Received March 7, 2018, accepted April 3, 2018, date of publication April 9, 2018, date of current version May 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2824319

Distributed Public Vehicle System Based on Fog Nodes and Vehicular Sensing

YONGXUAN LAI[1,2], FAN YANG[2,3], LU ZHANG[1], AND ZIYU LIN[4]

[1]Software School, Xiamen University, Xiamen 361005, China
[2]Shenzhen Research Institute, Xiamen University, Shenzhen 518000, China
[3]Department of Automation, Xiamen University, Xiamen 360000, China
[4]Department of Computer Science, Xiamen University, Xiamen 360000, China

Corresponding author: Yongxuan Lai (laiyx@xmu.edu.cn)

This work was supported in part by the Natural Science Foundation of China under Grant 61672441 and Grant 61472335, in part by the Shenzhen Basic Research Program under Grant JCYJ20170818141325209, in part by the National Key Technology Support Program under Grant 2015BAH16FF01, and in part by the State Scholarship Fund of China Scholarship Council under Grant 201706315020.

ABSTRACT Recent years there has been increasing concern about the rider demand responsive systems and the vehicular ad hoc networks. On one hand, centralised taxi platforms such as Uber and Didi Taxi are popular and changing our daily life; on the other hand, vehicles are equipped with more and more sensors and are capable to calculate, store, and communicate with other vehicles or road side units, forming vehicle-to-vehicle or vehicle-to-infrastructure communications. However, little effort has been devoted to integrating these two fields. In this paper, we propose a distributed public vehicle (PV) system that integrates the rider demand responsive system and *ad hoc* vehicular technologies, where the concept of fog computing and vehicular sensing are adopted for the system design. The challenges lie in that the PV scheduling problem itself is NP-hard, and careful design of scheduling and cooperation schemes among nodes are needed as they are ubiquitously connected at the edge of networks. The proposed PV system adopts a heuristic request insertion algorithm and a cooperative strategy among vehicle nodes, fog nodes, and the cloud to dispatch requests and to schedule routes for PVs. Experimental studies on real-world data sets demonstrate that the proposed scheme achieves higher service ratio of requests and better efficiency than other transit methods. Furthermore, the distributed vehicular sensing is demonstrated to be capable of collecting feasible metadata for scheduling applications. To the best of our knowledge, this paper is the first report on the integration of fog nodes and vehicular sensing for the rider request responsive scheduling systems.

INDEX TERMS Fog nodes, public vehicles, route scheduling, vehicle path problem, vehicular sensing.

I. INTRODUCTION

Due to the development of mobile-oriented cloud architectures and technologies, recent years there has been increasing concern about the Rider Demand Responsive Systems (RDRS) [1]–[3]. Platforms such as Uber and Didi have brought great changes to the daily lives of urban citizens. Users or riders submit requests to RDRS to demand transit services; RDRS then makes a match between the incoming requests and vehicles, and guides the drivers to pick up the riders. Much research effort has been focused on increasing the utility and sharing factors of the vehicles, where carpooling and ridesharing are becoming hot topics [4]. More recently, with the progress on driverless/autonomous electric vehicles, a new type of high occupancy vehicles called *public vehicles* (PV) have emerged [5]. Public vehicles, together with the centralized cloud-based demand-responsive system, provide ridesharing trips with service guarantee to supplement or replace buses, private cars, and taxis in urban areas.

On the other aspect, with the development of vehicular and communication technologies, there emerges a new technology called Vehicular Ad Hoc Networks (VANETs) that integrate the capabilities of new generation wireless networks to vehicles [6]–[8]. IEEE 802 committee define wireless communication standard IEEE 802.11p [9], which serves specifically for vehicle-to-infrastructure (V2I) communication. The Federal Communications Commission has allocated 75 MHz of bandwidth, which operates on 5.9 GHz channel for short range communications. Vehicles communicate with other vehicles directly forming vehicle-to-vehicle communication (V2V), or communicate with fixed equipment next to the road, referred to as road side unit (RSU), forming V2I communications. VANETs enable the concept of Smart Car and Intelligent Transportation Systems (ITS), in which information, sensing and communication technologies are applied in the fields of trip services, road transportation, and traffic management [10].

However, little effort has been devoted to integrating these two fields. This paper aims to propose a distributed public vehicle system that integrates the rider demand responsive system and VANETs technologies. Specially, the concept of fog computing and vehicular sensing are adopted for the system design. Fog computing extends the traditional cloud computing paradigm to the edge of networks [7], [11], [12], where fog nodes are a new kind of nodes that are capable of carrying out a substantial amount of storage (rather than stored primarily in cloud data centers), communication (rather than routed over the internet backbone), control, configuration, measurement and management. For example, Intel's Next Unit of Computing [13] is a small-form-factor computer, whose motherboard measures 4 × 4 inches and could be integrated to the road site units deployed at the edge of networks. Fog nodes like these are able to gather and maintain metadata about the road network, requests, and vehicles. These gathered metadata could then be used to guide the dispatch of requests and be used as input for the matches between riders and PVs.

There are several advantages of integrating a distributed public vehicle system with fog computing and vehicular sensing. First, data readings or metadata are stored and gathered at the fog nodes, which are close to the data sources and at the edge of the network, rather than being uploaded to the cloud by 4G communications. The V2I communications are adopted as the main communication channel, which incurs much lower economic cost and time latency. Second, fog nodes could perform the calculations locally to make matches between PVs and riders. This enables the distributed computing and processing of the public vehicle scheduling problem. Recently, several applications [10], [12] on VANETs have emerged to embrace the idea that data are processed using computing resources located at the edge of the network, accessible through wireless protocols, and optionally using remote resources in the cloud.

The challenges of distributed public vehicle system lie in two aspects. First, the public vehicle scheduling problem is a member of the general class of the Dial-a-Ride Problem [3], [14], which is NP-hard. Second, vehicular nodes are ubiquitously connected at the edge of networks, so there should be careful design of scheduling and cooperation schemes among fog nodes, as well as collaboration between vehicular nodes and the cloud [15], [16]. The proposed scheme adopts a heuristic request insertion algorithm and a cooperative strategy among vehicles nodes, fog nodes, and the cloud to schedule the PVs and serve the requests from riders. The main contributions of this paper are as follows:

- We model the demand responsive transit service in a distributed environment that integrates fog nodes and vehicular sensing. Fog nodes act as intermediate nodes to store and gather metadata, which are sensed and extracted by vehicular nodes. Fog nodes also receive requests that are dispatched from the cloud, and locally make assignments among the riders and PVs.
- We propose a Fog-based Public Vehicle System (FPVS). The system includes components of metadata gathering, cost estimation, request answering and route scheduling. A heuristic algorithm based on fog-cloud coordination is proposed to dispatch requests and to schedule routes for PVs.
- We conduct experiments on real-world datasets to verify the effectiveness of the proposed scheme. The distributed vehicular sensing is capable of collecting feasible metadata for scheduling applications, and FPVS achieves higher service ratio of requests and better efficiency than other transit methods. To the best of our knowledge, this is the first report on the integration of fog nodes and vehicular sensing for the rider demand responsive systems.

