lin-Kernighan TSP heuristic, can be used to optimize paths of PVs. The precedence, capacity, and QoS constraints should be added in the PV scenario. This paper does not consider the case $k \geq 3$, because the complexity would be too large. When tackling PVP, the check for one PV takes the time proportional to c_p. Therefore, the complexity of the local optimization solution using 2-OPT increases to $O(N_pc_p^3)$.

V. Performance Evaluation

In this section, three different transportation systems are simulated to study their characteristics and performance.

A. Experiment Settings

Experiment settings include the principle of generating requests, three scenarios (PVs, conventional vehicles, and Uber Pool), congestion model, and parameter settings.

The principle of generating trip requests is discussed here. This paper uses Shanghai daily traffic characteristics [21] to configure the peak and nonpeak traffic patterns. Fig. 4 shows the distribution of trip requests by time of the day (24 hours). Two 4-hour windows, 6:00–10:00 and 15:00–19:00, are referred as peak traffic time. Others are referred as nonpeak traffic time.

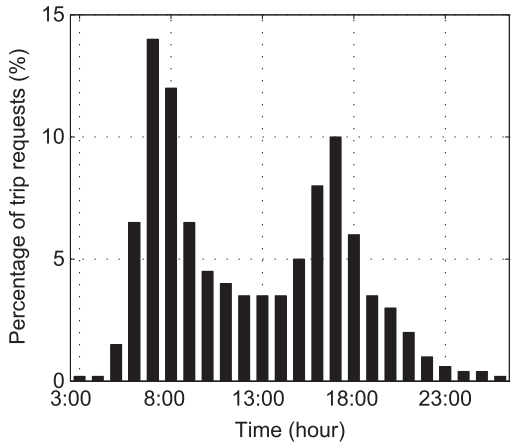


Fig. 4. Distribution of trip requests.

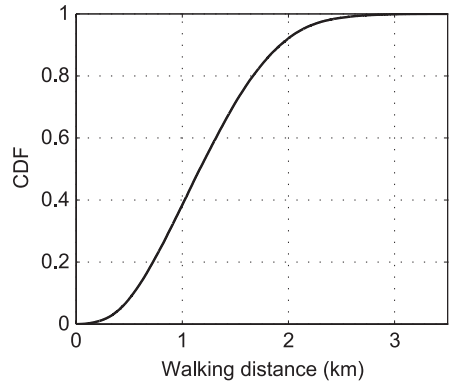


Fig. 5. CDF of walking distance.

Assume that, in each hour, the generating time obeys a uniform distribution, and the origins-destinations of requests are generated uniformly in the whole urban area. Let $n_{\rm sh}$ denote the number of passengers sharing one request. If $n_{\rm sh} = 1$, consider that each vehicle in Uber Pool can serve four riders with close origins and destinations, and similar request submitting time. Every four requests are named as a *request set*, which are generated with close origins and destinations. For simplicity, assume the request submitting time of the four requests is identical. The spatial model of generating requests is as follows: *First*, generate one request uniformly among the road network, with the distance from its origin to the destination not shorter than $d_{\rm lb}$. *Second*, in two squares (0.8 km $*$ 0.8 km) centering at its origin and destination, generate the other three requests making sure their origins and destinations are distributed uniformly in the two squares. If $n_{\rm sh} = 2$, each request set has two requests. If $n_{\rm sh} = 3$ or 4, each request set has only one request.

In performance study, consider the PV system as scenario I, $S_{\rm I}$; conventional vehicle system (details will be shown later) as scenario II, $S_{\rm II}$; and Uber Pool as scenario III, $S_{\rm III}$. For example, in $S_{\rm I}$, only PVs can travel and serve passengers. Here, we want to know how much transportation efficiency can be improved if using PV system compared with other transportation systems.

In $S_{\rm I}$ (PV system), the initial locations of PVs are randomly distributed in the urban area following a uniform distribution. These initial locations almost do not have any effect on the final performance if the number of requests in one day is very large, e.g., 100,000. PVs are planned to work 24 hours (one trip after another). Vacant PVs stop at the destinations of passengers they serve.

