An Edge Computing Based Public Vehicle System for Smart Transportation

Jie Lin, Wei Yu, Xinyu Yang, Peng Zhao, Hanlin Zhang, and Wei Zhao

***Abstract*—As a key smart transportation service, public vehicle systems are intended to improve traffic efficiency and vehicle occupancy ratios, and to reduce the number of vehicles on roads, by inducing travelers to share rides with others. Despite the clear logic behind this service, achieving a viable model for matching multiple riders to vehicles with low latency and high satisfaction remains an open issue. In this paper, we propose an Edge Computing based Public Vehicle (ECPV) system to improve traffic efficiency and vehicle occupancy ratios by scheduling ridesharing among travelers and reduce the delay of decision making by leveraging edge computing. Particularly, by introducing a metric of traveler satisfaction jointly considering travel time, distance, and costs (i.e., charge), we formalize public vehicle scheduling problem as an optimization problem with maximizing traveler satisfaction as objective to reduce travel time and improve traffic efficiency. Further, as achieving globally optimal matching of rides and travelers is time consuming, to reduce the delay of decision making (i.e., response to ride-traveler pairs matching) and improve real time ridesharing service and traveler satisfaction, an edge computing based ride request transmission mechanism and a tree based heuristic matching mechanism are proposed to effectively exchange ride requests and select appropriate vehicles for mostof ride requests on edge devices. Also, our ECPV system considers driverless cars as public vehicles to remove the subjective influence of drivers and improve the efficiency of vehicle scheduling, and introduces a graph partition based depot placement mechanism to determine vehicleparking locations and balance vehicle distribution across system. Through extensive performance evaluations, our experimental results show that our proposed ECPV system can effectively match ride requests to public vehicles, reduce travel times and distances for all trips, reducecharges to travelers, and improve vehicle occupancy.**

***Index Terms*—Edge computing, internet of things, mobile service, public vehicle system and ridesharing.**

I. Introduction

IN SMART transportation, satisfying the transportation needs of travelers and improving traffic efficiency (i.e., improving vehicle occupancy ratio and reducing the number of vehicles on road) are essential for public vehicle (PV) systems to be successful [1]–[4]. To achieve these goals, a number of schemes have focused on ridesharing among travelers (via vehicle and ride scheduling), by matching ride requests to both idle and in-service vehicles [3]–[12], generally by minimizing total travel time or distance [7], [8], [10]. At the same time, other efforts have focused on reducing computational overhead for matching vehicle-rider pairs to reduce ride request response latency, and reduce the number of active public vehicles on the roads while still guaranteeing traffic efficiency, thereby mitigating traffic congestion [3]–[5], [13].

Nonetheless, PV systems that consider the integrated impacts of multiple personal preferences (e.g., travel distance, travel time, and travel charge) have not been carefully investigated. In addition, a typical ridesharing decision will be completed in a data center that is far away from the users, leading to long latency, especially when a large number of ride requests in a short time period, directly affecting the decision time of ride-traveler pairs matching and Quality of Experience (QoE) of travelers. Thus, this calls for the design of a distributed public vehicle system that considers multiple preferences of travelers in vehicle scheduling, reduces the volume of computational tasks in the data center to reduce the response latency of ride requests, and achieves high vehicle occupancy ratio to improve traffic efficiency.

To address these issues, in this paper we propose an **E***dge* **C***omputing Based* **P***ublic* **V***ehicle* (ECPV) System to improve traffic efficiency and vehicle occupancy ratios by scheduling public vehicles with consideration of traveler satisfaction and real time service. Particularly, edge computing is introduced in our paper to reduce the latency and improve traveler QoE. In our proposed system, unmanned ground vehicles (i.e., self-driving cars) are considered to be public vehicles to enable travelers to reach their destinations. Unmanned ground vehicles can not only eliminate the subjective influence of route selection during vehicle scheduling, but also enable special traffic rules (reversible lanes, etc.), thereby improving the efficiency of vehicle scheduling and traffic safety, as well as the great quality of traveler experiences.

The contributions in this paper are summarized as follows.

First, the traveler's satisfaction is introduced in the ECPV system to comprehensively reflect the utilities/profits of travelers in ridesharing and public vehicle (PV) scheduling (i.e., matching rider-vehicle pairs). Our ECPV system is formalized as an optimization problem with the objective of maximizing traveler satisfaction so as to reduce travel time and improve traffic efficiency. To achieve reasonable travel charge sharing

Manuscript received January 14, 2020; revised July 13, 2020; accepted September 27, 2020. Date of publication October 2, 2020; date of current version November 12, 2020. The review of this article was coordinated by Dr. A.-C. Pang. *(Corresponding author: Wei Yu.)*

Jie Lin, Xinyu Yang, and Peng Zhao are with the School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: jielin@mail.xjtu.edu.cn; yxyphd@mail.xjtu.edu.cn; p.zhao@mail.xjtu.edu.cn).

Wei Yu is with the Department of Computer and Information Science, Towson University, Towson, MD 21252 USA (e-mail: wyu@towson.edu).

Hanlin Zhang is with the College of Computer Science and Technology, Qingdao University, Qingdao 266071, China (e-mail: hanlin@qdu.edu.cn).

Wei Zhao is with the American University of Sharjah, Sharjah 26666, United Arab Emirates (e-mail: wzhao@aus.edu).

Digital Object Identifier 10.1109/TVT.2020.3028497

among ridesharing travelers, a travel charge model is proposed to determine the charge for each traveler, where a high ridesharing efficiency (or vehicle occupancy ratio) yields a low travel charge for each participant, thereby encouraging travelers to share rides.

Second, to reduce the delay of decision making, an edge computing based ride request transmission (ECRT) mechanism and a tree based heuristic matching (THM) mechanism are jointly proposed to transmit ride requests launched by travelers and select appropriate vehicles to match most of the ride requests in edge devices, achieving effective matching of rider-vehicle pairs with the objective of achieving high traveler satisfaction, traffic efficiency, and real time service of ride request response. Benefiting from the distributed architecture and edge devices being close to end-users, the *edge computing based ride request transmission mechanism* can deliver ride requests to nearby network edge devices, which then *tree based heuristic matching mechanism* can match vehicle-rider pairs and return PV scheduling responses quickly. By doing this, computation tasks can effectively be offloaded from cloud to edge devices so that the latency of ride request response can be reduced and the real time service of ride-traveler pairs matching can be realized. Several ride matching strategies are also considered in our *tree based heuristic matching mechanism*. Via the collaborative computing of vehicles, edge devices and cloud, our ECPV system can effectively match ride requests to vehicles with high traveler satisfaction and vehicle occupancy ratio, as well as low ride request response latency, thereby improving traffic efficiency and providing real time service of ride-traveler pairs matching.

Third, to effectively balance vehicle distribution and improve traffic efficiency in the global transportation system, a graph partition based depot placement (GPDP) mechanism is proposed to select appropriate locations for vehicle parking. The number of vehicles on roads directly affects the speed and efficiency of traffic in transportation systems, and reducing the number of vehicles on the road will improve traffic efficiency. In our proposed ECPV system, idle vehicles that have completed their current arranged ride requests will travel to nearby depots to wait new requests. Thus, choosing suitable locations to place depots can effectively balance vehicle distribution and improve traffic efficiency in the global transportation system.

Lastly, a number of simulation evaluations have been conducted to demonstrate the effectiveness of our ECPV system in terms of traveler satisfaction, travel time, travel distance, travel charge and vehicle occupancy ratio with the implementation of several ride matching strategies. Our experimental results show that, in comparison with the scenario lacking any ridesharing and existing schemes (T-Share [8] and PTRider [14]), our ECPV can effectively reduce the delay of decision making, improve traveler satisfaction, reduce travel charges for travelers, and reduce total travel distance/time, as well as improve vehicle occupancy ratio.

The remainder of the paper is organized as follows: In Section II, we present the system model. In Section III, we introduce the problem formalization. In Section IV, we present our proposed ECPV system. In Section V, we provide our performance evaluation on the effectiveness of our ECPV system. We conduct a literature review in Section VI. Finally, we conclude the paper in Section VII.

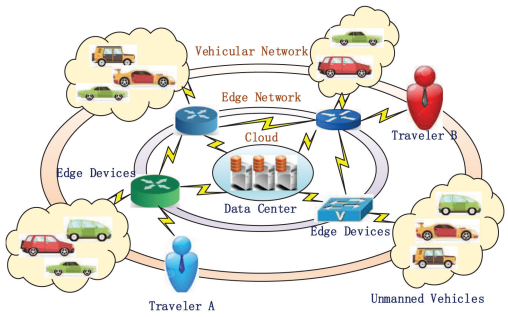


Fig. 1. System model.

II. System Model

In our system model, a three-tier network framework, including vehicular network, edge network, and cloud, is considered in our model to deliver ride requests of travelers and vehicle scheduling responses of the PV system, as shown in Fig. 1, where the unmanned ground vehicles on roads are organized as a vehicular network, and road-side units serving as edge devices are organized as an edge network. We consider the deployment of road-side units as edge devices in our system, with the intention of improving the operation and effectiveness of the vehicular network without road-side units.

Vehicles can communicate with edge devices through wireless networks to share route information and receive vehicle scheduling responses, while the cloud (i.e., data center) can communicate with edge devices via wired networks to collect and store global data and provide global analysis. The edge devices can communicate with each other through either wireless network or wired network. In our system, with consideration for the scalability, flexibility, and deployment and maintenance cost, wireless networks are considered to connect edge devices.

In our abstracted road network, each vertex can be deployed with an edge device, and the edge network organized by all edge devices can fully cover the road network. Vehicles always communicate with the nearest edge devices to share information. To reduce the deployment cost, road side units can be utilized as edge devices with the addition of upgraded computation resources. Thus, upgrading existing roadside units in vehicular networks will be an effective way to reduce the cost of edge device deployment.

Our model assumes that travelers can launch a ride request at anytime and anywhere with no need to make a vehicle appointment in advance. Further, vehicles cannot remove any accepted ride requests from their scheduled route. When a vehicle becomes idle (i.e., no ride request to be served in the vehicle), the vehicle will travel to the nearest depot to wait for new ride requests. Our model assumes that a ride request will only be served by one vehicle in our ECPV system, and completing a trip with multiple vehicles is out of the scope of this paper. Our model also assumes that vehicles always transit the shortest route between destinations.

III. Problem Formalization

In the following, we first introduce some definitions and a travel charge model. We then formalize the public vehicle

TABLE I
NOTATION

t:	Time slot
rq:	Ride request
pv:	Public vehicle
ed:	Edge device
s_{rq}:	The pick-up point of ride request rq
d_{rq}:	The drop-off point of ride request rq
RE_{pv}:	The set of ride requests that are being or will be served by public vehicle pv
RQ_{in}, RQ_{new}:	The set of in-vehicle ride requests and new ride requests, respectively
SL:	The share level of a ride request or a vehicle
Num_{rq}:	The number of travelers of a ride request
Num_{pv}:	The number of public vehicles in the whole system
Num_{depot}:	The number of depots placed in the transportation system
C_{depot}:	The capacity of a depot
RL_{pv}:	The scheduled route list of vehicle pv
RS_{rq}:	The set of route segments of ride request rq
TI_{pv}:	The insertion-tree of vehicle pv
SAT_{rq}:	The satisfaction of ride request rq
$T_{rq}^{real}, D_{rq}^{real}, P_{rq}^{real}$:	The real travel time, distance, and charge of ride request rq, respectively
$T_{rq}^{exp}, D_{rq}^{exp}, P_{rq}^{exp}$:	The expected travel time, distance, and charge of ride request rq, respectively
C_{pv}:	The total capacity of vehicle pv (i.e., the number of seats in vehicle pv)
c_{pv}^{w}:	The capacity of vehicle pv at point w, i.e., the number of remaining seats in vehicle pv at a point (i.e., location) w
$\Delta T_{rq}, \Delta D_{rq}, \Delta P_{rq}$:	The detour time, distance, and charge of ride request rq, respectively
$\alpha, \beta.\gamma$:	Preference parameters of travel time, distance, and cost in travel satisfaction, respectively
Pr:	The cost for one vehicle to travel one mile

scheduling in our ECPV system and discuss the selection of corresponding parameters. All notations are shown in Table I. In this paper, a ride request rq is used to represent the traveler who launches the ride request.