The rest of the paper is structured as follows: section II describes the related work; section III introduces some preliminaries and defines the problem; section IV presents the overview of the FPVS scheme; section V and VI describe the detailed algorithms of request dispatch, route scheduling, and metadata maintenance; section VII describes the environmental setup and analyzes the simulation results, and finally section VIII concludes the paper.

II. RELATED WORK

In this section we review three categories of related works to position our work in the research community.

A. DEMAND-RESPONSIVE TRANSIT SERVICE

Demand-responsive transit service is an alternative travel method to personal vehicles, carpool/vanpool and regular transit service. presented protocols for It is comprised of a number of customer requests that need to be served door-to-door or curb-to-curb by a set of vehicles [17], [18].

One important issue in demand-responsive transit service is to devise a real-time matching algorithm that determines the best vehicle (taxi, cab, bus) to satisfy incoming service requests. Ma *et al.* [1] proposed a taxi searching algorithm using a spatio-temporal index to quickly retrieve candidate taxis that are likely to satisfy a user request. The algorithm checks each candidate taxi and inserts the query's trip into the schedule of the taxi that satisfies the query with minimum additional incurred travel distance. Based on [1], Ma *et al.* [2] reported a real-time taxi-sharing system based on the mobile-cloud architecture. Drivers and passengers exchange service and demands using an application installed on their smartphones, and the taxi that minimizes the increased travel distance of the ride request would be selected to pick up the new passenger. Zhu *et al.* [5]

proposed a heuristic precedence constrained origin-destination insertion algorithm for the public vehicle system to minimize vehicles' total travel distance with service guarantee such as low detour ratio. Based on [5], the same authors proposed a path planning strategy that focuses on a limited potential search area for each vehicle by filtering out requests that violate passenger service quality level [19], and studied the joint transportation and charging scheduling for PV systems to balance the transportation and charging demands, ensuring the long-term operation [20]. More recently, Cheng *et al.* [4] formulated the utility-aware ridesharing problem on road networks. It assigns time-constrained riders to capacity-constrained vehicles to maximize the entire utility value, which includes the vehicle-related utility, the riders-related utility, and the trajectory-related utility.

The aforementioned schemes assume a centralised server to execute algorithms for demand-responsive transit service, and the desired metadata are available as input for the algorithms. Different from those algorithms, the proposed scheme focuses on distributive processing of the algorithm, where matching and scheduling algorithms are executed on the fog nodes, and hence avoids a bottleneck of computing and storage. Moreover, the proposed scheme integrates the metadata gathering into the whole framework, where data is gathered and stored at distributed fog nodes in the edge of networks.

Demand-responsive transit service could be abstracted as a member of the general class of the Dial-a-Ride Problem [3], [14], which focuses on scenarios of planning schedules for vehicles, subject to the time constraints on pickup and delivery events. The proposed approach dispatches requests to distributed fog nodes that maintain lists of vehicles, which actually partitions a large dial-a-ride problem into multiple smaller ones that are easier to solve.

B. VEHICULAR SENSING

Cooperative vehicular and urban sensing is at the heart of the intelligent and green city traffic management. Lee *et al.* [21] proposed the MobEyes system for proactive urban monitoring. The system exploits the vehicle mobility to opportunistically diffuse concise summaries of the sensed data, and it harvests these summaries and builds a low-cost distributed index of the stored data to support various applications. Hull *et al.* [22] proposed a data management system CarTel for querying and collecting data from mobile vehicles, which enables the application development with data collected. Delot *et al.* [23] proposed a pull-based data gathering strategy called GeoVanet, which adopts a DHT-based (DHT, dynamic hash table) model to identify a fixed geographical location where a mailbox is dedicated to the query. Users are able to send queries to a set of cars and find the desired information in a bounded time. Płaczek [24] introduced a method of selective data collection for traffic control applications. The underlying idea is to detect the necessity of data transfers on the basis of uncertainty determination of the traffic control decisions, and sensor data are transmitted from vehicles to the control node only at selected time moments. Skordylis and Trigoni *et al.* [25] presented protocols for traffic-monitoring in vehicular networks. They defined two operation modes, multi-hop forwarding (MF) mode and delay-tolerant mode (DM). During MF mode messages are forwarded through the shortest path to destination, while in DM mode messages are only forwarded at intersections to keep them inside the shortest path when the current carrier moves away.

As the volume of sensed data might be large, there are also some research on reducing the volume of sensed data and the cost of gathering them. Li *et al.* [26] proposed a cooperative storage solution in vehicular sensor networks for mobile surveillance. Nodes first capture images from links/streets and then eliminate redundant data by exchanging image tags between vehicles, and it also includes a distributed storage balancing mechanism to offload data from heavy-load nodes to light-load nodes. Lai *et al.* [10] proposed an efficient continuous event-monitoring framework based on fog nodes in VANETs, where a two-level threshold strategy is adopted to suppress unnecessary data upload and transmissions. In the monitoring phase, nodes sense the environment in low cost sensing mode. When the probability of event is high and exceeds some threshold, nodes transfer to the event-checking phase, where some nodes would be selected to transfer to the deep sensing mode to generate more accurate data of the environment.

The aforementioned approaches could be used for the proposed scheme to gather the metadata. FPVS takes full advantage of the computing and storage resources of vehicular nodes and fog nodes to perform calculations locally, which extracts knowledge from the raw sensed data, so a much smaller amount of sensed data are transmitted in the network.

C. FOG/EDGE COMPUTING

Fog nodes are able to provide computation, storage, and networking services between the end nodes and traditional clouds. Fog reduces service latency, and improves QoS, resulting in superior user-experience [10], [11]. Within the concept of fog/edge computing, more and more fog nodes are deployed at the edge of networks for various applications. Bonomi *et al.* [11] defined the characteristics of fog computing and its role in the Internet of Things. They emphasized the fact that the fog brings new elements to the realm of Internet of Things through reduction of service latency and improvement of QoS. Sharma and Wang [27] proposed a framework for coordinated processing between edge and cloud computing/processing by integrating advantages from both platforms. It exploits the network-wide knowledge and historical information available at the cloud center to guide edge computing units towards satisfying various performance requirements.

Another concept that is highly related to the fog computing is VANET Cloud, which extends traditional cloud computing paradigm to VANETs. Eltoweissy *et al.* [28] for the first time coined the term of Autonomous Vehicular Clouds (AVC), where a group of largely autonomous vehicles whose corporate computing, sensing, communication, and physical resources can be coordinated and dynamically allocated to authorized users. Hao *et al.* [15] gave a detailed description of fog computing and proposed a flexible software architecture to incorporate different design choices and user-specified polices.

The proposed scheme belongs to the applications of fog computing in VANETs. Yet to the best of our knowledge, it is the first report to schedule the demand-responsive PVs within the fog computing paradigm. The challenge of fog computing lies in that the nodes are ubiquitously connected at the edge of network, and there should be careful design of scheduling and cooperation schemes among the fog nodes, as well as collaboration between vehicular nodes and the cloud.

III. PRELIMINARIES AND PROBLEM DEFINITION

This section introduces some preliminaries of the paper, including the vehicular network model, the PVs and requests, and the pubic vehicle path problem.