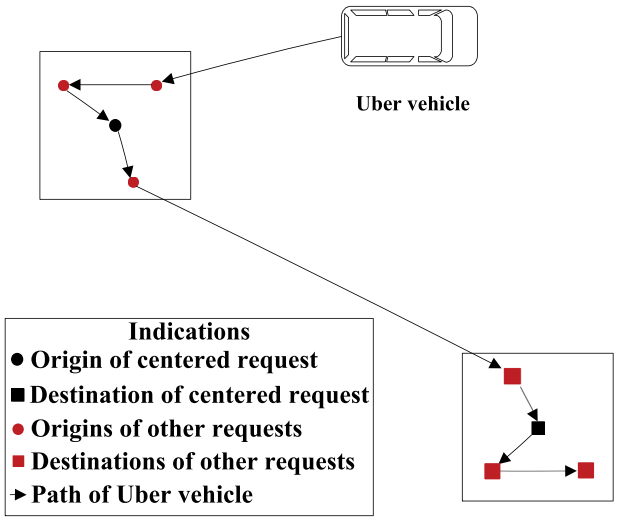


Fig. 6. Uber Pool.

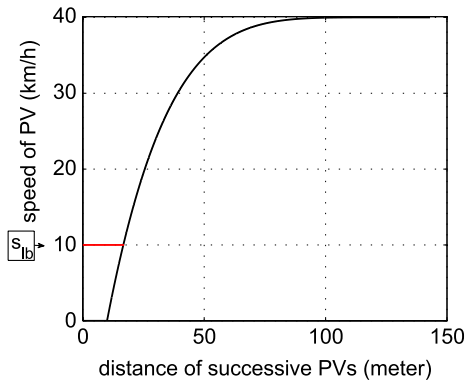


Fig. 7. Congestion model.

In $S_{\rm II}$, conventional vehicle (CV) system or conventional vehicles (CVs) include buses, cars (private cars), and taxis. The number of buses, cars, and taxis in the CV system is 500, 5,000, and 1,100 respectively. Therefore, the number of CVs ($N_{\rm CV}$) is 6,600. In one day (5:00–24:00), the bus interval at peak and non-peak time is 8 and 12 minutes, respectively. Configure l_b routes of buses, as well as their stops. Private cars always serve requests along the shortest paths from their origins to destinations. For a taxi trip request, calculate the distance from the trip origin to nearby on-call taxis, and then schedule the nearest taxi to fulfill the trip request. The initial locations of taxis are randomly distributed which follows a uniform distribution, which is the same as PVs in $S_{\rm I}$. A taxi also follows the shortest path to pick the next assigned request and then to reach its destination. Upon arrival at the destination, a taxi will stop there.

How to determine a bus, car, or taxi trip in $S_{\rm II}$? Select the trip requests whose request submitting time is in the time window 5:00–24:00, and assume all of them will be fulfilled by buses. The CDF of walking distance (including origin to the corresponding bus stop, and destination to the corresponding bus stop) if all passengers take buses is computed as shown in Fig. 5. The requests with short walking distance are assigned to buses, which can improve the traffic efficiency of buses, and others are assigned to cars or taxis. The average walking distance usually increases as percentage ρ_b increases. Let $\Omega(\rho_b)$ be the maximum walking distance as a function of ρ_b. For example, if $\rho_b = 0.4$, the maximum walking distance, Ω (0.4) is about 1 km. Thus, for a trip request r with r_t in the time window

TABLE II
VARIABLES AND DEFINITIONS USED IN SIMULATIONS

Variables	Definition	Unit	Values
$s_b^m, s_c^m, s_t^m, s_p^m, s_u^m$	maximum speed of buses, cars, taxis, PVs, and Uber vehicles respectively.	km/h	30, 40, 40, 40, 40
s_w	speed of walking.	km/h	5
t_p	time spent in picking one request.	second	6
t_d	time spent in dropping one request.	second	6
c_b, c_c, c_t, c_p, c_u	capacity of buses, cars, taxis, PVs, and Uber vehicles, respectively.	-	40, 4, 4, 16, 4
ρ_b, ρ_c, ρ_t	percentages of trip requests with buses, cars, and taxis in S_{II}.	-	0.7, 0.1, 0.2
n_c'	number of times each car is used in one day.	-	2
l_b	number of bus routes.	-	50
n_b	number of buses of each bus route.	-	10
N_b	total number of buses. $N_b = l_b * n_b$.	-	500
N_c	total number of cars.	-	5,000
N_t	total number of taxis.	-	1,100
N_{CV}	total number of CVs in S_{II}. $N_{CV} = N_b + N_c + N_t$.	-	6,600
N_u	total number of Uber vehicles in S_{III}.	-	1,400
n_{sh}	number of passengers sharing one request.	-	1∼4