A. Definitions

To clearly formalize our public vehicle scheduling problem, some concepts are defined as follows.

Definition 1. Ride-request: a ride request $rq =< rq, t_{rq}, t_{pick}, Num_{rq}, s_{rq}, d_{rq}, SL_{rq} >$ represents that at time t_{rq}, Num_{rq} travelers plan to travel from the pick-up point s_{rq} to the drop-off point d_{rq} with sharing level SL_{rq}, and t_{pick} is the time slot when the ride request is picked up.

The sharing level, denoted as SL, is introduced in our system and includes three levels: (i) *No-Share* represents travelers who are not willing to share rides with other travelers, (ii) *All-Share* represents travelers who are willing to share rides with anyone, and (iii) *Female-only-Share* represents travelers who are only willing to share rides with females. The *Female-only-Share* level can only be requested by female travelers and is used to improve the travel safety of females. In the same manner, *Male-only-Share* can be used as well.

Definition 2. Vehicle Route List: a vehicle route list $RL_{pv} = < SL_{pv}, w_1, c_{pv}^{w_1}, \ldots, w_n, c_{pv}^{w_n} >$ represents all points (locations) that vehicle pv will pass through to complete its transportation service, where w_i represents either the pick-up point or the drop-off point of a ride request that is arranged for vehicle pv, $c_{pv}^{w_i}$ represents the capacity of vehicle pv (i.e., the remaining acceptable number of travelers at point w_i), and SL_{pv} is the sharing level of vehicle pv and is co-determined by sharing levels of all in-vehicle travelers.

According to Definitions 1 and 2, in our ECPV system, public vehicle scheduling can be considered as a point insertion problem. Here, for a new ride request, our ECPV system should select a suitable vehicle and insert the pick-up/drop-off points of the new ride request into the route list of this vehicle. The objective is to improve the traveler's quality of experience and the quality of service of the public vehicle system.

For any given trip, a smaller travel time, shorter travel distance, and lower travel charge will achieve a higher quality of traveler experience. Thus, in our ECPV system, traveler satisfaction is introduced to represent the quality of traveler experiences in ridesharing, where the personal travel time, distance, and charge are integrated to achieve a comprehensive reflection of travelers' preferences. The travel satisfaction of a valid inserted ride request (i.e., the traveler) in our system is defined as Equation (1), where the satisfaction is *Negative Exponential Distribution* with respect to travel time, travel distance, and travel charge.

$$\begin{cases} SAT_{rq} = e^{(1-(\alpha \cdot \Delta T + \beta \cdot \Delta D + \gamma \cdot \Delta P))} \\ \alpha + \beta + \gamma = 1 \end{cases}, \tag{1}$$

where α, β, and γ are the preference parameters, and ΔT, ΔD and ΔP are the detour time, detour distance, and detour charge for ride request rq, respectively, which are defined in Equation (2).

$$\Delta T = \frac{T^{real}}{T^{exp}}, \Delta D = \frac{D^{real}}{D^{exp}}, \Delta P = \frac{P^{real}}{P^{exp}}, \tag{2}$$

where $T^{real}, T^{exp}, D_{real}, D_{exp}, P^{exp}$, and P^{real} are the real and expected travel times, distances, and charges for a ride request, respectively, which are defined as follows.

In our system, the different preferences of users for travel time, travel distance and travel charge are considered in the travel satisfaction formalization (Equation (1)), where α, β and γ are introduced as the preference parameters to affect the rider-vehicle matching decision. The sum of α, β and γ is 1, and each of them represents a ratio against the total travel satisfaction. Our system aims to match rider-vehicle pairs with more satisfaction for users. That is, when our system makes a rider-vehicle matching decision, the different preferences of users have already been considered. Some other considerations can also be easily added in the satisfaction formalization in Equation (1). Using the travel satisfaction as a metric to match rider-vehicle pairs, all different preferences of users can be considered in our public vehicle system.

Definition 3: Real Travel Time, denoted as T^{real}, is the total time to complete a ride request, including waiting time and transit time, i.e., from the time slot when the ride request is launched to the time slot when the travelers arrive at their destinations, represented as

$$T_{rq}^{real} = t_{rq}^{drop} - t_{rq}^{launch}, \tag{3}$$

where t_{rq}^{drop} and t_{rq}^{launch} represent the time slot of ride request rq completed and launched, respectively.

Definition 4: Real Travel Distance, denoted as D^{real}, is the total distance traveled from pick-up point to the drop-off point along the scheduled route list, represented as

$$D_{rq}^{real} = D(s_{rq}, w_i) + \sum_{x=i+1}^{j-1} D(w_x, w_{x+1}) + D(w_j, d_{rq}), \quad (4)$$

where $D(w_x, w_{x+1})$ represent the distance from point w_x to point w_{x+1}, and $(s_{rq}, w_i, w_{i+1}, \ldots, w_{j-1}, w_j, d_{rq})$ belongs to the scheduled route list of the vehicle that accepts this ride request rq.

Definition 5: **Real Travel Charge**, denoted as P^{real}, is the total charge for a traveler to travel from the pick-up point to the drop-off point along the scheduled route list of the public vehicle. Travelers in the same vehicle with the same travel route can share the charge.

Also, our model introduces the expected travel time, distance, and charge, denoted as T^{exp}, D^{exp}, and P^{exp}, respectively. The expected travel time/distance represents the shortest time/distance from a pick-up point to a drop-off point by the shortest path algorithm (e.g., Dijkstra [15]) with consideration for the minimum travel time/distance on each road as the edge weight in road network $G < V, E >$. The expected travel charge for a traveler can be calculated as $P^{exp} = D^{exp} \cdot Pr$, where Pr is the cost for a vehicle to travel one mile.

B. Travel Charge Model

When a ride request is added into a route list of a vehicle, the detour distance will appear to pick-up/drop-off new travelers. For example, the route of vehicle pv can be represented as $< w_1, w_2, \ldots, w_n >$. When a new ride request rq is added, the route can be updated as $< w_1, \ldots, w_i, s_{rq}, w_{i+1}, \ldots, w_{j-1}, d_{rq}, w_j, \ldots, w_n >$. In this case, the detour travel distance will appear for travelers whose trip includes route segments $< w_i, w_{i+1} >$ and $< w_{j-1}, w_j >$. In this way, the travel charge of these in-vehicle travelers will increase, due to the increase in real travel distance. Obviously, this is not fair for these in-vehicle travelers.

Thus, to achieve fair travel charge, the new travelers in our ECPV system will owe a cost for the detour distance caused by their ride requests. Also, these new travelers will share the travel charge for inserted route segments, (i.e., $< w_i, w_{i+1} >$ and $< w_{j-1}, w_j >$), because they are considered sharing riders on these route segments by other in-vehicle travelers. Thus, the real travel charge of a new-in traveler can be presented as

$$P_{new-in}^{real} = (D(w_i, s_{rq}) + D(s_{rq}, w_{i+1}) - D(w_i, w_{i+1}) + D(w_{j-1}, d_{rq}) + D(d_{rq}, w_j) - D(w_{j-1}, w_j) + \sum_{x=i+1}^{j-1} \frac{D(w_x, w_{x+1})}{C_{pv} - c_{pv}^{w_x} + 1}) \cdot Pr, \quad (5)$$

where C_{pv} is the total capacity (number of seats) of vehicle pv, $c_{pv}^{w_x}$ is the number of remaining seats in vehicle pv at point w_x, and Pr is the cost for one vehicle to travel one mile.

After adding a new ride request, the travel charge to travelers who share route segments with the new-in ride request will be reduced and the travel charge reduction to in-vehicle traveler rq_i can be represented as

$$\bar{P}_{rq_i}^{real} = \sum_{<w_x, w_{x+1}> \in RS_{rq} \cap RS_{rq_i}} \frac{D(w_x, w_{x+1})}{C_{pv} - c_{pv}^{w_x}} \cdot \frac{Pr}{C_{pv} - c_{pv}^{w_x} + 1}, \quad (6)$$

where RS_{rq} and RS_{rq_i} are the set of route segments of the new-in ride request rq and in-vehicle ride request rq_i. Thus, the updated travel charge for in-vehicle traveler rq_i can be represented as

$$P_{rq_i,updated}^{real} = \begin{cases} P_{rq_i,old}^{real} - \bar{P}_{rq_i}^{real} & (RS_{rq} \cap RS_{rq_i} \neq \varnothing) \\ P_{rq_i,old}^{real} & (RS_{rq} \cap RS_{rq_i} = \varnothing) \end{cases}, \quad (7)$$

where $P_{rq_i,old}^{real}$ and $P_{rq_i,updated}^{real}$ represent the travel charges of in-vehicle traveler rq_i before and after the new ride request is added, respectively. Also, the number of remaining seats at these vertices, which now take the travel route of the new-in ride request rq, should be updated by

$$c_{pv,updated}^{w_x} = \begin{cases} c_{pv,old}^{w_x} - 1 & (< w_x, w_{x+1} > \in RS_{rq}) \\ c_{pv,old}^{w_x} & (< w_x, w_{x+1} > \notin RS_{rq}) \end{cases}, \quad (8)$$

where $c_{pv,old}^{w_x}$ and $c_{pv,updated}^{w_x}$ are the numbers of remaining seats of vehicle pv in vertex w_x before and after the new ride request is added, respectively.

Our proposed travel charge model (i.e., Equations (5) $\sim$ (7)) meets the *Charge Balance Rule* (Lemma 1) and *Non-increasing Charge Rule* (Lemma 2).

Lemma 1. Charge Balance Rule: The total travel charge of all travelers in vehicle pv is equal to the travel charge of vehicle pv to complete its travel route list, which can represented as

$$\sum_{pv} P_{rq}^{real} = \sum_{x=1}^{n-1} (D(w_x, w_{x+1})) \cdot Pr. \quad (9)$$

Proof: As stated in Definition 5 and our travel charge model, the detour distance incurred in picking up new-in ride request travelers will be considered as the part of travel route of the new-in ride request. Thus, a route segment $< w_i, w_{i+1} >$ in the route list of vehicle pv must belong to the travel routes of at least one ride request assigned to vehicle pv.

Assuming that route $< w_i, w_{i+1} >$ belongs to the travel route of only one ride request (e.g., rq_1), based on Definition 5, the travel charge of ride request rq_1 can be represented as

$$P_{rq_1}^{real}(< w_i, w_{i+1} >) = D(w_x, w_{x+1}) \cdot Pr. \quad (10)$$

That is, the travel charge of a ride request on route segment $< w_i, w_{i+1} >$ is equal to travel cost. When a new ride request rq_2 is inserted into vehicle pv, the following three scenarios can result: (i) route segment $< w_i, w_{i+1} >$ does not belong to the travel route of ride request rq_2, (ii) route segment $< w_i, w_{i+1} >$ is a part of the route of ride request rq_2, and (iii) the pick-up/drop-off point of ride request rq_2 is inserted in the route segment $< w_i, w_{i+1} >$.