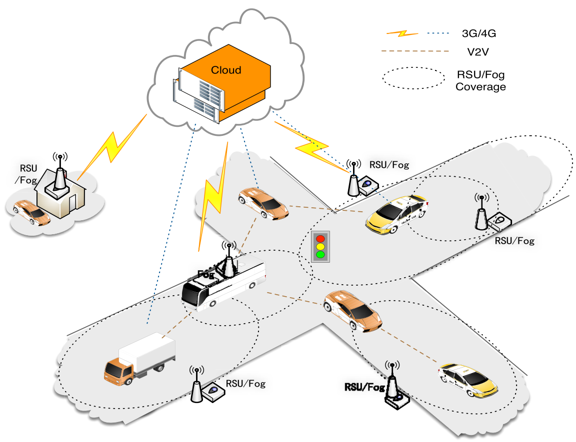


FIGURE 1. Illustration of a vehicular network with fog nodes (RSUs).

A. VEHICULAR NETWORK MODEL

Fig. 1 presents a three-layered vehicular network that consists of network layer, fog layer, and the cloud layer. At the network layer, vehicular nodes collect various sensing data, including speeds, locations, and etc. These data could be uploaded through V2I communications, and stored at the fog nodes. Fog nodes are with computing, storage and communication capabilities. Although vehicles can also be fog nodes, we assume only the road side units (RSUs) are adopted as the fog nodes. RSUs locate at the edge of the network and cooperate with the cloud, adopting a "fog-cloud" collaborative computing and storage strategy to provide a unified, efficient, and low-latency services for various applications. The three-layered vehicular cloud system makes some position-relevant and real-time applications possible.

B. PUBLIC VEHICLES AND REQUESTS

Vehicular nodes move on roads. Among them are a set of PVs that are deployed to fulfil the requests of riders. PVs are one type of high occupancy vehicles and may be driverless or autonomous electric vehicles. They provide ridesharing trips with service guarantee to supplement or replace buses, private cars, and taxis in urban areas.

Each vehicle, e.g. v, has a current location l_v and capacity c_v. PVs cruise on the roads/streets. They interact with the fog nodes and the cloud to fulfil transit requests from riders. Each request, e.g. r, is associated with a creation timestamp t_0, an origin location o, a destination location d, and a constraint pickup time window $[t_1, t_2]$. Requests are submitted to the cloud by riders through their mobile clients. The cloud would dispatch the requests to fog nodes, where the fog nodes would do local calculations based on gathered metadata to determine whether to accept or reject the requests.

C. PUBLIC VEHICLE PATH PROBLEM

Given a set of n requests R and a set of m vehicles V, a match between R and V could be represented by an $n \times m$ matrix $\mathcal{M}$ at current time t. Each individual item in $\mathcal{M}$ is denoted by $M_{i,j}$ and is set 0 by default. Suppose v is the i^{th} vehicle, r is the j^{th} request, $M_{i,j}$ is set to 1 if v is assigned to pick up request r, i.e. $M_{i,j} = 1$. $M_{i,j} = 1$ corresponds to an assignment: (v, r, t). A match is *feasible* if the following conditions are met:

$$\sum_{i=1}^{n} M_{i,j} \leq 1, \quad j = 1, .., m \tag{1}$$

$$\zeta(r) \leq \theta, \quad r \in R \tag{2}$$

$$\gamma(r.o) \in [r.t_1, r.t_2], \quad r \in R \tag{3}$$

$$\gamma(r.o) < \gamma(r.d), \quad r \in R \tag{4}$$

$$n_v \leq c_v, \quad v \in V \tag{5}$$

Constraint 1 implies that for a specific request there is at most one vehicle assigned to it. $\zeta(r)$ denotes the detour ratio of a request, which is further defined by Eq. 9 in section V-A. Constraint 2 means that the detour ratio for each vehicle is smaller than a system defined threshold θ. $\gamma(r.o)$, $\gamma(r.d)$ denote the time when the vehicle visits the origin and destination of r along the path. Constraint 3 implies the request is picked up within the constraint time window, and constraint 4 implies the request should be picked up before being delivered to the destination. n_v denotes the current number of riders in vehicle v; c_v denotes the capacity of v. Constraint 5 implies that the number of riders should be within the capacity of the vehicle.

Having these constraints satisfied, the overall cost of a match $\mathcal{M}$ between R and V is:

$$cost(R, V, \mathcal{M})_t = \sum_{M_{i,j}=1} ct(v, r, t) \tag{6}$$

where $ct(v, r, t)$ is the cost of assigning v to r at time t. Suppose $\mathbb{M}$ is the set of possible matches, the public vehicle system is to find a feasible match that minimises $cost(R, V, \mathcal{M})_t$:

$$\mathcal{M}^*_t = \underset{\mathcal{M}}{\text{argmin}} \{cost(R, V, \mathcal{M})_t : \mathcal{M} \in \mathbb{M}\} \tag{7}$$

The goal is to find a match $\mathcal{M}$ with minimal cost at time t. It is also called the Public Vehicle Path (PVP) problem, which is a member of Dial-a-Ride Problem [14] and NP-hard. Several metrics could be used to define function *cost* and to measure the utility of matching between the PVs and riders. From the view of operator companies, the number of picked up riders, the total distance travelled by the vehicle, and the overall revenue-cost ratio are the main concerns; yet from the view of riders, the waiting time, the total trip time, and the detour ratio are their main concerns [4]. Some of the measures are conflicting to others, and it is not possible to satisfy them all.

In real situations, large number of requests are submitted by riders and received at the scheduling system in realtime. So the problem has larger complexity with dynamic finite capacity and with more constraints, e.g. the time. This paper focuses on the PVP problem under distributed vehicular network environment, which includes the ride matching between PVs and riders, the path scheduling of vehicles, and the integrating the abundant resources of fog nodes with the vehicular nodes.

IV. FOG-BASED PUBLIC VEHICLE SCHEDULING SYSTEM

In this section we present an overview of the FPVS and describes the metadata gathering through vehicular sensing.

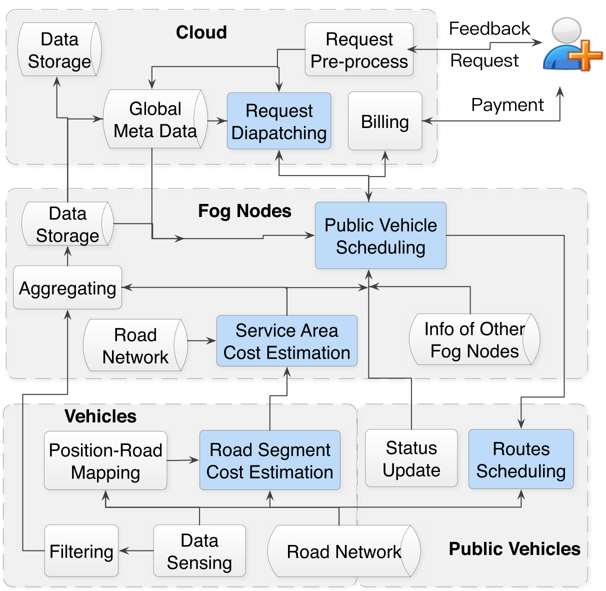


FIGURE 2. Overview of the distributed public vehicle scheduling system.