5:00–24:00, calculate its walking distance to its bus stop, if it is shorter than $\Omega(\rho_b)$, this trip will be assigned as a bus trip. Otherwise, the trip request will be assigned as a car trip or a taxi trip based on the ratio of $\rho_c/(\rho_c+\rho_t)$ and $\rho_t/(\rho_c+\rho_t)$, respectively.

In $S_{\rm III}$ (Uber Pool), for a request set which has not been scheduled, choose the nearest empty vehicle, and it would travel to the nearest origin of requests, and then to the next nearest. . ., and to the last origin. Then, it travels to the nearest destination of requests, and then to the next nearest. . ., and to the last one until all the requests are dropped at their destinations. Fig. 6 shows the scenario of Uber Pool $(n_{\rm sh}=1)$.

To evaluate the traffic conditions, this paper introduces a traffic congestion model [22] based on car-following theory as Verhoef describes. If the distance between two successive vehicles is short than a critical distance $\beta_{\min}$ (Unit: meter), the speed is 0, and if the distance is longer than another critical distance $\beta_{\max}$ (Unit: meter), the speed is the maximum speed, otherwise, the speed is between the two thresholds. Take two successive PVs p_1 and p_2 (p_1 is in front of p_2) as an example, and let β_{p_1,p_2} (Unit: meter) denote the distance between p_1 and p_2. The congestion model is formulated by Eqn. (17). In the real world, the speed of two successive vehicles does not generally falls to 0 if they are very near. Therefore, modify the model and add one parameter $s_{\rm lb}$ (Unit: km/h) to denote the lower bound of speed. Fig. 7 illustrates the relationships between β_{p_1,p_2} and $s(p_2)$. Here, $s_p^m=40$, and $s_{\rm lb}=10$, and $\beta_{\min}=10$, and $\beta_{\max}=130$. In this paper, assume all type of vehicles (buses, cars, taxis, PVs and Uber vehicles) share the identical $s_{\rm lb}$, $\beta_{\min}$ and $\beta_{\max}$.

$$s(p_2)=\begin{cases}0, & \text{if } \beta_{p_1,p_2}\le\beta_{\min}\\ s_p^m-\frac{s_p^m}{(\beta_{\max}-\beta_{\min})^5} & \\ \quad *(\beta_{\max}-\beta_{p_1,p_2})^5, & \text{if } \beta_{\min}<\beta_{p_1,p_2}<\beta_{\max}\\ s_p^m, & \text{if } \beta_{p_1,p_2}\ge\beta_{\max}.\end{cases} \tag{17}$$

Simulations are built in ESRI ArcGIS 10.2 using C++ under Windows OS based on the road network of Shanghai City, particularly, in the downtown area of about 50 km^2. Within a 24-hour time window and following the peak and nonpeak traffic patterns, there are 100,000 trip requests which are geographically distributed in that area. The speed of all the vehicles is calculated by the traffic congestion model based on car-following theory. All the vehicles travel as the above description to serve passengers until all the passengers arrive at destinations. Table II lists the variables and definitions.

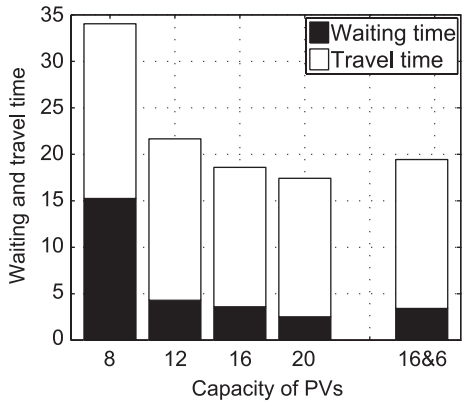


Fig. 8. Waiting–travel time versus capacity.