For scenario (i), the travel cost of segment $< w_i, w_{i+1} >$ will be paid only by ride request rq_1. For scenario (ii), the travel cost of segment $< w_i, w_{i+1} >$ will be shared by ride request rq_1 and rq_2. That is

$$P_{rq_1}^{real}(< w_i, w_{i+1} >) + P_{rq_2}^{real}(< w_i, w_{i+1} >) = \frac{D(w_x, w_{x+1})}{2} \cdot 2 \cdot Pr = D(w_x, w_{x+1}) \cdot Pr. \quad (11)$$

For scenario (iii), assuming the pick-up point s_{rq_2} of ride request rq_2 is inserted into route segment $< w_i, w_{i+1} >$, the route segment is then divided into $< w_i, s_{rq_2} >$ and $< s_{rq_2}, w_{i+1} >$. The travel charge of ride request rq_2 and rq_1 can be determined by Equations (5) and (7), respectively, represented as

$$P_{rq_2}^{real}(< w_i, w_{i+1} >) = (D(w_i, s_{rq_2}) + D(s_{rq_2}, w_{i+1}) - D(w_i, w_{i+1}) + \frac{D(w_i, w_{i+1})}{2}) \cdot Pr. \quad (12)$$

$$P_{rq_1}^{real}(< w_i, w_{i+1} >) = \frac{D(w_i, w_{i+1})}{2} \cdot Pr. \quad (13)$$

Then,

$$P_{rq_1}^{real}(< w_i, w_{i+1} >) + P_{rq_2}^{real}(< w_i, w_{i+1} >) = (D(w_i, s_{rq_2}) + D(s_{rq_2}, w_{i+1})) \cdot Pr. \quad (14)$$

As stated in Equation (14), the combined travel charge of ride request rq_1 and rq_2 on road segments $< w_i, s_{rq_2} >$ and $< s_{rq_2}, w_{i+1} >$ will be equal to the product of travel distance and per mile travel cost.

Based on the Mathematical Induction method, Lemma 1 (i.e., Equation (9)) can be proven by adding all route segments in the route list of vehicle pv. ■

Lemma 2. Non-increasing Charge Rule: When a new ride request is inserted into the route list of vehicle pv, the travel charges for in-vehicle travelers will not increase.

Proof: Based on our travel charge model, when a new ride request is added to a vehicle, the updated travel charge of in-vehicle ride requests can be determined by Equations (6) and (7). Because the reduced travel charge of an in-vehicle ride request rq (i.e., Equation (6)) is always larger than zero, the updated travel charge of the in-vehicle ride request is either invariant when no route segment is shared by the in-vehicle ride request and new-in ride request, or reduced when one or more route segments are shared by the in-vehicle ride request and new-in ride request, based on Equation (7). Thus, our travel charge model conforms to the non-increasing charge rule. ■

C. Problem Formalization

As stated above, public vehicle scheduling (i.e., ride matching) in our system can be formalized as an optimization problem with the objective of maximizing the total satisfaction of travelers. Note that, when our system determines whether a new ride request can be matched to a vehicle, some constraints should be guaranteed, which are listed as follows: (i) one ride request can only be served by one vehicle, (ii) the number of ride requests that are served simultaneously by a public vehicle should not be larger than a pre-defined value (i.e., the pre-defined service capacity of a vehicle), (iii) the number of in-vehicle travelers should not exceed the capacity of a vehicle, (iv) a new-in ride request cannot be matched to a vehicle when the sharing level of the new-in ride request or the vehicle is no-share', (v) a ride request can be matched to a vehicle if and only if they belong to the same sharing level, and (vi) if and only if travelers' travel satisfactions in sharing are larger than either the current satisfactions or pre-defined threshold, travelers will share rides (i.e., the new ride request is allowed to be inserted in the traveling route of this vehicle). With this limitation, the scenario that a traveler stays in a vehicle for a long time due to the insertion of new ride requests will not occur.

Based on above statements, in a time slot t, assuming that m public vehicles are in the transportation system and n ride requests are launched, the optimization problem can be represented as

$$\mathbf{x} = \arg\max_{\mathbf{x}} \left\{ \sum_{rq_i \in RQ_{in} \cap RQ_{new}} \left(SAT_{rq_i} \cdot \sum_{j=1}^{m} x_{rq_i j} \right) \right\} \quad (15)$$

Subject to

$$\sum_{j=1}^{m} x_{rq_i j} \begin{cases} = 1 \ (\forall rq_i \in RQ_{in}) \\ \leq 1 \ (\forall rq_i \in RQ_{new}) \end{cases} \quad (16)$$

$$x_{rq_i j} = \begin{cases} 1 \ (\forall rq_i \in RE_{pv_j}) \\ 0 \ (\forall rq_i \notin RE_{pv_j}) \end{cases} \quad (17)$$

$$SL_{rq_i} = \begin{cases} 1 \text{ (All-Share)} \\ 2 \text{ (Female-Only)} \\ 5 \text{ (No-Share)} \end{cases} \quad (18)$$

$$SL_{pv_j} = \begin{cases} SL_{rq_1} \ (pv_j \text{is busy}) \\ -1 \quad (pv_j \text{is idle}) \end{cases} \quad (19)$$

$$x_{rq_i j} \cdot Ind(s_{rq_i}, RL_{pv_j}) \leq x_{rq_i j} \cdot Ind(d_{rq_i}, RL_{pv_j}) \quad (20)$$

$$\forall rq_i \in RQ_{new},$$

$$SAT_{rq_i} > min(SAT_{rq_i}^{Old}, SAT^{threshold}) \quad (21)$$

$$\forall rq_i \in RQ_{new},$$

$$\begin{cases} x_{rq_i j} \cdot (N_{RE_{pv_j}} + 1) \leq N_{RE}^{max} \\ x_{rq_i j} \cdot (c_{pv_j}^{s_{rq_i}} + Num_{rq_i}) \leq C_{pv_j} \\ x_{rq_i j} \cdot (SL_{pv_j} + SL_{rq_i}) \leq 5 \\ \min(x_{rq_i j} \cdot SL_{pv_j}, x_{rq_i j} \cdot \|SL_{pv_j} - SL_{rq_i}\|) \leq 0 \end{cases} \quad (22)$$

$$\text{Equation } (1) \sim (14), \quad (23)$$

where RQ_{in} and RQ_{new} are the set of in-vehicle and new-in ride requests, respectively, RE_{pv_j} is the set of ride requests that are being served or will be served by vehicle pv_j, N_{RE}^{max} is the maximum number of ride requests that can be served simultaneously by a public vehicle, and $Ind(s_{rq_i}, RL_{pv_j})$ and $Ind(d_{rq_i}, RL_{pv_j})$ represent the sequence numbers of s_{rq_i} and d_{rq_i} in the route list of vehicle pv_j, respectively, and $SAT_{rq_i}^{Old}$ represents the satisfaction of ride request rq_i at last time slot, and $SAT^{threshold}$ is the pre-defined satisfaction threshold.

D. Discussion of α, β and γ

As described in Equation (1), α, β and γ are introduced as the preference parameters to affect the rider-vehicle matchings of travelers. The sum of α, β and γ is 1, and each represents the ratio against total travel satisfaction. Note that the weight can be chosen according to travelers' personal preferences. We now consider a case of selecting weights when travelers have an acceptable detour time/distance ratio. Here, we use an example of the acceptable detour time/distance ratio being 1.5 (i.e., the real travel time and distance is 1.5 times the expected travel time and the distance) to show how to select the weight of preference parameters α, β and γ.

We assume that when the acceptable detour time/distance ratio is achieved, and the real travel charge is minimized (i.e., sharing is maximized), the satisfaction of the traveler will be equal to the expected satisfaction of the original route without ridesharing (i.e., complete trip alone with expected time, distance, and charge), that is

$$T^{real} = \frac{3}{2} \cdot T^{exp}, D^{real} = \frac{3}{2} \cdot D^{exp}, P^{real} = \frac{P^{exp}}{C_{pv}}, \quad (24)$$

where C_{pv} is the capacity of vehicles. Substituting Equation (24) into Equation (1) and (2), we obtain

$$SAT_{rq} = e^{\cdot\left(\frac{(C_{pv}-1)\cdot\gamma}{C_{pv}} - \frac{\alpha}{2} - \frac{\beta}{2}\right)}. \quad (25)$$

Typically, the detour time increases proportionally with detour distance, thus the travel distance and travel time can have the same weight in satisfaction determination. That is, the parameter α and β can be equivalent in Equation (1). Thus,

$$\frac{(C_{pv}-1)\cdot\gamma}{C_{pv}} - \alpha = 0 \quad \text{or} \quad \frac{(\mathrm{C_{pv}}-1)\cdot\gamma}{\mathrm{C_{pv}}} - \beta = 0. \quad (26)$$

Based on Equation (1), α, β, and γ can be selected, represented as

$$\alpha = \frac{C_{pv}-1}{3\cdot C_{pv}-2}, \beta = \frac{C_{pv}-1}{3\cdot C_{pv}-2}, \gamma = \frac{C_{pv}}{3\cdot C_{pv}-2}. \quad (27)$$

For example, if each public vehicle has 5 seats (i.e., $C_{pv} = 5$), an appropriate selection of (α, β, γ) can be $(4/13, 4/13, 5/13)$. Note that the above statement considers the possible selection of parameter (α, β, γ) to stimulate travelers to share rides, as well as balance the impacts of travel charge and travel time/distance on traveler satisfaction.

IV. Edge Computing Based Public Vehicle System

In our ECPV system, an *edge computing based ride request transmission mechanism* and a *tree based heuristic matching mechanism* are jointly proposed to transmit and respond to ride requests launched by travelers and select appropriate vehicles to service ride requests (i.e., ride matching), achieving an effective matching of ride-traveler pairs with the objective of improving traveler satisfaction and traffic efficiency and reducing the delay of ride-traveler pairs matching. Further, a *graph partition based depot placement mechanism* is proposed to determine locations for idle vehicle parking for the purpose of balancing vehicle distribution across transportation system.

We consider that the optimization problem is solved periodically. In each time slot, an edge device collects all ride requests and vehicle information in its area and matches all received ride requests to appropriate vehicles. The time slot can be a small period (e.g., 5 s or 10 s). Due to the integration with edge computing and edge devices, most ride requests can be matched in edge devices with acceptable response latency, even if a batch of requests is received.

A. Edge Computing Based Ride Request Transmission Mechanism

Due to the distributed architecture of edge computing and the advantage of edge devices being close to end-users [16]–[18], the computation resources of edge devices can be used by end-users to complete computation tasks with low latency. If all ride requests are directly delivered to the cloud and all matchings of ride requests and public vehicles are computed in the data center, a high response latency would occur for ride requests for travelers, even if a large amount of computation resources are provisioned in the data center, and thus real-time service response would be impossible due to high link congestion on the network, especially under a large number of ride requests launched simultaneously. Thus, the function of edge computing in our scheme is to reduce service response latency, which is necessary to provide real-time service response. The edge devices in our scheme complete rider-vehicle matching in their regions and locally respond to ride requests with low latency, thereby resulting in higher quality ridesharing service and improved traveler quality of experience, as well as reducing the latency of service responses to the ride requests of travelers. The function of the data center in our scheme is to collect, store and analyze global traffic information in the smart transportation system and assist edge devices to complete cross-region rider-vehicle matching.

In our proposed ECRT mechanism, a three-tier network architecture, including vehicular network, edge network, and cloud, is considered, as shown in Fig. 1. With consideration for the mobility of vehicles, when a vehicle enters the area of an edge device, the vehicle should send a check-in message to this edge device to declare that the vehicle can be locally scheduled by that edge device. When the vehicle exits the area of this edge device, the vehicle sends a check-out message. Our model assumes that the edge network can cover the entire target area. If a vehicle is in an area with multiple edge devices, the vehicle sends its check-in message to the closest edge device.

Note that a large number of research efforts have been devoted toward developing and improving vehicular networks, which can be directly integrated into our system with great performance for data transmission [19]–[22]. In this paper, we instead focus on an emergent application based upon vehicular network technologies.