A. SYSTEM OVERVIEW

Fig. 2 depicts the framework of the proposed Fog-based Public Vehicle Scheduling system (FPVS).

The first part of procedures are the *metadata gathering and cost estimation*. Vehicles on the road sense the environment and generate data readings. These readings, which might be filtered and compressed if necessary, are routed to fog nodes or cloud for storage. Local calculations could be done in vehicular nodes, where positions are mapped to road segments, and the cost of travelling on road segments are estimated. These estimations are uploaded to fog nodes, based on which the fogs nodes further integrate and estimate the travelling cost within their service areas. Fog nodes also exchange metadata and information with other fog nodes.

The second part of procedures are the *request answering and route scheduling*. A request is submitted by a user and routed to the cloud. It is stored and pre-processed at the cloud, and would be dispatched to corresponding fog nodes to find matched vehicles. A fog node would assemble metadata that include the time cost of road segments within its service area, the status of the PVs, and data from the cloud and other fog nodes, to calculate whether to accept or reject the request. Public vehicle scheduling algorithms are executed locally on fog nodes. The feedback is routed back to the cloud and returned to the user or rider. If the request is accepted, the route scheduling modules on fog nodes and PVs would cooperate to guide the vehicles, and payments for the trip would be processed. Note that vehicles, fog nodes, and the cloud are assumed to have some common data and knowledge, e.g. the road network of the city, and they could do calculations distributively, with different order of computing capabilities. Upon a request, a rider can only be served by a single PV from his/her origin to destination, while the multi-hop scheme is not in the scope of this paper.

B. METADATA GATHERING AND COST ESTIMATION

As illustrated in Fig. 1, fog nodes are deployed at the edge of the network, and they receive periodical sensed readings from the vehicular nodes. A fog node is denoted by f, and is attached to a location l_f and a service area A_f. Vehicles within the service area are able to communicate with their fog node through direct V2I or multi-hop V2V communications. Each vehicle is equipped with a GPS device and is able to generate raw GPS readings. Here we assume vehicular nodes can do local "position to road segment" mapping. When a vehicle arrives at an intersection, it would send a data reading $d(id, t, I, dr)$ to the fog node, where id is the identification of the vehicle, t is the timestamp, I is the vehicle's current intersection, and dr is the direction that the vehicle heads to.

A road segment, denoted by $S(I_1, I_2)$, is determined by two intersections I_1, I_2. Given two sensing readings that are generated by a vehicle when travelling along a road segment $S(I_1, I_2)$ at time slot ts, i.e. $d_1 = (v, t_1, I_1, dr), d_2 = (v, t_2, I_2, dr), t_1, t_2 \in ts$, the cost of S measured by v is:

$$mv(v, S, ts) = t_2 - t_1 \tag{8}$$

The cost calculation is done locally at vehicular nodes, and at intersection I_2 the measured value $mv(v, S, ts)$ is uploaded to the fog node. This shifts the GPS mapping and cost estimations to the vehicular nodes, which avoids fog nodes becoming bottlenecks when uploading raw GPS records, especially when there are large number of vehicles on the roads.

Given a set of measured mv values, the time cost along road segment S at time slot ts could be estimated as follows:

$$cost(S, ts) = \frac{1}{|\Omega(S, ts)|} \sum_{mv \in \Omega(S,ts)} mv \tag{9}$$

where $\Omega(S, ts)$ is the set of received mv values that estimate the cost of S within time slot ts, and $|X|$ is the cardinality of set X. Metadata about the requests and PVs are also gathered and maintained at fog nodes and the cloud. For fog node f, the list of PVs within its service area is denoted by V_f, and the list of requests under its supervision is denoted by R_f.

V. REQUEST DISPATCH AND ROUTE SCHEDULING

A request is submitted by a user/rider from a mobile client to the FPVS. Due to the popularity of mobile phones, we assume the request is uploaded to the cloud through the 4G channel, though other channels like DSRC (Dedicated short-range communications [29]), WIFI could also be used for the uploading of requests. In this section we first introduce some terms and definitions related to the scheme, and then describe the details of request dispatch and route scheduling procedures.

A. TERMS AND DEFINITIONS

1) CURRENT PATH

Assume the requests currently assigned to vehicle v is denoted by $R = R_0 \cup R_1$. R_0 is the set of requests that have been picked up, so only the destinations need to be concerned; R_1 is the set of requests that have not been picked up, both the origins and destinations need to be considered. We denote the set of destinations of R_0 as U_0^d, and the set of origins and destinations of R_1 as U_1^o and U_1^d respectively. Given the set of vertexes $U = U_0^d \cup U_1^o \cup U_1^d$, the *current path* is denoted by *path*:

$$path = \{x_0, x_1, x_2, \ldots, x_k\}, \quad x_i \in U, \; k = |U| \tag{10}$$

where x_0 is the current location of v, i.e. l_v. For any specific request $r(t_0, x_i, x_j, [t_1, t_2]) \in R_1$, the pickup time is within the constraint window:

$$ct + cost(x_0, x_i) \in [t_1, t_2], \quad i < j \tag{11}$$

where ct denotes the current time, $cost(x_0, x_i)$ is the time cost travelling from current location x_0 to x_i along the path. $i < j$ means the origin is visited before the destination.

As shown in Fig. 3, a PV moves long a path to provide trip service. a^+, a^- denote the pickup and dropoff locations of rider a. The current path consists of two parts. The already visited path is $l_v \rightarrow a^+ \rightarrow b^+ \rightarrow b^- \rightarrow c^+$, and the path scheduled to visit is $c^+ \rightarrow a^- \rightarrow d^+ \rightarrow c^- \rightarrow d^-$.

2) COST OF PATH

Given a path=$\{x_0, x_1, x_2, \ldots, x_k\}$, its cost is defined as the minimal accumulated time cost from node x_0 to x_k starting at time t. The cost of path at t is denoted by $cost(path, t)$:

$$cost(path, t) = min(\tau_k - \tau_0) \tag{12}$$

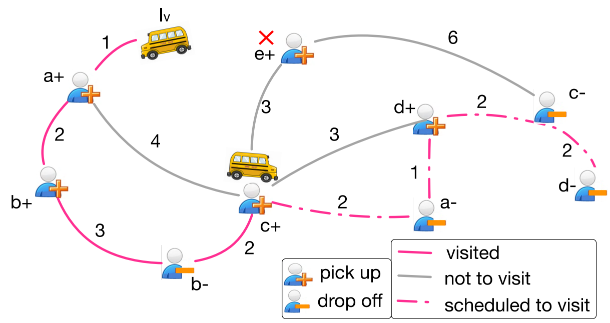


FIGURE 3. Illustration of the current path for a vehicle. a^+, b^- denote the pickup and drop off locations of rider a. The already visited path is $l_v \rightarrow a^+ \rightarrow b^+ \rightarrow b^- \rightarrow c^+$, and the path scheduled to visit is $c^+ \rightarrow a^- \rightarrow d^+ \rightarrow c^- \rightarrow d^-$.

where τ_i denotes the time point when x_i is visited, which is defined in a recursively way as follows:

$$\tau_0 = t, \quad \tau_i = \tau_{i-1} + cost(x_{i-1}, x_i, \tau_{i-1}), \; i \in [1, k] \tag{13}$$

where $cost(x_{i-1}, x_i, \tau_{i-1})$ is the time cost for a vehicle to travel from x_{i-1} to x_i starting at time point τ_{i-1}. The cost calculation is time-dependent.