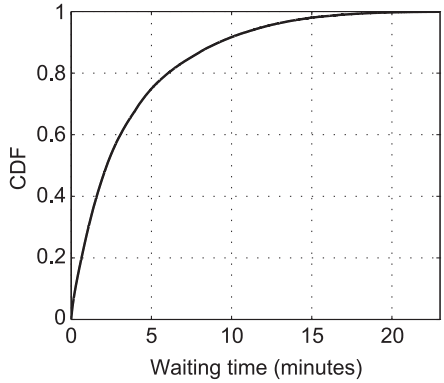


Fig. 9. CDF of waiting time in the PV system.

In this paper, $d_{\rm lb}$ is set to 5, $s_{\rm lb}$ is set to 10, $\beta_{\min}$ is set to 10, $\beta_{\max}$ is set to 130, and Δ is set to 0.3. The value $d_{\rm lb}$ makes the performance of the PV system and other transportation systems more stable. If $d_{\rm lb}$ varies a lot, the performance of the PV system and Uber Pool varies a lot. With respect to $s_{\rm lb}$, in the real world, if the distance between two consecutive vehicles is less than 10 meters, the speed decreases to a low value, e.g., 10 km/h. Each passenger may prefer low detour ratio, which reflects the quality of service. It will be widely accepted if the detour ratio is not larger than 0.3, however, the price may be lower if Δ is higher.

B. Performance

The capacity of PVs is determined by the traffic flow in simulations. Then six metrics (e.g., waiting-travel time, vehicles' total

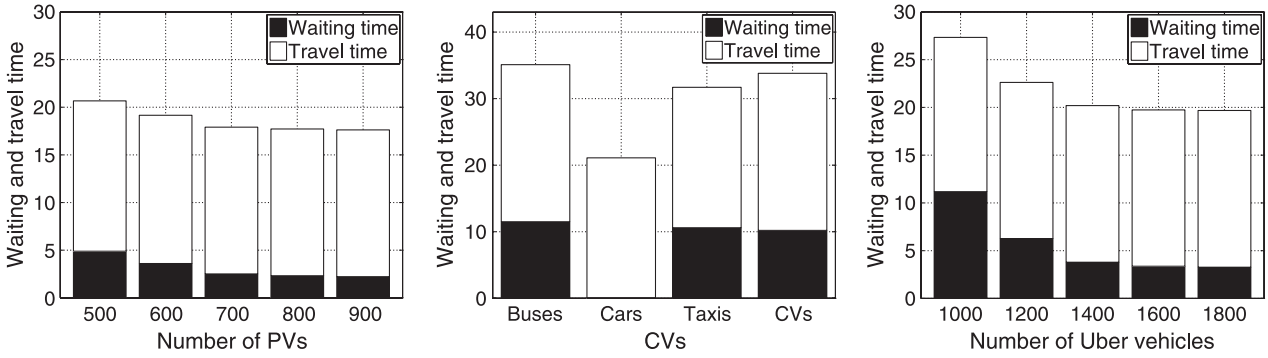


Fig. 10. Waiting–travel time in the PV system, CV system, and Uber Pool.

travel distance) are presented to evaluate the performance of the PV system compared with CV system and Uber Pool.

How to determine the capacity of PVs? As can be seen from Fig. 8, when the capacity of PVs increases, the average waiting time and travel time decrease. Small capacity will lead to small sharing factor and degrade performance. However, big capacity will not lead to full utilization of vehicles. From Fig. 8, when capacity is larger than 16, the performance almost does not improve, therefore c_p is set to 16 in simulations. This paper sets the different capacity of PVs for peak and nonpeak time ($N_p = 600$, $n_{\rm sh} = 1$). At the peak time, use the PVs with large capacity (e.g., $c_p = 16$), while at nonpeak time, use the PVs with small capacity (e.g., $c_p = 6$). The total time (waiting plus travel time) is only about one minute more than the case with strictly fixed capacity $c_p = 16$.