Travelers can launch ride requests from their mobile devices and requests can be delivered to the local edge devices via wireless networks. The ride request can be represented as $RideRequest = < rq, t_c, t_{pick}, Num_{rq}, s_{rq}, d_{rq}, SL_{rq} >$, where t_c is the current time slot and t_{pick} is the time to pick

the traveler up. When the edge device, denoted as ed, receives a ride request, it starts the local vehicle scheduling to find an appropriate vehicle for this ride request. If an appropriate vehicle can be found in the coverage area of this edge device, the request response will be sent back to travelers and the selected vehicle, which can be represented as $RideResponse = < ed, pv, t_{arrive}, SL_{pv}, RL_{pv} >$, where t_{arrive} is the time that vehicle pv will arrive at the pick-up point of the ride request rq.

If no appropriate vehicle can be assigned to the ride request in the area of edge device ed, the edge device will send the ride request to its neighboring edge devices to find an appropriate vehicle for the fcomplete trip alone with expected as

$$ed \rightarrow ed_n : (ed, RideRequest) \tag{28}$$

$$ed_n \rightarrow ed : (ed_n, RideResponse), \tag{29}$$

where ed_n is the neighboring edge device of edge device ed.

If all neighboring edge devices cannot select an appropriate vehicle for the ride request, edge device ed will directly send the ride request to cloud to choose an appropriate vehicle globally. This procedure can be represented as

$$ed \rightarrow Cloud : (ed, RideRequest) \tag{30}$$

$$Cloud \rightarrow ed : \begin{cases} (Cloud, RideResponse) \\ (Cloud, Null) \end{cases}, \tag{31}$$

where $(Cloud, Null)$ indicates that there are no appropriate vehicles for the ride request at this time slot. In this case, the ride request will be deferred to the next time slot for ride matching.

In our system, a new ride request will be first served by its closest edge devices. If there is not an available vehicle, the new ride request will be served by the neighboring edge devices or by the data center. Hence, although in our system each vehicle is assigned to a single edge device, the vehicle can also be matched to ride requests of other edge devices through communication among edge devices, thereby ensuring the efficiency of vehicle-rider pair matching in our system. Assigning each vehicle to a single edge device in each time increment can ensure that, at any time, a vehicle can only communicate with one edge device for rider-vehicle matching, thereby avoiding the scenario in which a ride request is served by multiple edge devices and assigned to different vehicles, while also avoiding the scenario in which a vehicle is matched to multiple ride requests at the same time.

In reality, most ride requests will be matched to vehicles which are close to where the ride requests are launched from, because in this case, the waiting time of travelers will not be long and the probability of successful matching ride requests with vehicles will be high. Thus, our edge computing based ride request transmission mechanism can complete most ride matching in the edge devices, thereby reducing the computation load of the cloud and the ride request latency and achieving real time services to travelers. Note that, since vehicles send check-in messages to the edge device once they enter the area of this edge device, the edge device can always have the latest information of vehicles in its area, and based on such formation, the edge device can locally match most of travelers to vehicles in its area. Even if an edge device may need the aid of its neighboring edge devices (Equations (28) and (29)), the latest information of vehicles will be shared by edge devices to their neighbors within a local area while other edge devices may not need to have such information. Thus, the information inconsistency of our ECPV system in a local area can be avoided, and it may not be necessary to be taken care in the whole transportation system as a significant portion of matching vehicle-rider pairs will be performed at edge nodes locally.

Due to processing rider-vehicle matching locally, the edge devices in our system only need their local traffic information and do not need to obtain global traffic information from the data center. The only information needed from data center is the matching results of ride requests that cannot be successfully completed via local matching, requiring global matching in the data center instead. Assuming that the edge devices connect to the data center with 1 Gbps networks (i.e., the transmission speed is 128 MBs), the delivery of one ride request and receipt of the response between an edge device and the data center requires 64·2 B data, and the communication cost between edge-device and data center is 972 μs in ideal communication conditions. This communication cost is much larger than the time cost of computation, as shown in Table III of following section.

As mentioned, in our system, most ride requests are matched locally and only a small number of ride requests will be sent to the data center for global matching. Thus, even if the edge devices cannot obtain global matching results in time, the impact on our system is only marginal.

B. Tree Based Heuristic Matching Mechanism

Due to the global optimal matching of ride requests and vehicles cannot be achieved in an accepted delay, in this section, a tree based heuristic matching (THM) mechanism is proposed to achieve locally effective matching of ride-traveler pairs with the objective of improving traveler satisfaction and traffic efficiency, as well as reducing the delay of ride request response. In our THM mechanism, each vehicle creates and maintains an insertion tree to show the vertices that vehicles can pass through, and edge devices can conduct ride matching based on these insertion trees and return the real time service response to travelers.

1) Insertion Tree Creation: The travel route of a vehicle, denoted as $< w_0, w_1, \ldots, w_m >$ can be alternatively represented as a list of vertices in the road network, denoted as $< v_0, v_1, \ldots, v_n >$, where n should be not equal to m, because a long road must be divided into several road segments to ensure communication areas of all edge devices cover the entire road network. Fig. 2(a) illustrates an example of a road network, where each vertex represents an intersection (i.e., $< v_0, v_1, \ldots, v_n >$) and each link represents a road.

When a ride request is assigned to a vehicle, the travel route of this vehicle can be created and represented as $< v_0, v_1, v_2 >$, where v_0, v_1, and v_2 represent the current location, pick-up point, and drop-off point, respectively. As shown in Fig. 2-(b). Then, the vehicle can create an insertion tree that is organized by vertices, to show which vertices the vehicle can pass through without reducing the satisfaction of in-vehicle travelers. Fig. 2-(c) shows the insertion-tree of the vehicle based on the current traveling route shown in Fig. 2-(b). In Fig. 2-(c), the root node

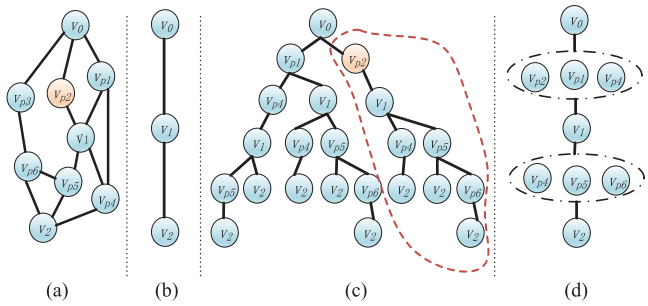


Fig. 2. Road network and insertion-trees.

is the current vertex, corresponding to v_0, and the leaf node is the final destination vertex, corresponding to v_2, and each branch of this tree should include vertex v_1. That means traveling from v_0 to v_2 must pass through v_1. Other branch nodes in the insertion-tree (denoted as v_{p1}, v_{p2}, v_{p4}, v_{p5}, and v_{p6} in Fig. 2-(c)) are called detour vertexes, representing vertexes that can be passed through during the trip from v_0 to v_2.

If the pick up point and drop off point of a new ride request are both detour vertexes of the insertion-tree and included in one branch, and the pick up point is ancestor node of drop off point, the new ride request can be considered to be inserted into the traveling route of this vehicle. Note that, a new ride request in which only the pick up point is the detour vertexes of the insertion-tree can also be considered to be inserted into the traveling route of this vehicle, and in this scenario the drop off point of this new ride request only can be inserted into the end of traveling route list (i.e., inserted as end-nodes in each branch of insertion-tree).

Obviously, the selection of detour vertices is essential for creating the insertion tree. In our scheme, the vehicle will select detour vertices of their travel route in a loose way. That is, these detour vertices are selected by considering only the increase in detour time and distance, while the resulting detour travel charge and impact on satisfaction are not considered in this step, but instead will be checked by edge devices during the matching of ride requests to vehicles. Travelers can choose their personal detour time/distance ratio (i.e., α and β in Equation 1) to determine the detour vertex. In this paper, we take the detour time/distance ratio to be 1.5 simply as an example to state the basic idea and process of our system.

Obviously, a vertex can be inserted into different positions in the insertion tree, because it may be successfully inserted into different route segments of a travel route. As shown in Fig. 2-(c), v_{p4} becomes a different branch node depending on the insertion tree and route that are inserted, such as $< v_0, v_1 >$ and $< v_1, v_2 >$. In addition, to avoid circles in travel routes, loops are not allowed in any branch of the insertion tree.

As the vehicle drives along the travel route, the insertion tree of the vehicle should be updated, and the useless subtrees of the tree should be removed. As shown in Fig. 2-(c), when a vehicle moves to v_{p2}, only the subtree with v_{p2} as the root node can provide realizable travel routes (i.e., subtree surrounded by a red dotted line in Fig. 2-(c)), and other subtrees will be useless. During the trip, the insertion-tree of a vehicle will be constantly updated, due to the useless subtree is stripped and new nodes (i.e., pick up points and drop off points of new ride requests) are added in insertion-tree. Finally, when the vehicle completes all transportation services, it will clear the insertion tree and become an idle vehicle to wait for new ride requests.

In some situations, the travel route of a vehicle is quite long and the insertion tree will be very large, because many detour vertexes can be successfully inserted into a number of route segments of the large route, resulting in the insertion tree being difficult or impossible to store and deliver. To address this issue, the insertion tree can be simplified as a tree with a single branch, and the branch nodes in the simplified tree are the sets of detour vertexs that can be successfully inserted into the corresponding route segments. For example, Fig. 2-(d) has two branch nodes and each of them is a set of detour vertexes (e.g., the first branch node includes vertex v_{p2}, v_{p1}, v_{p4}, and the second branch node includes v_{p4}, v_{p5}, v_{p6}). Each vertex in these two-branch nodes represents that this vertex can be successfully inserted into the route $< v0, v1 >$ or $< v1, v2 >$.

Although the simplified insertion tree reduces the storage requirements in vehicles and is easier to create, it will increase the computation of ride-vehicle matching in edge devices. The reason is that the traveling order of detour vertices that a vehicle can pass through is clearly revealed in the intact insertion tree, but is not included in the simplified insertion tree, resulting in extra computation in edge devices to determine the possible traveling order of detour vertices.

2) Matching Between Ride Requests and Vehicles: Once a public vehicle enters the communication coverage area of an edge device, the insertion tree of the vehicle is sent to the edge device. When the edge device receives a ride request, it first checks whether there are vehicles within this area, whose sharing level is the same as the sharing level of the request. If such vehicles exist, they are considered as candidates, and the edge device detects their insertion trees in turn, finds the branches, considers whether both pick-up vertex and drop-off vertex of the ride request are included, and further assesses that the pick-up vertex is the ancestor node of drop-off vertex. Meanwhile, the number of travelers in any vertex cannot exceed the capacity of the vehicle. This means that there should be enough seats left for the new ride request between the pick-up and drop-off vertices. If no eligible branch exists in any insertion trees (i.e., no vehicle can share a ride with this ride request), the ride request will be sent to neighbor edge devices, and eventually the cloud, to determine ride matching in the same manner.

In some situations, multiple eligible branches will exist in different insertion trees (i.e., multiple eligible vehicles in this area can share a ride with this ride request). More likely still, multiple eligible branches will exist in a single insertion tree, meaning that the pick-up/drop-off vertices can be inserted into different positions (i.e., route segments) of the travel route of a vehicle. In this case, edge device needs to determine both which eligible vehicle should be selected and which position of the route should be selected to insert the pick-up and drop-off vertices of the new ride request.

Algorithm 1: Rider-Vehicle Matching.

Input: Local public vehicles $(pv_1, \ldots, pv_n)$, corresponding insertion-trees $(TI_{pv_1}, \ldots, TI_{pv_n})$, and ride request (rq_{new}).

Output: Eligible vehicle (pv_e) and eligible insertion route segment $(< w_i, w_{i+1} >, < w_{j-1}, w_j >)$.