As shown in Fig. 3, the numbers on the edges indicate the time cost travelling along the edges starting at position l_v. The total cost of the current path is: $cost(l_v, a^+) + cost(a^+, b^+) + cost(b^+, b^-) + cost(b^-, c^+) + cost(c^+, a^-) + cost(a^-, d^+) + cost(d^+, c^-) + cost(c^-, d^-) = 1 + 2 + 3 + 2 + 2 + 1 + 2 + 2 = 15$. In this research we assume a time-dependent road network of speeds is maintained at fog nodes, where the time cost of road segments could be accessed from a speed database that is extracted based on predictions using the historical trajectories and gathered vehicular readings [30]. The time cost is adopted as the main metric, while other metrics are used as constraints.

3) DETOUR RATIO

If request $r(t_0, o, d, [t_1, t_2])$ is accepted by a vehicle that has a scheduled *path*, the vehicle would travel along the *path* to pick up the rider at o within time window $[t_1, t_2]$ and then deliver the rider to d. The detour ratio is defined as:

$$\zeta(r) = \frac{cost(path, r.o, r.d)}{cost(r.o, r.d)} \tag{14}$$

where $cost(r.o, r.d)$ is the minimum cost of travelling from o to d in the road network, and $cost(path, r.o, r.d)$ is the cost of travelling from o to d along the scheduled *path*.

As shown in Fig. 3, the road segments along which rider a travels is $a^+ \rightarrow b^+ \rightarrow b^- \rightarrow c^+ \rightarrow a^-$, so the cost is $2 + 3 + 2 + 2 = 9$. Yet the shortest path of rider a from its origin to desination is $a^+ \rightarrow c^+ \rightarrow a^-$, whose cost is $4 + 2 = 6$. So the detour ratio is $9/6 = 1.5$. For rider e, his/her request is rejected because accepting it violates some detour constraints of other riders.

For a set of requests R, the average detour ratio is also defined as:

$$\zeta(R) = \frac{\sum_{r \in R} \zeta(r)}{|R|} \tag{15}$$

4) AVERAGE NUMBER OF ON-BOARD RIDERS

For a scheduled path, the average number of on-board riders is defined as follows:

$$\varphi(path) = \frac{\sum_{x \in path} ob(x)}{|path|} \tag{16}$$

where $ob(x)$ is the number of on-board riders on vertex x, and $|path|$ is the number of vertexes along the path.

5) LOAD OF SERVICE

Having the detour ratio and average number of on-board riders defined, the *load of service* of vehicle v is defined as follows:

$$ls(v) = \alpha \frac{\varphi(path)}{c_v} + (1 - \alpha)\frac{\zeta(R_f)}{\theta} \tag{17}$$

where c_v is the capacity of vehicle v, R_f is the set of requests being served by f, θ is the system constraint of detour ratio for all taxis, and $\alpha \in [0, 1]$ is the balance factor. The load of service for a fog node f is defined as the average of the top-K vehicles:

$$ls(f) = \frac{\sum_{v \in C_f^K} ls(v)}{|C_f^K|} \tag{18}$$

where $C_f^K \subseteq V_f$ is the set of top-K vehicles ordered by ls in the service area of f. K is a system defined parameter that could be tuned by the operator. The ls values of the fog nodes would be routed to the cloud, so the cloud has knowledge of the load of service statuses of all the fog nodes.

B. DISPATCH OF REQUESTS

The cloud receives requests from riders and dispatches them to fog nodes for further processing. Given a request, e.g. $r(t_0, o, d, [t_1, t_2])$, the cloud first calculates a set of fog nodes that are within the neighbourhood of r, denoted by $N(r)$:

$$N(r) = \{f \mid f \in F \wedge dist(o, l_f) < \delta\} \tag{19}$$

where F is the set of fog nodes, $dist(o, l_f)$ is the distance between o and the location of f, and δ is a predefined parameter. Set $N(r)$ is then ordered by the ls values in ascending order, and the top K' fog nodes are selected as the targeted fog nodes, to which the request would be routed. Here K' is a system defined parameter.

C. SELECTION OF PATH OF MINIMAL COST

As mentioned previously, the demand-responsive transit service could be abstracted as a special member of the general class of the Dial-a-Ride Problem that focuses on scenarios of planning schedules for vehicles, subject to the time constraints on pickup and delivery events. Since the problem is NP-hard [3], [14], here we present a heuristic request insertion and route scheduling algorithm.

Assuming at a time point there are m PVs in the service area of a fog node, correspondingly there are m paths of the vehicles. When a fog node receives a request, it calculates the inserting cost for all the paths, and selects a vehicle with the lowest cost.

In the following we assume the request being processed at the cloud is $r(t_0, o, d, [t_1, t_2])$. Given a *path* $\{x_0, x_1, x_2, \ldots, x_k\}$, the cost of inserting r is defined as:

$$\begin{aligned} insert\_cost(path, r, t_0) &= min\{\pi(path, r, i, j, t_0) \\ &\qquad \mid i, j \in [0, k],\ i \le j\} \\ \pi(path, r, i, j, t_0) &= cost(\{x_0, \ldots, x_i, o, \ldots, x_j, d, \\ &\qquad \ldots, x_k\}, t_0) - cost(path, t_0) \end{aligned} \tag{20}$$

where $cost(path, t_0)$ is the cost of the *path* starting at time t_0, $\pi(path, r, i, j, t_0)$ is extra cost of the *path* after inserting o at x_i and inserting d at x_j. The insertion should also satisfy the time constraint of the request. If the constraints are not satisfied, the insertion cost $\pi(path, r, i, j, t_0) = \infty$. A path is said to be *compatible* to a request if $insert\_cost(path, r, t_0) \neq \infty$, which means r could be inserted to the *path* without violating any constraints. To insert a new request, we could select one location x_i on the path to insert o, then after the location o select another location to insert d. There are $k + 1$ possible insertion positions, so the overall complexity of the insertion is k^2. The insertion algorithm is similar to [5], and we suggest interested readers to the reference for a detailed description of the algorithm. Note that the insertion points of a path is smaller than twice of the capacity of a vehicle, i.e. $k < 2c_v$.