The *first* metric is the waiting and travel time. Fig. 9 shows CDF of waiting time of passengers in the PV system (600 PVs): The maximum waiting time is about 24 minutes. 75% of passengers should wait less than 5 minutes, and 17% of passengers should wait 5–10 minutes, and only 8% of passengers should wait for more than 10 minutes. The average waiting time is 3.6 minutes. The principle of scheduling rank of the proposed algorithm can reduce the average waiting time. Therefore, the short waiting time and low detour ratio both guarantee the personal comfort of passengers. Fig. 10 illustrates the waiting-travel time of people in the PV system, CV system and Uber Pool, respectively. The average travel time is 15 minutes in the PV system. With respect to CVs, the waiting time for buses is split into three parts: the time of walking from the origin to corresponding bus stop, the time of waiting for buses at a bus stop, and the time of walking from corresponding bus stop to the destination. Clearly, the total time of the people who take cars is the shortest, because there is no waiting time. In the PV system or Uber Pool, as N_p or N_u increases the total time (waiting time plus traveling time) decreases. Uber Pool has a larger number of vehicles and longer time compared to PVs due to two reasons. On one hand, the capacity of Uber vehicles is 4, which is not as large as that of PVs. On the other hand, as more vehicles lead to higher traffic congestion, the average speed decreases.

In summary, 600 PVs or 1,400 Uber vehicles can produce a better performance (i.e., total time, waiting time plus travel time) than that by 6,600 CVs since they reduce the road congestion and hence increase the vehicle speed. The number of vehicles in the PV system is reduced by around 90% and 57% compared with the CV system and Uber Pool, respectively.

The *second* metric is the total travel distance of vehicles, which is proportional to the consumed energy. Fig. 11 plots the total distance traveled in three scenarios, $S_{\rm I}$, $S_{\rm II}$, and $S_{\rm III}$. The total traveling distance of PVs is reduced by 34% and 14% compared with the CV system and Uber Pool, respectively.

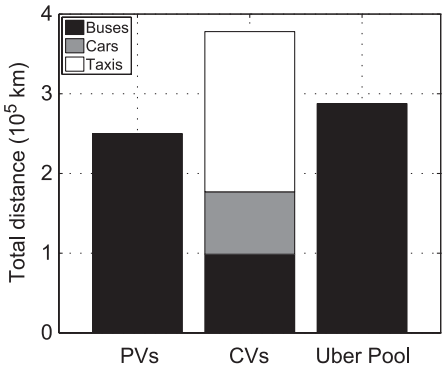


Fig. 11. Total travel distance of vehicles.

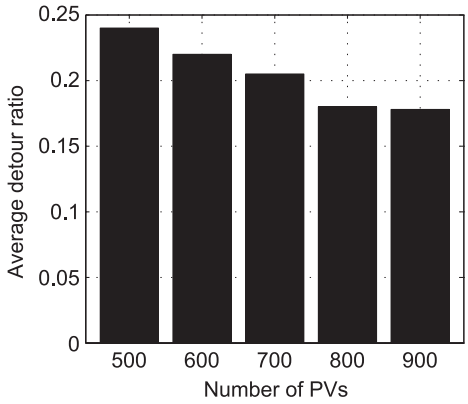


Fig. 12. Average detour ratio in the PV system.

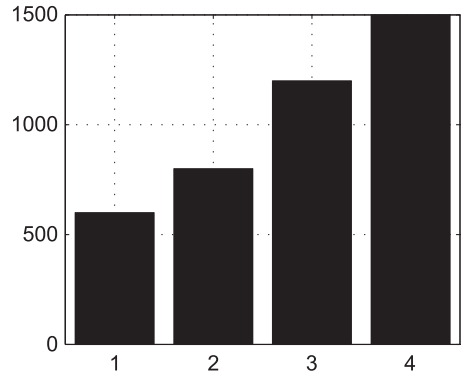


Fig. 13. Number of required PVs versus $n_{\rm sh}$.