1: $S_{SVBM} = 0, S_{LVBM} = 0$;
2: **for** All local vehicles $\{pv_l\}$ **do**
3: **if** $s_{rq_{new}}, d_{rq_{new}} \in B[TI_{pv_l}]$ & $Ind(s_{rq_{new}}, B[TI_{pv_l}]) < Ind(d_{rq_{new}}, B[TI_{pv_l}])$ && $\forall v \in \{B[TI_{pv_l}](s_{rq_{new}} \uparrow), \ldots, B[TI_{pv_l}](d_{rq_{new}} \downarrow)\}, c^v_{pv} \leq C_{pv}$ **then**
4: $pv_l \rightarrow \boldsymbol{E_{pv}}$;
5: **end if**
6: **end for**
7: **for** $\forall pv_x \in \boldsymbol{E_{pv}}$ **do**
8: **for** $\forall(< w_a, w_{a+1} >, < w_{b-1}, w_b >) \in TI_{pv_x}$ **do**
9: **if** $\forall rq_{in} \in \boldsymbol{RQ_{in,pv_x}}, SAT_{rq_{in}} \geq SAT^0$ **then**
10: $(pv_e, < w_i, w_{i+1} >, < w_{j-1}, w_j >) = (pv_x, < w_a, w_{a+1} >, < w_{b-1}, w_b >)$; break(SFM);
11: **if** $S_{SVBM} < (\sum_{rq_{in} \in RQ_{in,pv_x}} SAT_{rq_{in}} + SAT_{rq_{new}})$ **then**
12: $S_{SVBM} = (\sum_{rq_{in} \in RQ_{in,pv_x}} SAT_{rq_{in}} + SAT_{rq_{new}}$;
13: $(pv_e, < w_i, w_{i+1} >, < w_{j-1}, w_j >) = (pv_x, < w_a, w_{a+1} >, < w_{b-1}, w_b >)$;
14: **end if**
15: **end if**
16: **end for**
17: **if** $S_{LVBM} < S_{SVBM}$ **then**
18: $S_{LVBM} = S_{SVBM}$;
19: $(pv'_e, < w'_i, w'_{i+1} >, < w'_{j-1}, w'_j >) = (pv_e, < w_i, w_{i+1} >, < w_{j-1}, w_j >)$;
20: **end if**
21: **end for**
22: $(pv_e, < w_i, w_{i+1} >, < w_{j-1}, w_j >) = (pv'_e, < w'_i, w'_{i+1} >, < w'_{j-1}, w'_j >)$;

In our ECPV system, the following ride matching strategies are considered: (i) *Single-First-Match (SFM)*: It matches the ride request to the first eligible vehicle and inserts pick-up and drop-off vertices in the first eligible positions. (ii) *Single-Vehicle-Best-Match (SVBM)*: It matches the ride request to the first eligible vehicle and inserts pick-up and drop-off vertices in eligible positions that achieve the maximum satisfaction of all travelers in this vehicle. (iii) *Local-Vehicle-Best-Match (LVBM)*: It matches the ride request to the best eligible vehicle and inserts pick-up and drop-off vertices in eligible positions that achieve the greatest satisfaction of all travelers in this vehicle in comparison with other eligible vehicles in this area.

The rider-vehicle matching in our ECPV system is shown in Algorithm 1, where $\boldsymbol{E_{pv}}$ represents the set of eligible vehicles in the local area covered by an edge device, $\boldsymbol{RQ_{in,pv_x}}$ represents all ride requests in vehicle pv, $B[TI_{pv_l}]$ represents the branch of insertion-tree TI_{pv_l}, and $B[TI_{pv_l}](s_{rq_{new}} \uparrow)$ and $B[TI_{pv_l}](d_{rq_{new}} \downarrow)$ represent the parent node of $s_{rq_{new}}$ in $B[TI_{pv_l}]$ and child node of $d_{rq_{new}}$ in $B[TI_{pv_l}]$, respectively. In Algorithm 1, the eligible vehicle set is first determined by the nearby edge device (Lines 2∼7), and then the new ride request can be arranged to an eligible vehicle by one of the aforementioned ride matching methods, including SFM (Lines 7∼10 and 22), SVBM (Lines 7∼9 and 11∼13 and 22), LVBM (Lines 7∼9, 11∼12, 17∼22). In Algorithm 1, although all mentioned ride matching methods are described, only one of them needs to be implemented in edge devices when the ECPV system is deployed.

As shown in Algorithm 1, it is obvious that the computation complexities of SFM, SVBM and LVBM are $O(MN)$, $O(MN^2)$ and $O(M^2N^2)$, respectively, where M is the average number of vehicles in a local region and N is the average number of requests in a vehicle route. Obviously, SFM achieves the lowest computation complexity, and LVBM achieves the highest computation complexity. SFM is a first-well algorithm, i.e., it matches a ride request to the first-found eligible vehicle and inserts pick-up and drop-off vertexes in the first-found eligible positions in the route list of the first-found eligible vehicle. LVBM is 'local-well' algorithms. Note that, 'local-well' means that, when an edge-device receives a new ride request, the edge-devices matches the new ride request to the most eligible vehicle in its local area (i.e., all vehicles checked into this edge device at this time slot) and insert pick-up and drop-off vertexes in the most eligible positions in the route list of the matched eligible vehicle. Thus, SFM may achieve lower optimality performance, while LVBM may achieve higher optimality performance.

Also, most rider-vehicle matchings are conducted in edge devices locally, and thereby the globally optimal matching cannot be achieved in our system. However, locally effective rider-vehicle matching also has many benefits, such as increasing the travel satisfaction of travelers, improving the vehicle occupancy ratio (i.e., ridesharing efficiency), reducing traffic congestion, improving traffic efficiency, achieving low ride matching response latency, and reducing computational tasks in the cloud and congestion on the cloud network. Our ECPV system can also integrate with dynamic route schemes to alter traveler routes between two consecutive vertices in route list (i.e., a route segment) with real time traffic conditions, thereby reducing the traveling time of travelers.

Our system can integrate with dynamic route guidance schemes to adapt to scenarios with traffic congestion. Via introducing dynamic route guidance (e.g., DEDR [23]), the route can be altered during travel according to traffic states, and the travel time can also be accurately estimated even if traffic congestion occurs. However, traffic congestion at peak travel time hours will certainly cause decreased traveler satisfaction, which is simply an aspect of bad traffic and not an aspect of inaccurate travel time estimation.

C. Graph Partition Based Depot Placement Mechanism

Traffic efficiency is directly related to the number of vehicles driving on roads. When a large number of vehicles, especially

idle vehicles without passengers, drive on roads, traffic congestion is greater and traffic efficiency is low. Thus, to improve traffic efficiency, idle vehicles that do not have any ride requests or have completed their service should drive to depots to park and wait for new ride requests. To reduce and balance the overall waiting time of travelers (i.e., the time of idle vehicle from depots to pick-up points of travelers), the suitable depot locations should be determined.

To address this issue, a graph partition based depot placement mechanism is proposed to select suitable depot locations. In our mechanism, the area of service is partitioned into several regions and a depot is placed in each region. The number of partitions (i.e., the number of depots in the whole area) depends on the number of public vehicles in the transportation system and the capacity of each depot, i.e., $Num_{depot} = \frac{Num_{pv}}{C_{depot}}$, where Num_{depot} and Num_{pv} represent the number of depots and vehicles in the system, respectively, and C_{depot} is the capacity of a depot. The capacity of each depot can be distinct. However, to simplify our evaluation, we assume the capacity of all depots to be the same.

The basic idea of the graph partition based depot placement mechanism is to partition the road network $G =< V, E >$ into Num_{depot} regions and select one location (i.e., one vertex in road network G) in each region to place a depot. The objective is to minimize the total travel time from all vertices to the nearest depot. Thus, our depot placement mechanism can be formalized as an optimization problem, represented as

$$\mathbf{x} = \mathbf{arg}\min_{\mathbf{x}} \{TotalWeight\} \tag{32}$$

$$\textbf{Subject to}$$

$$\forall v_i \in |V|, \quad \sum_{j=1}^{|V|} x_{v_i v_j} = 1 \tag{33}$$

$$x_{v_i v_j} = \begin{cases} 1 \ (v_i \in P[v_j]) \\ 0 \ (v_i \notin P[v_j]) \end{cases} \tag{34}$$

$$\sum_{j=1}^{|V|} \left(\min \left(1, \sum_{i=1}^{|V|} x_{v_i v_j} \right) \right) = Num_{depot} \tag{35}$$

$$TotalWeight = \sum_{j=1}^{|V|} \sum_{i=1}^{|V|} \left(x_{v_i v_j} \cdot T^{exp}(v_i, v_j) \right), \tag{36}$$

where $TotalWeight$ is the total shortest travel time of all locations to their nearest depots, $P[v_j]$ is the partitioned region with v_j as the depot location, and $T^{exp}(v_i, v_j)$ represents the shortest travel time from v_i to v_j in road network G.

Equation (32) is designed to minimize the total weight. In this paper, the shortest travel time is selected as the metric to represent the total weight (shown in Equation (36)). Note that other metrics, such as distance or distribution of ride requests, could be used to represent total weight in our model. Equations (33) and (34) represent that each vertex in the road network can only belong to one partitioned region, where $x_{v_i v_j}$ is a binary parameter. Equation (35) shows that the total number of partitioned regions is Num_{depot}. Based on Equation (32) $\sim$ (36), matrix $\mathbf{x}$ can be obtained with respect to $x_{v_i v_j}$, in which $x_{v_i v_j} = 1$ indicates that vertices v_i and v_j are in the same region, and for any vertex v_j, if $\sum_{i=1}^{|V|} x_{v_i v_j}$ is larger than zero, vertex v_j is selected as the location to place the depot. Note that the locations for depot placement are determined prior to the deployment of the entire public vehicle system and the selected locations will not be altered over time. As the placement of depots does not need to be determined in real time, our mechanism does not take much effort to reduce the computation complexity of this optimization problem.

TABLE II
COMPUTATION OVERHEAD

	SFM	SVBM	LVBM
Edge	$O(MN)$	$O(MN^2)$	$O(M^2N^2)$
Centralized	$O(L^2MN)$	$O(L^3MN^2)$	$O(L^4M^2N^2)$

D. Computation Overhead and Latency Analysis

Based on the algorithm workflow in Algorithm 1, the computation complexities of SFM, SVBM, and LVBM with edge and centralized approaches are listed in Table II, where L is the number of partitioned regions in the entire transportation system (i.e., L edge devices exist in the whole system), M is the average number of vehicles in a local region, and N is the average number of requests in a vehicle route.

We assume that edge devices connect to the data center with 1 Gbps networks (i.e., the transmission speed is 128 MB/s), and the CPUs in edge devices and the data center are 1.7 GHz and 3.1 GHz, respectively. Each edge device only has one core, while the data center has 16 cores. As edge devices are close to vehicles, the transmission latency between vehicles and edge devices is ignored. In this scenario, transmitting 1 B data from an edge device to the data center requires 7.6 μs (i.e., $1/(128 * 1024) = 7.6\ \mu s$). Also, typically computation speed can be represented as MIPS (i.e., millions of instructions per second), and 1 MIPS requires a 4 MHz CPU. In this way, processing one instruction in an edge device with a 1.7 GHz CPU requires $2.4 * 10^{-3}$ μs (i.e., $4/(1700 * 1000000) = 2.4 * 10^{-3}\ \mu s$), while a data center with 16 core of a 3.1 GHz CPU requires $8.1 * 10^{-5}$ μs (i.e., $4/(16 * 3100 * 1000000) = 8.1 * 10^{-5}\ \mu s$). Obviously, the transmission latency is much larger than the computation latency. Note that, the time refers to the average speed of processing one instruction in either the edge device or data center.