Then the vehicle with the minimal insertion cost is selected as candidate vehicle (cv) to serve this request, denoted as:

$$cv = \underset{v}{\operatorname{argmin}}\{insert\_cost(v.path, r) \mid v.path \in PATH\} \tag{21}$$

where $PATH$ is the set of paths in fog f, and $v.path$ is the current path of vehicle v. Having the candidate PV calculated, the insertion cost is defined as:

$$icost(f, r) = insert\_cost(cv.path, r) \tag{22}$$

D. FINAL DECISION OF A REQUEST

Each fog node in set $N(r)$ would then route a message, e.g. *msg*, that contains the value of *icost* to a mediate node to make final decisions about the request. The mediate node could be the cloud, or the fog node that covers the origin of the request if fog nodes are inter-connected. Message *msg* is in the form $\langle id, icost(f, r), cv, ls(cv)\rangle$, where *id* indicates the fog node, cv is the candidate PV, and $ls(cv)$ is cv's value of load of service. When the mediate node receives all the messages, it compares all the values of insertion cost and finds the minimal one.

Suppose $\langle f^*, icost^*(f^*, r), cv^*, ls(cv^*)\rangle$ is the message with the smallest insertion cost for r, there are two cases of the value $icost^*(f^*, r)$:

1) If $icost^*(f^*, r) = \infty$, r is rejected. This means the insertion violates some constraints. A rejection message $reject(r)$ is routed to the cloud, and then sent to inform the user/rider.
2) If $icost^*(f^*, r) \neq \infty$, r is accepted. Public vehicle cv^* at fog f^* is assigned to r, and an assignment message (r, cv^*, f^*) is routed to fog f^* and the cloud. An acceptance message $accept(r)$ is also sent from the cloud to the rider to notify him/her to wait for the pickup. Fog node f^* would broadcast the assignment message, and vehicle cv^* would update its current path to pickup the rider of r when receiving the message.

If there are multiple fog nodes that have the same insertion cost $icost^*(r, f^*)$, the one with the smallest value of load of service $ls(cv^*)$ will be selected as the final match to the request. In very rare cases that even the $ls(cv^*)$ values are equal, FPVS will just select one fog and one PV in random for the match assignment.

Algorithm 1 Message Handling When Dispatching Requests and Making Request-Vehicle Matching

```
1  if receives r at the Cloud then
2      get neighbouring fogs N(r) according to Eq. 19;
3      Ω ← order N(r) by the ls values;
4      Ω(K') ← top K' fog nodes in Ω;
5      for each f ∈ Ω(K') do
6          send r to fog node f;
7  if receives r at fog node v then
8      calculate icost(f, r) according to Eq. 20, 21, 22;
9      msg ← ⟨id, icost(f, r), cv, ls(cv)⟩;
10     send msg to the cloud;
11 if receives msg at the cloud then
12     ⟨id, icost(f, r), cv, ls(cv)⟩ ← msg;
13     add ⟨id, icost(f, r), cv, ls(cv)⟩ to Ω';
14     if |Ω'| == |Ω(K')| then
15         icost*(f*, r) ← smallest insertion cost in Ω';
16         if icost*(f*, r) == ∞ then
17             send reject(r) to rider of r;
18         else
19             send accept(cv*, r) to rider of r;
20             send accept(cv*, r) to f*;
21 if receive accept(cv*, r) at fog node f* then
22     broadcast assign(r, cv*) at f*;
23 if receive assign(r, cv*) at cv* then
24     insert r into the current path of cv*;
25     reschedule the path to pickup r;
```

E. Algorithm Description

Algorithm 1 is the pseudocode of message handling when dispatching requests and making request-vehicle matching.

When the cloud receives request r, it gets the set of neighbouring fog nodes $N(r)$ according to Eq. 19 (line 2), orders the set by the value of load of service, and selects the top K' fog nodes, denoted by $\Omega(K')$ (lines 3-4). Then the cloud forwards r to each fog node in $\Omega(K')$ (line 5). This request is then received at fog nodes. The fog node, e.g. f, calculates the minimal insertion cost $icost(f, r)$ according to Eq. 20, 21, 22 (line 8). A message that wraps $icost(f, r)$, cv, and $ls(cv)$ is sent back to the cloud (lines 9-10). Here the cloud is used as the intermediate node. It receives the msg message, extracts the data segments related to r, and accumulates these data to a set denoted by Ω' (lines 12-13). When all the messages from nodes in $\Omega(K')$ are received, i.e. $|\Omega'|$ equals $|\Omega(K')|$, the cloud compares all the values of insertion cost and finds the minimal one, i.e. $icost^*(f^*, r)$ (line 15). If the minimal cost is ∞, the request is rejected and a rejection message is sent to the rider of the request (line 17); otherwise, the acceptance message $accept(cv^*, r)$ is sent to both the rider and fog node f^* (lines 19-20). The vehicle cv^* is in the service area of fog node f^*, and it is going to be scheduled to pick up the rider of the request. Finally, when fog node f^* receives the $accept(cv^*, r)$ message, it broadcasts a message $assign(r, cv^*)$ within its coverage (line 22). When cv^* receives the message, it would insert r into the current path and reschedule its path to pick up the rider of r (lines 24-25).

VI. MAINTENANCE OF METADATA

The proposed scheme is a distributed framework that integrates vehicular nodes, fog nodes, and the cloud. The vehicular sensed data and metadata defined in section IV-B are maintained in a cooperative manner among these nodes. In this section we first introduce types of the metadata, then describe a passive delay-tolerant approach for metadata upload, and briefly discuss the issue of handing over PVs among fog nodes.

A. TYPES OF METADATA

Generally, there are roughly three types of metadata in the FPVS.

1) Metadata of vehicles: include the current path, the set of requests, the remaining seats of the vehicle, and etc. These data are generated when PVs or passengers join in or leave the system. Given these data, the load of service within the service area could be estimated.
2) Metadata of traffic condition: include the network of road, the speeds of road segments, traffic lights, and realtime traffic conditions such as accidents, and etc. Some cost calculations, e.g. speeds of road segments, are done locally at vehicular nodes and then uploaded to the fog node. The travelling cost could be estimated in real time based on these data.
3) Metadata of ride demand: include the time, the origin and destinations of trip requests. Given these data, i.e. the accumulated historical dataset, the pattern of trips within the area could be estimated. Several recent research have addressed this issue within a centralised framework [31], [32].

In this research we mainly utilise the first two types of metadata, and assume the ride demand, i.e. the third type of metadata, is given at real time. The metadata are maintained and updated, a large part of which are through the V2V and V2I communications. However, due to the sparseness of some suburban areas and the lack of enough fog nodes, it is not uncommon that some regions in the service area of a fog might not be covered by be communication range of the fog. In these scenarios, vehicles would forward their metadata to their neighboring nodes using the classic V2V protocols, such as MaxProp [33], CBF [34] and TO-GO [35], and then upload the data through V2I communications.

B. PASSIVE DELAY-TOLERANTE METADATA UPLOAD

FPVS adopts a passive delay-tolerant strategy to diffuse and upload metadata. If a vehicular node, e.g. v, is not within the coverage of any fog node, it would calculate an expected length of time interval to connect to a fog node, denoted by β:

$$\beta = cost(l_v, b_f) \tag{23}$$

where l_v is the current location of v, b_f is the border of the coverage area of fog node f, and $cost(l_v, b_f)$ is the estimated cost of time for v to travel to the coverage area of f. As depicted in Fig. 2, each vehicular node has a time-dependent road network stored in its local storage, so it is able to estimate the travelling cost $cost(l_v, b_f)$. Here we omit the detailed descriptions of the trivial calculations. It is easy to see that β is also the expected length of time interval for v to upload its metadata through the V2I communications.