The *third* metric is the detour ratio of requests in the PV system, which reflects the service quality. Detour is a cost of sharing. For ridesharing, a PV may need to detour leading to a

TABLE III
AVERAGE SPEED OF VEHICLES ($n_{\rm sh} = 1$, $N_p = 600$, $N_u = 1{,}400$)

	Peak Time 6:00-10:00	Nonpeak Time 10:00-15:00	Peak Time 15:00-19:00	Nonpeak Time 19:00-6:00	Peak Time	Nonpeak Time
$s_{\rm bus}$	20.98	22.11	21.96	23.10	21.64	22.65
$s_{\rm car}$	23.24	26.63	24.64	28.92	24.06	27.11
$s_{\rm taxi}$	23.37	25.66	23.55	29.10	23.48	27.27
$s_{\rm PV}$	32.75	34.49	33.66	34.91	33.34	34.60
$s_{\rm Uber}$	31.12	34.68	32.32	34.93	31.72	34.77

TABLE IV
SHARING FACTORS FOR PVS, BUSES, CVS, AND UBER VEHICLES

	Peak Time 6:00-10:00, 15:00-19:00	Nonpeak Time 10:00-15:00, 19:00-6:00
$\phi_{\rm PV}$	4.54	2.15
$\phi_{\rm bus}$	8.02	3.91
$\phi_{\rm CV}$	1.99	1.18
$\phi_{\rm Uber}$	3.51	3.40

reduction in the transportation efficiency. As shown in Fig. 12, the average detour ratio $\bar{\delta}$ decreases with more PVs. When the number of PVs decreases, the traveling time increases because the detour ratio increases.

The *fourth* metric is the required number of vehicles. Consider the scenario of multiple passengers per request. Fig. 13 shows the required number of vehicles in the PV system, where a benchmark is set to 20 minutes (total time, waiting plus travel time). If a request has more than one passenger, to reach a similar performance achieved in a case of single passenger per request, more PVs will be required. In the following, assume that only one passenger per request.

The *fifth* metric is the average speed, which reflects the traffic congestion. Table III illustrates the average speed during several time periods, where $s_{\rm bus}$, $s_{\rm car}$, $s_{\rm taxi}$, $s_{\rm PV}$, and $s_{\rm Uber}$ (Unit: km/h) denote the average speed of buses, cars, taxis, PVs, and Uber vehicles, respectively. Less number of vehicles in the PV system and Uber Pool lead lower level of congestion, resulting in a higher vehicle speed.

The *sixth* metric is sharing factor, ϕ, which is the average number of requests in a vehicle. Sharing factor of PVs $\phi_{\rm PV}$ can be calculated by Eqn. (18). Therein d_r^s is the shortest path distance of the request r (from origin to destination), and d_r^t is the actual travel distance of r, and $d_{\rm PV}$ is the total travel distance of all PVs. $d_{\rm PV} = \sum_r d_r^s/((\phi_{\rm PV})/(1+\bar{\delta}))$. Thus, $\phi_{\rm PV}/1+\bar{\delta}$ measures the quality of a routing algorithm. Now, this paper introduces a factor in T-share [17], relative distance rate (RDR). The relationship between $\phi_{\rm PV}$ and ${\rm RDR}_{\rm PV}$ is also shown by Eqn. (18). Simulations show that RDR of the PV system is about 27%.

$$\phi_{\rm PV} = \frac{1}{{\rm RDR}_{\rm PV}} = \frac{\sum_r d_r^t}{d_{\rm PV}} = (1+\bar{\delta}) * \frac{\sum_r d_r^s}{d_{\rm PV}}. \qquad (18)$$

In Table IV, the following sharing factors are summarized, $\phi_{\rm PV}$, $\phi_{\rm bus}$, $\phi_{\rm CV}$ and $\phi_{\rm Uber}$ over peak time and nonpeak time. The sharing factor is higher at the peak time due to more requests. The buses have a higher sharing factor due to their large capacity and fixed routes. The sharing factor of Uber Pool at the nonpeak time is higher than that of PVs as Uber Pool can organize the requests better, however with the cost of longer waiting and traveling time.

VI. DISCUSSION AND CONCLUSION

We consider a new paradigm, the PV system, which is supported by a government or a company to provide ridesharing trip service in urban areas to replace the current buses, (private) cars and taxis, although there are many problems to be handled. As a ridesharing platform, the PV system is a new approach and is different from T-Share and Via [23] for several reasons. *First*, the PV system is designed as a potentially widely used transportation system to replace cars, taxis, and buses in urban areas. T-Share is only used in taxi sharing. *Second*, the aim of our scheduling algorithm is to provide dynamic low-cost ridesharing trips with service guarantee such as low detour. However, the scheduling strategy of T-Share is: each passenger is served by the taxi with minimum increasing of distance. There is not trip service guarantee both in T-Share and Via. *Third*, PVs cooperate with each other to achieve better performance. For example, passengers can transfer among different PVs. However, in other ridesharing systems (e.g., T-Share and Via), drivers compete with each other for more profit.