As an example, we consider that a smart transportation system with 45 regions, each region has 3 vehicles and 3 ride requests, and the edge devices connect to the data center with 1 Gbps networks (i.e., the transmission speed is 128 MB/s), and the delivery of one ride request and receipt of the response between an edge device and the data center requires 64·2 B data, and then the communication cost between edge-devices and data center is 972 μs in ideal communication conditions. This communication cost is much larger than the time cost of computation, resulting in larger response latency, as shown in Table III. The time costs shown in the table are derived through theoretical analysis

TABLE III
COMPUTATION AND COMMUNCIATION LATENCY (μs)

	SFM			SVBM			LTBM		
	computation	communication	total	computation	communication	total	computation	communication	total
Edge	0.022	–	0.022	0.065	–	0.065	0.194	–	0.194
Centralized	1.476	972.8	974.276	199.290	972.8	1172.09	26904.200	972.8	27877.000

Fig. 3. Partial traffic map of Xi'an, China.

under ideal communication conditions. It is primarily our intent to demonstrate that the time costs in our edge based schemes are much lower than those in the centralized scheme. Due to the unstable nature of transmission in realistic communication conditions, the centralized scheme will incur even worse latency.

As the focus of our paper is to investigate ridesharing in public vehicle systems, the computation offloading problem is not our focus. We note that existing offloading schemes, such as computation offloading scheme in edge computing [24]–[28], could be used in our system to achieve the trade-off of computation and communication between edge-devices. In future work, we shall conduct an in-depth investigation of the trade-off between computation and communication, and consider mechanisms to achieve this.

V. PERFORMANCE EVALUATION

A. Evaluation Setup

We conducted evaluations based on a simplified version of a real traffic map [23], as shown in Fig. 3, in which a road includes two lanes, each in one direction, to simplify our evaluations. Note that, although our system can integrate with dynamic route guidance schemes [23], [29]–[32] to determine travel routes in congested traffic scenarios, traffic congestion is not considered here.

We evaluate our depot strategy (namely GDPD) in comparison with the random depot strategy. In the random depot strategy, the depot locations are randomly selected from the map. The evaluation results of the random depot strategy represent the average value of 100 attempts, where the pick-up and drop-off locations are randomly selected from the map in each attempt. Also, the pre-defined satisfaction threshold for each traveler is 75 %, and the travel charge (i.e., cost) per mile is \$2.

In our evaluations, three scenarios are considered: (i) 3 vehicles and 18 requests, (ii) 10 vehicles and 64 requests, and (iii) 50 vehicles and 533 requests, which are evaluated in a transportation system. Additionally, we evaluate performance with two distributions of pick-up and drop-off positions over the city map: uniform distribution and non-uniform distribution. The former is to randomly choose the pick-up and drop-off positions over the city map for each ride request with the same probability, while the latter is to choose the pick-up and drop-off positions over the city map for each ride request with different probabilities (e.g., when going to work, travelers would choose workplaces and homes as the pick-up location and drop-off location with higher probability).

Also, in our simulation, each vehicle will maintain a route path which is a sequence of locations (i.e., the sequence of pick-up locations and drop-off locations of the travelers in the vehicle). When a ride request is matched to a vehicle, there could be two route paths for the vehicle: the old route path (i.e., the location sequence before the pick-up location and drop-off location of the new ride request have been inserted) and the new route path (i.e., the location sequence after the pick-up location and drop-off location of the new ride request have been inserted). Based on these two route paths, the traveling time, traveling distance and traveling charge of the in-vehicle travelers can be determined by our model, and then the current traveling satisfaction of the in-vehicle travelers (i.e., the satisfaction with the old route path) and the in-share traveling satisfaction of the in-vehicle travelers (i.e., the satisfaction with the new route path) can be assessed by our satisfaction model. Note that edge devices are used to complete local matching and the data center is used to complete global matching.

To measure the effectiveness of our ECPV system, the following metrics are considered: (i) *Travel satisfaction* is defined as the satisfaction of travelers on their trips, as shown in Equation (1), (ii) *Travel time* is defined as the time to complete the trip from ride request launched to destination reached, (iii) *Travel distance* is defined as the total distance traveled to complete the trip from the pick-up points to drop-off points, (iv) *Travel charge* is defined as the payment incurred by travelers for their trips, and (v) *Vehicle occupancy ratio* is defined as the average number of travelers in vehicles at each time slot. Also, the three proposed ride matching strategies (i.e., SFM, SVBM, LVBM) in Section IV-B2 and two existing ridesharing methods, namely T-Share [8] and PTRider [14], are evaluated in our ECPV system, and are compared with a non-sharing system (i.e., no ride requests can be shared). All evaluations in this paper were completed in Matlab 2014a. In our evaluations, computers serving as edge devices were provisioned with a 1.7 GHz CPU and 4 GB RAM. Note that, in our evaluation figures, VN and RN represent the number of vehicles and ride requests, respectively.

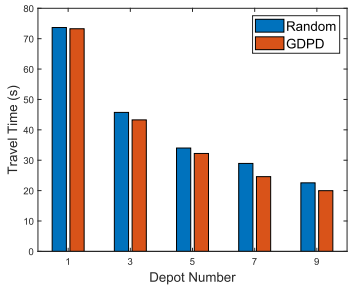


Fig. 4. Vehicle placement selection.

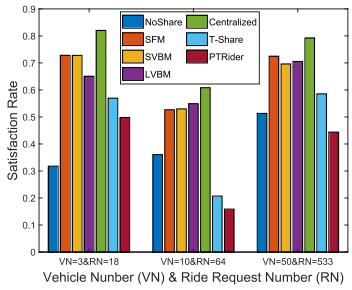


Fig. 5. Average satisfaction of ride requests.

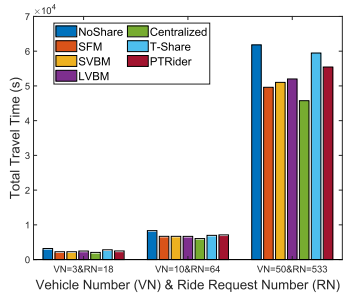


Fig. 6. Total travel time.

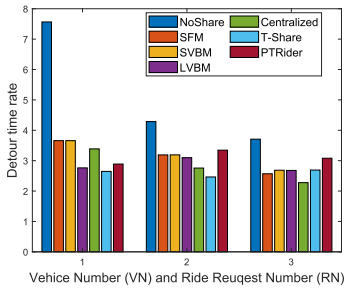


Fig. 7. Detour time rate.

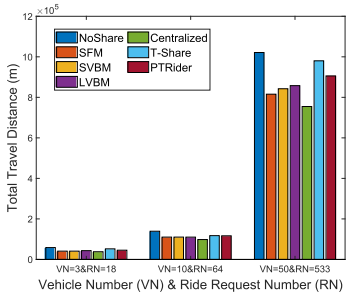


Fig. 8. Total travel distance.

B. Evaluation Results

Depot Placement: Fig. 4 shows the traveling time from the depot location to the pick-up location for various numbers of depots. As shown in Fig. 4, our GDPD strategy always achieves lower traveling time in comparison with the random depot strategy. This means that, using the depot locations in our strategy, travelers will have less waiting time. Additionally, as the number of depots increases, the traveling time decreases. Note that the evaluation results in the random depot strategy were averaged over 100 attempts, in which the pick-up and drop-off locations are randomly selected from the map in each attempt. The results in Fig. 4 demonstrate that our depot strategy is capable of selecting more effective depot locations.

Traveling Satisfaction: Fig. 5 illustrates the average travel satisfactions of all ride requests under different ride-matching strategies in various scenarios. As we can see from the figure, in comparison with the non-sharing scenario, existing sharing schemes (i.e., T-Share and PTRider), and the centralized scheme, all ride matching strategies proposed in our ECPV system (i.e., SFM, SVBM, LVBM) can achieve greater travel satisfaction for riders. Here, the centralized scheme indicates that all vehicle-rider pairs are matched in the data center so that global optimal matching between vehicles and ride requests can be achieved. Although the centralized scheme can achieve better satisfaction than our edge based schemes, the computation complexities and response latency in the centralized scheme are much higher, as shown in Table II. The non-sharing scenario achieves the lowest satisfaction value due to having the longest waiting times for travelers to get an available vehicle. In our ECPV, waiting time is considered as a part of real travel time, and long waiting times affect the satisfaction of travelers. The reason for low satisfaction values for T-share and PTRider are the high detour distance and high travel charge, respectively.

Traveling Time: Fig. 6 shows the total travel times to complete all ride requests, and demonstrates that, to complete all requests, vehicles need to be driving for greater total time in the non-sharing scenario. In addition, the total travel times of the T-Share and PTRider schemes are larger than those of our ECPV system. Fig. 7 shows the total detour time ratios for different ride matching strategies and the non-sharing scenario. The detour time ratio is defined as the ratio of real travel time (including the waiting time) to expected travel time. As shown in Fig. 7, the detour time ratio of the non-sharing scenario is the largest, and our ECPV system (i.e., SFM, SVBM, LVBM) achieves lower detour time ratios than other existing schemes (i.e., T-Share and PTRider).

Note that, the detour time rate in our system can be represented as $R_t = \frac{T_r}{T_e} = \frac{T_w + T_d}{T_e}$ where R_t represents the detour time ratio, T_r and T_e are the real and expected traveling times, respectively, and T_w and T_d are the waiting and driving times, respectively. Hence, as the definition of detour time ratio, the long waiting time will lead to large detour time ratio, and thus we state that the high detour time ratio is caused by long waiting times.

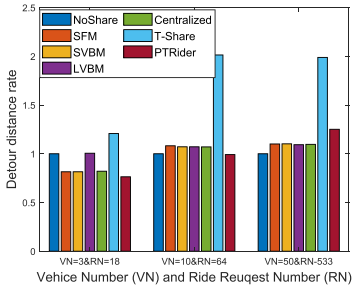


Fig. 9. Detour distance rate.

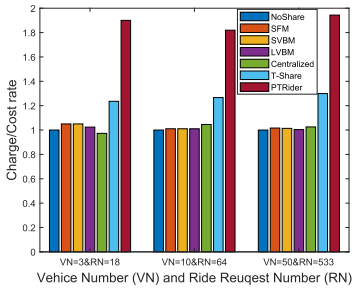


Fig. 10. Travel charge/cost rate.

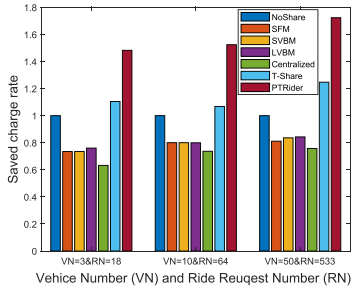


Fig. 11. Saved charge rate.

Traveling Distance: Fig. 8 shows the total travel distances to complete all ride requests of all travelers, demonstrating that ridesharing can reduce the total distance of vehicle travel, and that our scheme can achieve shorter travel distance than the T-Share and PTRider schemes. Fig. 9 shows the total detour distance ratios. The detour distance ratio is defined as the ratio of real travel distance to expected travel distance (i.e., the shortest distance to complete the ride request). As shown in Fig. 9, T-Share incurs the largest detour distance ratios, and the detour distance ratios of the other ridesharing schemes are around 1.

Traveling Charge: Fig. 10 shows the rate of cost and charge, which shows that all our proposed ride matching strategies (i.e., SFM, SVBM, LVBM) can achieve approximate balance between cost and charge (i.e., the rate is around 1). However, the charge in PTRider is much larger than the cost (i.e., the rate is larger than 1). This means that travelers in PTRider need to pay more money to complete their trips. Fig. 11 shows the saved charges of all ride requests. The saved charge rate is defined as the average rate of the charge of travelers in ridesharing to the charge that is needed in non-sharing. Fig. 11 shows that all our proposed ride matching strategies achieve low saved charge

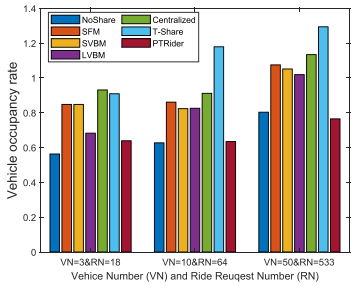


Fig. 12. Vehicle occupancy ratio.

rates, which means that in our strategies travelers need to pay less for their trips.