A message that wraps the metadata of a vehicular node v is denoted by $msg(t_0, \eta, v, data)$, where t_0 is the timestamp when the message is generated, η is an interval of delay within which the message should be uploaded to fog or the cloud before $t_0 + \eta$, $data$ is the detailed metadata. According to the current time now and the length of time interval β, the vehicular node v, which is not covered by any fog nodes, would adopt different strategies to diffuse msg:

1) $now + \beta \leq t_0 + \eta$. It means node v is able to enter the coverage of a fog node before the deadline of metadata uploading. Therefore, the metadata is just carried at v.
2) $now + \beta > t_0 + \eta$ and node v encounters a node x that satisfies $now + x.\beta \leq t_0 + \eta$. It means x is going to enter the coverage of a fog before the deadline of msg. In this case, msg is forwarded to node x, and x is responsible to upload the message, or re-forward it to other nodes.
3) $(t_0 + \eta) - (now + \beta) < \epsilon$, where $\epsilon > 0$ is a predefined small value of time interval. It means the deadline of the message upload is approaching. In this case, the message is uploaded through the 4G channel, where the metadata are first routed to the cloud, and then forwarded to the fog that vehicles currently belong to.

Algorithm 2 the is pseudocode of message handling for the metadata upload at vehicular nodes. Lines 3-7 correspond to strategy 2, lines 8-12 correspond to strategy 3. Note that in line 8, the node monitors all the messages in its storage by periodically checking. The *check_time_unit* is a system defined time interval for checking messages that are close to their deadlines of upload. Those messages are to be sent to the cloud by 4G, and then forwarded to the fog nodes that vehicles currently belong to. Similarly, the fog is able to route messages to PVs either through the V2I/V2V communications, or through the cloud as an intermediate node.

Algorithm 2 Messages Handling of Metadata Upload at Vehicular Node v

1 **if** *generates or receives msg* **then**
2   insert *msg* local storage;
3 **if** *v encounters node x* **then**
4   **for** *all msg in local storage* **do**
5   **if** $now + \beta > msg.t_1$ *and* $now + x.\beta \leq msg.t_1$ **then**
6   send *msg* to node x;
7   remove *msg* from v;
8 **for** *every check_time_unit* **do**
9   **for** *all msg in local storage* **do**
10   **if** $msg.t_1 - (now + \beta) < \epsilon$ **then**
11   send *msg* to the cloud through 4G;
12   remove *msg* from v's storage;

C. HAND OVER AMONG FOG NODES

When a vehicle moves along its path, it transits from one fog's service area to another. Here we assume seamless scheme is used for the handover [36], and the IEEE 802.11p-based wireless access system could take advantage of the fixed-order placement of the RSUs and unidirectional movement of the vehicles along the highways. When the handover is done, the vehicle is assigned to a new fog node, and its metadata are sent to the new fog node and reside in that node.

VII. PERFORMANCE EVALUATION

A. ENVIRONMENTAL SETUP

We conduct experiments on the ONE platform [37] with real-world road network to verify the performance of the proposed FPVS scheme. The ONE is a popular simulation environment that is capable of generating node movement using different movement models and routing messages between nodes with various routing algorithms. The Xiamen Taxi Dataset [38] is used for the simulation. The dataset consists of the position trajectories that records the position of taxis and the operation trajectories that records the journey of taxis. The one-month trajectory dataset is of about 5000 taxicabs in Xiamen city, China during July 2014, totally about 220 million GPS position records and 8 million live trips. On the time aspect, the dataset covered all the daytime and nighttime on both weekdays and weekends, being able to disclose mobility patterns in heavy, moderate, and light traffic conditions. Fig. 4 shows the hourly average number of requests in Xiamen Island, Xiamen City on 16-17 th, July, 2014. Requests are

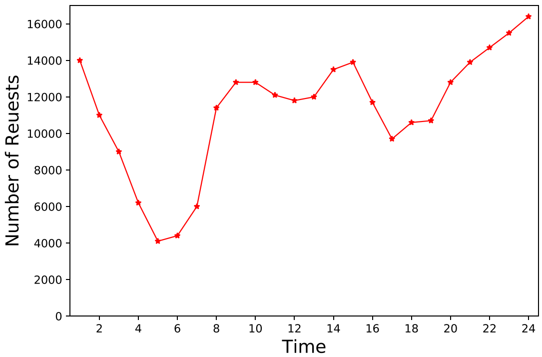


FIGURE 4. Hourly average number of requests in Xiamen Island, Xiamen City on 16-17 th, July, 2014.

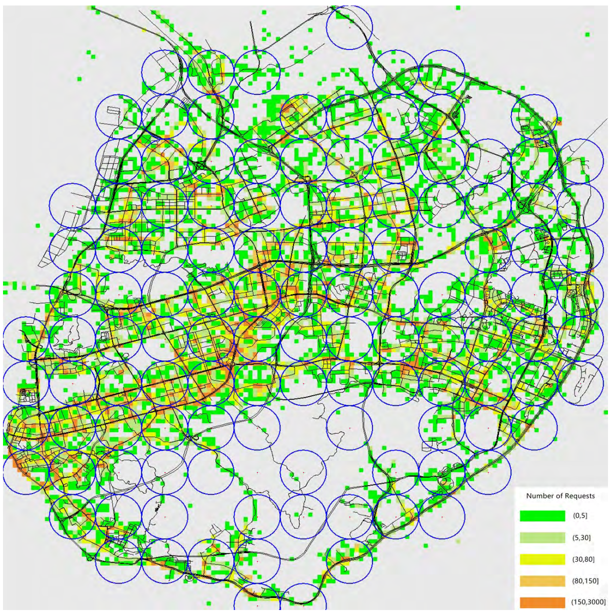


FIGURE 5. Accumulated number of requests in Xiamen Island, Xiamen City on 16-17 th, July, 2014. Requests are mapped to grids, and each grid has a width of 0.001 degree of longitude and a length of 0.001 degree of latitude. And 120 RSUs (blue circles) are evenly deployed on the map along the roads.

extracted from the dataset. Suppose the real pickup time of a rider in the dataset is pk, the constraint window of the request is set as $[pt-4, pt+4]$. By default, the threshold for the detour ratio is 1.8, K and K' that select the top K vehicles and the top K' fog nodes according to the load of service are both set 3. The balance factor α in Eq. 17 has relatively good performance at range [0.32,0.70], so it is set 0.5 by default. The communication range of fog nodes δ in Eq. 19 is 450 meters. Fig. 5 depicts the distribution of the requests based on grids. Each grid has a width of 0.001 degree of longitude and a length of 0.001 degree of latitude, and the number of requests that belong to grids and the locations of RSUs are showed. And 120 RSUs (blue circles) are evenly deployed on the map along the roads. The communication range of I2I or I2V used by the vehicles to exchange data is set to 300 meters, while the 4G channel does not have the limit of communication range.