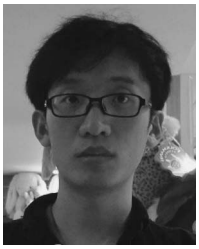
To reduce PVs' travel distance with preserving short waiting-travel time, this paper defines the PVP problem. An optimal solution through MILP is proposed. Then PCI algorithm for PVP is proposed, and then a local optimization method is introduced for performance improvement. Its performance has been studied with large simulations. The proposed algorithm can be practical in the near future. Simulation results show that PCI has good performance, which can greatly reduce the number of vehicles, travel distance and trip time. The number of vehicles in the PV system can be greatly reduced compared with CV system, and Uber Pool. Therefore, traffic congestion is mitigated. More importantly, the transportation cost for society, as well as the cost for individuals, will be significantly reduced.

In the PV system, if some passengers require too high comfort, e.g., the detour ratio is near to 0, the PV system may not provide such trip service since serving other passengers may cause some detour distance. Some other problems, e.g., pricing, safety, privacy, charging, and parking are very important issues yet will be studied in the future.

REFERENCES

[1] What's Happening With Car Occupancy? 2011. [Online]. Available: http://chartingtransport.com/?s=occupancy
[2] Public Transportation in Singapore. [Online]. Available: http://worksingapore.com/articles/live_4.php
[3] Google's Driverless Cars are 'Safer' Than Human Drivers, 2013. [Online]. Available: http://www.telegraph.co.uk/technology/google/10411238/Googles-driverless-cars-are-safer-than-human-drivers.html
[4] [Online]. Available: http://my.teslamotors.com/models/design
[5] M. Qiu, M. Zhong, J. Li, K. Gai, and Z. Zong, "Phase-change memory optimization for green cloud with genetic algorithm," *IEEE Trans. Comput.*, vol. 64, no. 12, pp. 3528–3540, Dec. 2015.
[6] D. N. Anderson, "Not just a taxi? For-profit ridesharing, driver strategies, and VMT," *Transportation*, vol. 41, no. 5, pp. 1099–1117, Sep. 2014.

[7] N. Bicocchi and M. Mamei, "Investigating ride sharing opportunities through mobility data analysis," *Pervasive Mobile Comput.*, vol. 14, pp. 83–94, Oct. 2014.
[8] V. Handke and H. Jonuschat, *Flexible Ridesharing*. Berlin, Germany: Springer-Verlag, 2013.
[9] B. Cici, A. Markopoulou, E. Frias-Martinez, and N. Laoutaris, "Assessing the potential of ride-sharing using mobile and social data: A tale of four cities," in *Proc. ACM Int. Joint Conf. Pervasive UbiComp*, 2014, pp. 201–211.
[10] P. Santi *et al.*, "Quantifying the benefits of vehicle pooling with shareability networks," *Proc. Nat. Acad. Sci.*, vol. 111, no. 37, pp. 13290–13294, Sep. 2014.
[11] L. P. Alexander and M. C. González, "Assessing the impact of real-time ridesharing on urban traffic using mobile phone data," in *Proc. ACM Int. Workshop UrbComp*, 2015, p. 9.
[12] M. Barth, M. Todd, and L. Xue, "User-based vehicle relocation techniques for multiple-station shared-use vehicle systems," presented at the Transportation Research Board 80th Annual Meeting, Washington, DC, USA, 2004, Paper 04-4161.
[13] S. Yan, C.-Y. Chen, and S.-C. Chang, "A car pooling model and solution method with stochastic vehicle travel times," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 47–61, Feb. 2014.
[14] J. Yousaf, J. Li, L. Chen, J. Tang, and X. Dai, "Generalized multipath planning model for ride-sharing systems," *Frontiers Comput. Sci.*, vol. 8, no. 1, pp. 100–118, Feb. 2014.
[15] W. He, K. Hwang, and D. Li, "Intelligent carpool routing for urban ridesharing by mining GPS trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2286–2296, Oct. 2014.
[16] D. Zhang *et al.*, "coRide: Carpool service with a win-win fare model for large-scale taxicab networks," in *Proc. ACM Conf. Embedded Netw. SenSys*, 2013, p. 9.
[17] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *Proc. IEEE ICDE*, 2013, pp. 410–421.
[18] I. L. Gørtz, "Hardness of preemptive finite capacity dial-a-ride," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Berlin, Germany: Springer-Verlag, 2006, pp. 200–211.
[19] A. Abadi, T. Rajabioun, and P. Ioannou, "Traffic flow prediction for road transportation networks with limited traffic data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 653–662, Apr. 2015.
[20] L. A. Wolsey and G. L. Nemhauser, *Integer and Combinatorial Optimization*. Hoboken, NJ, USA: Wiley, 2014.
[21] Spatio-Temporal Distribution of Trips in Shanghai, 2011. [Online]. Available: http://sh.eastday.com/qtmt/20110309/u1a863059.html
[22] E. T. Verhoef, "An integrated dynamic model of road traffic congestion based on simple car-following theory: Exploring hypercongestion," *J. Urban Econ.*, vol. 49, no. 3, pp. 505–542, May 2001.
[23] [Online]. Available: http://ridewithvia.com/