In addition, the reason of achieving highest travel chargecost rate and saved charge rate in PTRider is that PTRider considers the profits of drivers. The driver in the PTRider scheme will profit through compensation by conveying travelers, and the profits can be considered as the difference of the charge to travelers and the cost of the ride. Our system considers driverless cars as public vehicles, and thus does not need to consider the profits of drivers. The benefits of passengers that are considered are detour time and distance, and these are also considered by PTRider.

Vehicle Occupancy Ratios: Fig. 12 illustrates the vehicle occupancy ratios, which are defined as the average number of travelers in a vehicle at each time slot. As can be seen in Fig. 12, our ECPV system (i.e., SFM, SVBM, LVBM) can achieve higher vehicle occupancy ratio than the non-sharing scenario and the PTRider scheme. Although the T-Share scheme can achieve a larger vehicle occupancy ratio than our ECPV system, the detour distance rate is larger (as shown in Fig. 9), meaning that travelers need to stay in vehicles longer to complete their trips.

As shown in Fig. 5∼Fig. 12, the centralized scheme can always achieve slightly better performance than our edge based schemes. The reason for this is the centralized scheme is global optimal and our edge based schemes are only locally optimal. However, as mentioned in Section VI-D and Table II, the computation complexity of the centralized scheme is much larger than those of our edge based schemes. In addition, as shown in Table III, the total latency in the centralized scheme is much higher than that of our edge based schemes. Hence, although our edge based schemes cannot achieve better optimization performance than the centralized scheme, our edge based schemes can achieve lower service response latency to travelers, which is more suitable for public vehicle systems.

Different Pickup and Dropoff Location Distribution: Fig. 13∼Fig. 16 show the satisfaction rate, travel time rate, travel distance rate, and the saved charge rate in our ECPV system (i.e., SFM, SVBM, LVBM) with non-uniform and random ride request distributions. The satisfaction rate, travel time rate, travel distance rate, and saved charge rate are defined as the satisfaction, travel time, travel distance, and saved charge, respectively, in SFM, SVBM, LVBM over those of the non-sharing scenario. As shown in Fig. 13∼Fig. 16, the non-uniform ride request distribution can always achieve higher satisfaction, and lower travel time rate and travel distance rate, as well as saving more charge, in comparison with the random ride request distribution.

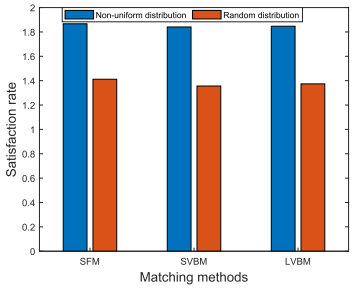


Fig. 13. Satisfaction rate in non-uniform and random ride request distribution.

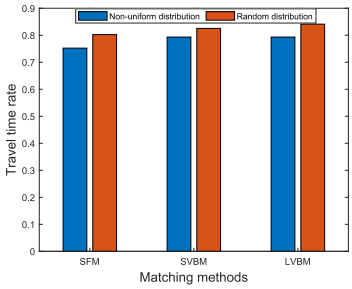


Fig. 14. Travel time rate in non-uniform and random ride request distribution.

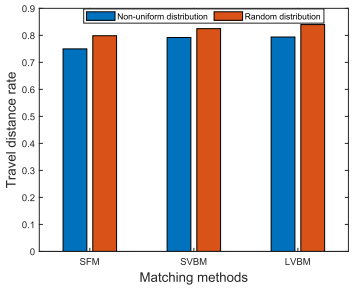


Fig. 15. Travel distance rate in non-uniform and random ride request distribution.

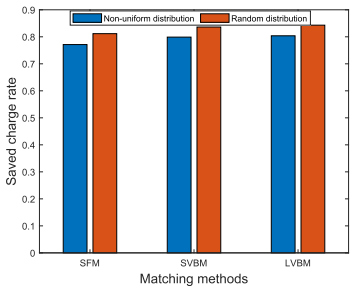


Fig. 16. Saved charge rate in non-uniform and random ride request distribution.

The reason for this is that travelers with non-uniform pick-up and drop-off positions will have much higher probability of sharing routes (i.e., the probability that two travelers have the same traveling routes) in comparison with travelers with random pick-up and drop-off positions.

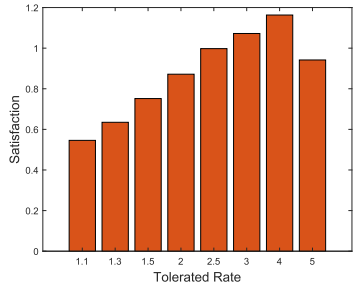


Fig. 17. Satisfaction in different tolerated rate.

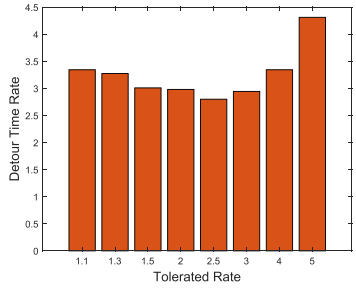


Fig. 18. Detour time rate in different tolerated rate.

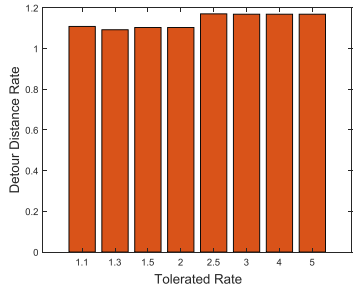


Fig. 19. Detour distance rate in different tolerated rate.

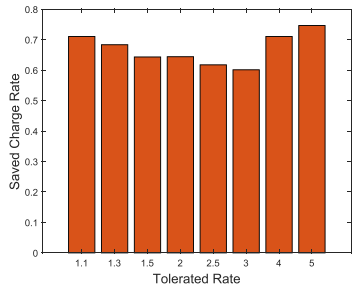


Fig. 20. Saved charge rate in different tolerated rate.

Fig. 17∼ Fig. 20 show the satisfaction rate, detour time rate, detour distance rate, and the saved charge rate in our ECPV system (using SFM as an example) with different tolerated ratios. As shown in Fig. 17∼Fig. 20, when the tolerated ratio is 4, the greatest satisfaction can be achieved. When the tolerated ratio is 2.5, the greatest detour time rate and saved charge rate can be achieved. When tolerated ratio is 1.3, the greatest detour distance rate is achieved. Hence, to balance the performance of satisfaction, detour time/distance rate, and saved charge rate, a suitable tolerated ratio in our system should be in the range 1.5 to 3.

VI. Related Works

In ride-trip sharing, schemes are designed to stimulate travelers to share vehicles when they have similar destinations or required routes [7]–[12], [33]–[37]. For example, Ma *et al.* [8], [10] proposed a dynamic taxi ridesharing service, namely *T-share*, which can serve real-time taxi requests and achieve taxi sharing among customers, reducing total travel distance. Similarly, Thangaraj *et al.* [33] proposed an optimized dynamic ridesharing system, denoted *Xhare-a-Ride*, which can provide multiple ride options for customers based on several constraints, such as detour and walking preferences. Some additional strategies have been developed to improve ridesharing efficiency by altering the pick-up times and locations [38], [39], while other strategies were developed for bus scheduling and multi-modal ridesharing [40], [41]. However, these strategies will lead to travel inconvenience because travelers must walk a distance to their destinations or cannot complete their trips in the expected time. Additional strategies were developed to reduce the computational overhead of ride matching [3], [5], [13], to investigate a minimum number of vehicles on roads to complete all ride requests [4], and others. Moreover, cost-sharing mechanisms and fair pricing models have been investigated to ensure the profits of both drivers and riders [42], [43]. In addition, pricing schemes (e.g., *Social Cost* [44]) that focus on the pricing of detour data transmission in vehicular networks, can provide some insights into the design of a travel charge model for public vehicle system.

Additionally, a number of efforts focused on edge computing and vehicle placement [45]–[50]. For instance, Rodrigues *et al.* [46] comprehensively investigated computation offloading and resource allocation in mobile edge computing. Xiong *et al.* [48] proposed a charging station placement scheme to optimally deploy charging stations with consideration for traveling cost and queue cost. Likewise, Verma *et al.* [50] investigated network methodologies for real-time analytics of massive IoT data.

Although existing ridesharing schemes and PV systems can induce travelers to share rides, most are cloud based schemes, where ride requests are launched and sent to the cloud, and ride matching is conducted in data centers. By doing this, heavy computation tasks and high link congestion will occur on cloud and network, especially when a large number of ride requests are launched in a short period of time. In this case, high latency in ridesharing responses to travelers will occur, significantly reducing QoE. Also, some schemes only consider travel distance/time minimization as objectives for matching ride requests with vehicles, and cannot take full account of the traveler satisfaction of customers.

Unlike existing schemes, in this study, we developed an edge computing based public vehicle system, denoted the ECPV system, which considers both the information transmission mechanism and the ride matching mechanism, as well as a depot placement mechanism, to prompt travelers to share rides during trips while achieving high user satisfaction. Particularly, in our ECPV system, an edge computing based ride request transmission mechanism is considered to take advantage of edge device to reduce the computation tasks in the data center, and the integration of multiple metrics, including travel distance, travel time, and travel charge, are considered in ride matching to illustrate and achieve traveler satisfaction. Also, self-driving cars are considered as public vehicles in our system.

VII. Conclusion

In this paper, we proposed an edge computing based public vehicle (ECPV) system to induce travelers to share rides. The public vehicle scheduling (i.e., ride matching) problem is formalized as an optimization problem with the objective of maximizing traveler satisfaction, which is calculated as a combination of travel distance, time, and travel charge. We jointly developed an edge computing based ride request transmission mechanism and a tree based heuristic matching mechanism to effectively match rider-vehicle pairs with considering traveler satisfaction. Additionally, a graph partition based depot placement mechanism was developed to select suitable locations to park idle vehicles, thereby reducing the number of vehicles on roads. Through extensive evaluations, our experimental results show that our ECPV system achieves better efficiency in terms of traveler satisfaction, distance, and time, travel charge to travelers, and vehicle occupancy ratio, compared to a system in which non-sharing is in place. As a future work, we will study ridesharing with consideration for vehicle transferring on trips (i.e., one trip served by multiple vehicles and also intend to conduct in-depth study of these related subjects, including edge computing and vehicle placement.

References

[1] J. Lin, W. Yu, N. Zhang, X. Yang, and L. Ge, "Data integrity attacks against dynamic route guidance in transportation-based cyber-physical systems: Modeling, analysis, and defense," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8738–8753, Sep. 2018.

[2] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of Things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[3] M. Zhu, X. Liu, and X. Wang, "An online ride-sharing path-planning strategy for public vehicle systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 616–627, Feb. 2019.

[4] M. Zhu *et al.*, "Public vehicles for future urban transportation," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3344–3353, Dec. 2016.

[5] M. Ota, H. Vo, C. Silva, and J. Freire, "Stars: Simulating taxi ride sharing at scale," *IEEE Trans. Big Data*, vol. 3, no. 3, pp. 349–361, Sep. 2017.

[6] Y. He, J. Ni, X. Wang, B. Niu, F. Li, and X. Shen, "Privacy-preserving partner selection for ride-sharing services," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5994–6005, Jul. 2018.

[7] Y. Lai, F. Yang, L. Zhang, and Z. Lin, "Distributed public vehicle system based on fog nodes and vehicular sensing," *IEEE Access*, vol. 6, pp. 22011–22024, 2018.

[8] S. Ma, Y. Zheng, and O. Wolfson, "T-share: A large-scale dynamic taxi ridesharing service," in *Proc. IEEE 29th Int. Conf. Data Eng.*, pp. 410–421, Apr. 2013.

[9] N. Ta, G. Li, T. Zhao, J. Feng, H. Ma, and Z. Gong, "An efficient ride-sharing framework for maximizing shared route," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 2, pp. 219–233, Feb. 2018.

[10] 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.