Ming Zhu is currently working toward the Ph.D. degree in the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China.

His research interests are in the areas of cyber–physical systems, Internet of Things, artificial intelligence, smart grids, and wireless communications.

Xiao-Yang Liu received the B.Eng. degree in computer science from Huazhong University of Science and Technology, Wuhan, China, in 2010. He is currently working toward the Ph.D. degree with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China.

His research interests include wireless communication, distributed systems, big data analysis, cyber security, and sparse optimization.

Feilong Tang (M'10) received the Ph.D. degree in computer science from Shanghai Jiao Tong University (SJTU), Shanghai, China, in 2005.

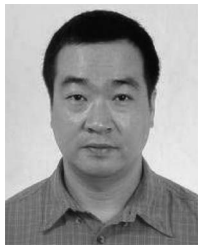
He is currently a Full Professor with the Department of Computer Science and Engineering, SJTU. His research interests include mobile cognitive networks, wireless sensor networks, clouding computing, and algorithm design and evaluation.

Dr. Tang is a Fellow of the Institution of Engineering and Technology. He has served as a Program Cochair for eight international conferences.

Meikang Qiu (M'03–SM'07) received the B.E. and M.E. degrees from Shanghai Jiao Tong University, Shanghai, China, in 1992 and 1998, respectively, and the M.S. and Ph.D. degrees in computer science from The University of Texas at Dallas, Richardson, TX, USA, in 2003 and 2007, respectively.

He is currently an Associate Professor of computer science with Pace University, New York, NY, USA, and an Adjunct Professor with Columbia University, New York. His research interests include cloud computing, data storage and security, embedded systems, cyber security, mobile networks, etc.

Dr. Qiu is a Senior Member of the Association for Computing Machinery.



Ruimin Shen received the Bachelor's and Master's degrees from Tsinghua University, Beijing, China, in 1988 and 1991, respectively, and the Ph.D. degree from the University of Hagen, Hagen, Germany, in 2013.

He is currently a Professor of computer science and engineering with Shanghai Jiaotong University, Shanghai, China. His research interests include cutting-edge technologies for "available anywhere and updatable anytime" distance education, such as mobile learning, standard natural classrooms, knowledge discovery, and data mining.

Wennie Shu (M'90–SM'99) received the Ph.D. degree from University of Illinois at Urbana–Champaign, Champaign, IL, USA.

She was with Yale University, New Haven, CT, USA; State University of New York at Buffalo, Buffalo, NY, USA; and University of Central Florida, Orlando, FL, USA. She is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM, USA. Her research interests include resource management, distributed systems, wireless networks.

Min-You Wu (S'84–M'85–SM'96) received the M.S. degree from the Graduate School of Academia Sinica, Beijing, China, in 1981 and the Ph.D. degree from Santa Clara University, Santa Clara, CA, USA, in 1984.

He is a Professor with the Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include grid computing, wireless networks, parallel and distributed systems, and compilers for parallel computers.