[11] Peng Cheng, Hao Xin, and Lei Chen, "Utility-aware ridesharing on road networks," in *Proc. ACM Int. Conf. Manage. Data*, SIGMOD, New York, NY, USA, 2017, pp. 1197–1210.

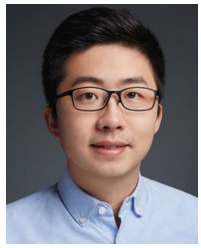
[12] M. Zhu, X. Liu, and X. Wang, "Joint transportation and charging scheduling in public vehicle systemsa game theoretic approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2407–2419, Aug. 2018.
[13] H. Yan, R. Jin, F. Bastani, and X. S. Wang, "Large scale real-time ridesharing with service guarantee on road networks," in *Proc. Int. Conf. Very Large Data Bases Endowment*, 2013, vol. 7, no. 14, pp. 1–14.
[14] L. Chen, Y. Gao, Z. Liu, X. Xiao, and C. S. Jensen, "PTRider: A price-and-time-aware ridesharing system," *Proc. VLDB Endowment*, vol. 11, no. 12, pp. 1938–1941, 2018.
[15] M. Wei and Y. Meng, "Research on the optimal route choice based on improved Dijkstra," in *Proc. IEEE Workshop Adv. Res. Technol. Industry Appl.*, Sep. 2014, pp. 303–306.
[16] W. Yu *et al.*, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2018.
[17] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surv. Tut.*, vol. 19, no. 3, pp. 1657–1681, Jul.–Sep. 2017.
[18] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2322–2358, Oct.–Dec. 2017.
[19] P. Chen, J. Liu, and W. Chen, "A fuel-saving and pollution-reducing dynamic taxi-sharing protocol in VANETs," in *Proc. IEEE 72nd Veh. Technol. Conf.*, Sep. 2010, pp. 1–5.
[20] A. Arora, M. Yun, T. Kim, Y. Zhou, and H. Choi, "Automated ride share selection using vehicular area networks," in *Proc. IEEE Int. Conf. Commun. Workshops*, Jun. 2009, pp. 1–6.
[21] C. Xu, F. Zhao, J. Guan, H. Zhang, and G. Muntean, "Qoe-driven user-centric VoD services in urban multihomed P2P-based vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 5, pp. 2273–2289, Jun. 2013.
[22] C. Li, Y. Zhang, T. H. Luan, and Y. Fu, "Building transmission backbone for highway vehicular networks: Framework and analysis," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8709–8722, Sep. 2018.
[23] J. Lin, W. Yu, X. Yang, Q. Yang, X. Fu, and W. Zhao, "A real-time en-route route guidance decision scheme for transportation-based cyberphysical systems," *IEEE Trans. Veh. Technol.*, vol. 66, no. 3, pp. 2551–2566, Mar. 2017.
[24] D. Zhang *et al.*, "Near-optimal and truthful online auction for computation offloading in green edge-computing systems," *IEEE Trans. Mobile Comput.*, vol. 19, no. 4, pp. 880–893, Apr. 2020.
[25] J. He, D. Zhang, Y. Zhou, and Y. Zhang, "A truthful online mechanism for collaborative computation offloading in mobile edge computing," *IEEE Trans. Ind. Informat.*, vol. 16. no. 7, pp. 4832–4841, Jul. 2020.
[26] Q. Pham, T. Leanh, N. H. Tran, B. J. Park, and C. S. Hong, "Decentralized computation offloading and resource allocation for mobile-edge computing: A matching game approach," *IEEE Access*, vol. 6, pp. 75868–75885, 2018.
[27] T. Zhang, "Data offloading in mobile edge computing: A coalition and pricing based approach," *IEEE Access*, vol. 6, pp. 2760–2767, 2018.
[28] Q. Pham, H. T. Nguyen, Z. Han, and W. Hwang, "Coalitional games for computation offloading in NOMA-enabled multi-access edge computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1982–1993, Feb. 2020.
[29] M. Khanjary, K. Faez, M. R. Meybodi, and M. Sabaei, "Persiangulf: An autonomous combined traffic signal controller and route guidance system," in *Proc. IEEE Veh. Technol. Conf.*, Sep. 2011, pp. 1–6.
[30] J.-W. Ding, C.-F. Wang, F.-H. Meng, and T.-Y. Wu, "Real-time vehicle route guidance using vehicle-to-vehicle communication," *IET Commun.*, vol. 4, no. 7, pp. 870–883, Apr. 2010.
[31] F. Tianheng, Y. Lin, G. Qing, H. Yanqing, Y. Ting, and Y. Bin, "A supervisory control strategy for plug-in hybrid electric vehicles based on energy demand prediction and route preview," *IEEE Trans. Veh. Technol.*, vol. 64, no. 5, pp. 1691–1700, May 2015.
[32] J. Pan, I. S. Popa, K. Zeitouni, and C. Borcea, "Proactive vehicular traffic rerouting for lower travel time," *IEEE Trans. Veh. Technol.*, vol. 62, no. 8, pp. 3551–3568, Oct. 2013.
[33] R. S. Thangaraj, K. Mukherjee, G. Raravi, A. Metrewar, N. Annamaneni, and K. Chattopadhyay, "Xhare-a-Ride: A search optimized dynamic ride sharing system with approximation guarantee," in *Proc. IEEE 33rd Int. Conf. Data Engi.*, Apr. 2017, pp. 1117–1128.
[34] V. Armant and K. N. Brown, "Minimizing the driving distance in ride sharing systems," in *Proc. IEEE 26th Int. Conf. Tools Artif. Intell.*, Nov. 2014, pp. 568–575.
[35] N. Bicocchi, M. Mamei, A. Sassi, and F. Zambonelli, "On recommending opportunistic rides," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3328–3338, Dec. 2017.
[36] S. Ma and O. Wolfson, "Analysis and evaluation of the slugging form of ridesharing," in *Proc. 21st Int. Conf. Adv. Geographic Inf. Syst.*, SIGSPATIAL13, New York, NY, USA, 2013, pp. 64–73.
[37] J. Lin, S. Sasidharan, S. Ma, and O. Wolfson, "A model of multimodal ridesharing and its analysis," in *Proc. 17th IEEE Int. Conf. Mobile Data Manage.*, 2016, pp. 164–173.
[38] V. Monteiro de Lira, R. Perego, C. Renso, S. Rinzivillo, and V. C. Times, "Boosting ride sharing with alternative destinations," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2290–2300, Jul. 2018.
[39] P. Goel, L. Kulik, and K. Ramamohanarao, "Optimal pick up point selection for effective ride sharing," *IEEE Trans. Big Data*, vol. 3, no. 2, pp. 154–168, Jun. 2017.
[40] T. Ma, "On-demand dynamic Bi-/multi-modal ride-sharing using optimal passenger-vehicle assignments," in *Proc. IEEE Int. Conf. Environ. Elect. Eng. IEEE Ind. Commercial Power Syst. Eur.*, Jun. 2017, pp. 1–5.
[41] X. Hao, W. Jin, and M. Wei, "Max-min ant system for bus transit multi-depot vehicle scheduling problem with route time constraints," in *Proc. 10th World Congr. Intell. Control Automat.*, Jul. 2012, pp. 555–560.
[42] M. Asghari, D. Deng, C. Shahabi, U. Demiryurek, and Y. Li, "Price-aware real-time ride-sharing at scale: An auction-based approach," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2016, pp. 1–10.
[43] M. Furuhata *et al.*, "Ridesharing: The state-of-the-art and future directions," *Transp. Res. Part B*, vol. 57, no. 57, pp. 28–46, 2013.
[44] O. Urra and S. Ilarri, "Spatial crowdsourcing with mobile agents in vehicular networks," *Veh. Commun.*, vol. 17, pp. 10–34, 2019.
[45] T. K. Rodrigues, K. Suto, and N. Kato, "Edge cloud server deployment with transmission power control through machine learning for 6G internet of things," *IEEE Trans. Emerg. Topics Comput.*, to be published, doi: 10.1109/TETC.2019.2963091.
[46] T. K. Rodrigues, K. Suto, H. Nishiyama, J. Liu, and N. Kato, "Machine learning meets computation and communication control in evolving edge and cloud: Challenges and future perspective," *IEEE Commun. Surv. Tut.*, vol. 22, no. 1, pp. 38–67, Jan.–Mar. 2020.
[47] Y. Shih, W. Chung, A. Pang, T. Chiu, and H. Wei, "Enabling low-latency applications in fog-radio access networks," *IEEE Netw.*, vol. 31, no. 1, pp. 52–58, Jan./Feb. 2017.
[48] Y. Xiong, J. Gan, B. An, C. Miao, and A. L. C. Bazzan, "Optimal electric vehicle fast charging station placement based on game theoretical framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2493–2504, Aug. 2018.
[49] S. F. Chou, Y. J. Yu, and A. C. Pang, "Mobile small cell deployment for service time maximization over next generation cellular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5398–5408, Jun. 2017.
[50] S. Verma, Y. Kawamoto, Z. M. Fadlullah, H. Nishiyama, and N. Kato, "A survey on network methodologies for real-time analytics of massive IoT data and open research issues," *IEEE Commun. Surv. Tut.*, vol. 19, no. 3, pp. 1457–1477, Jul.–Sep. 2017.

Jie Lin received the B.S. and Ph.D. degrees in the Department of Computer Science and Technology from Xi'an Jiaotong University, in 2009, and 2013, respectively. He is currently an Associate Professor with the School of Computer Science and Technology from Xi'an Jiaotong University. His research interest includes edge computing, internet of things, and security.

Wei Yu received the B.S. degree in electrical engineering from the Nanjing University of Technology, Nanjing, China, in 1992, the M.S. degree in electrical engineering from Tongji University, Shanghai, China, in 1995, and the Ph.D. degree in computer engineering from Texas A&M University, in 2008. He is currently a Full Professor with the Department of Computer and Information Sciences, Towson University, MD, USA. His research interest includes security and privacy, cyber-physical systems, Internet of Things, data science, and machine learning driven applications.

Xinyu Yang received the diploma in computer science and technology from the Xi'an Jiaotong University of China, in 2001 and the bachelor's, master's, and Ph.D. degrees from Xi'an Jiaotong University, in 1995, 1997, and 2001, respectively. He is currently a Professor with the School of Computer Science and Technology, Xi'an Jiaotong University. His research interest includes wireless communication, mobile ad hoc networks, and network security.

Peng Zhao received the B.S. and Ph.D. degrees in computer science and technology from Xian Jiaotong University, Xian, China, in 2007 and 2013, respectively. He is currently an Associate Professor with the School of Computer Science and Technology, Xi'an Jiaotong University. From 2013 to 2014, he was a Research Fellow with Nanyang Technological University, Singapore. His current research interest includes edge computing, network security, and deep learning.



Hanlin Zhang received the B.S. degree in software engineering from Qingdao University, in 2010. He received the M.S. degree in applied information technology and the Ph.D. degree in information technology from Towson University, MD, U.S., in 2011 and 2016, respectively. He is currently working with Qingdao University as an Assistant professor with the College of Computer Science and Technology. His research interest includes cloud computing security, blockchain technology and internet of things security.

Wei Zhao received the Ph.D. degree in computer and information sciences from the University of Massachusetts Amherst, Amherst, MA, USA, in 1986. Since 1986, he has been a Faculty Member with Amherst College, The University of Adelaide, and Texas A&M University. From 2005 to 2006, he was the Director for the Division of Computer and Network Systems, National Science Foundation, USA, when he was on leave from Texas A&M University, College Station, TX, USA, where he was the Senior Associate Vice President for research and a Professor of computer science. He was the Dean of the School of Science, Rensselaer Polytechnic Institute, Troy, NY, USA, from 2007 to 2008. He was the Rector of the University of Macau, Macau. He is currently a Chief Research Officer with the American University of Sharjah. As an elected IEEE fellow, he has made significant contributions in distributed computing, real-time systems, computer networks, and cyberspace security.