B. OVERALL PERFORMANCE

The service ratio, waiting time and share factor are used to measure the overall performance of the schemes. The service ratio is defined as the number of successful picked up requests divided by the total number of requests. The share factor is defined similar to [5]: $\frac{\sum_r d_r}{d_V}$, where d_r is the distance that a request r travels, and d_V is the total distance that all the vehicles travel.

TABLE 1. Overall performance of the schemes.

Scheme	Number of Vehicles	Service Ratio	Waiting Time (min)	Share Factor
Taxi	500	0.1131	3.32	0.874
Taxi	1000	0.2498	3.59	0.850
FPVS	200	**0.2137**	**6.45**	**5.675**
FPVS	400	**0.4038**	**6.32**	**5.322**

Table 1 compares the performance of the proposed pubic vehicle system with the taxi system. A taxi is assumed to be occupied by one rider, yet a PV can share and have as many as 10 riders. The simulation begins at the peak period of 9 pm to 12 pm, and has about 4.66×10^4 requests. From Table 1 we can see that the service ratio of FPVS is about 0.2137 when 200 PVs are deployed, which is very close to that of Taxis when 1000 vehicles are deployed. This is because PVs are shared among riders, and they have an average share factor as high as 5.322. The share factor of the Taxi scheme is lower than 1.0, i.e. about 0.85, which reflects the deadhead distances when no riders are on board. So FPVS is able to achieve the same service ratio with much fewer vehicles than Taxi. When about 400 PVs are deployed, about 40 percent of the requests are picked up successfully. Also, the waiting time in the FPVS is about 6.3 minutes, which is about 3 minutes more than that of the Taxi scheme.

C. FOG-BASED VEHICULAR SENSING

To verify the effectiveness of vehicular sensing and local processing, we also conducted three other schemes: 1) Central-Raw: vehicles connect to the cloud. All raw vehicular sensing data are routed to the cloud by the 4G channel; so do the requests and commands; 2) Central: similar to Central-Raw, but the sensed data is processed at local nodes. Only the aggregated metadata are uploaded to the cloud; 3) FPVS-Raw: similar to FPVS, vehicles connect to fog nodes and the cloud. But all raw vehicular sensing data are routed to the fog through V2V/V2V communications. All the 4919 taxis in the dataset are involved for the sensing and metadata gathering. The bandwidth of the 4G channel is set 20 Mbps/5 Mbps for the down/up links, the bandwidth of the V2V or V2I channel is 500 Kbps/250 Kbps for the down/up links. Ideal links are assumed when two nodes meet and establish a connection, and the requests, commends and metadata could be wrapped into one message respectively.

Fig. 6 compares the number of message transmissions of various schemes. The numbers of message transmissions of the Central and FPVS are much smaller than those of

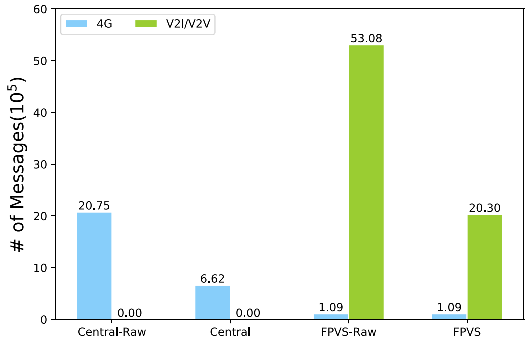


FIGURE 6. Comparison of message transmissions.

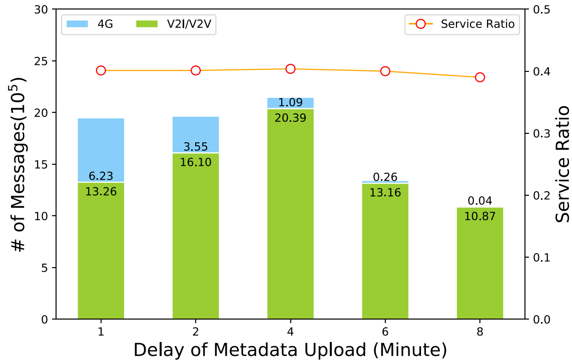


FIGURE 8. Impact of the delay of metadata upload.

the Central-Raw and FPVS-Raw, and more than 60 percent of messages are saved, which indicates the effectiveness of local processing. Vehicular nodes do some local processing and reduce the size of the metadata to be uploaded. Also, the number of total transmissions in FPVS is more than 2.5 times larger than that of the Central schemes. This is due to the infectious nature of V2V/V2I communications where messages are forwarded among encountered nodes to be uploaded. Yet according to [39], the cost of V2V/V2I communications would be much smaller than the 4G channel that is operated by centralized telecommunications companies. Although FPVS has larger message transmissions, its cost is actually lower because the price of 4G is higher than that of V2V/V2I. In this research, the cost of the 4G channel is set 10^{-2} \$/MB and the V2V/V2I channel is set 10^{-4} \$/MB. From Fig. 7 we can see that the Central-Raw scheme incurs the largest cost of 10.11\$, and FPVS has the least cost of 0.93 \$, which indicates the potential cost reduction of local processing at vehicular sensing applications.

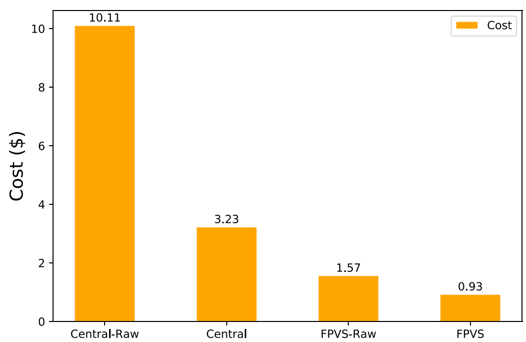


FIGURE 7. Cost of transmissions.

We also studied the impact of η, which is an interval of delay within which the message should be uploaded to fog or the cloud. As depicted in Fig. 8, the number of V2I/V2V messages increases, and the number of 4G messages decreases when η increases from 1 minute to 4 minutes. This is because larger delay means vehicular nodes have more time before the deadline of upload, and do not need to rely on the 4G channel for metadata gathering. Instead, nodes carry the metadata and forward the metadata messages when encountering vehicular nodes, and finally upload the data to the fog through V2I transmissions. However, when the delay grows larger, i.e. 6 or 8 minutes, the vehicular nodes have large probability to enter the coverage of a fog node, so they just carry the metadata till they re-connect to fog nodes. Forwarding messages to other vehicular nodes is suppressed in these scenarios, so the number of V2I/V2V messages decreases. Furthermore, when the upload delay is larger, the node could wrap multiple data readings into a message and upload it to the fog, which reduce the number of message transmissions. Fig. 8 also shows that delay has relatively little impact on the service ratio, which decreases from 0.4038 to 0.3902 when η increases from 4 to 8 minutes. As the cost of 4G is assumed to be more expensive than that of the V2I/V2V communications, the delay of metadata upload could be tuned to achieve a relatively low transmission cost and a high service ratio.

D. IMPACT OF DETOUR RATIO THRESHOLD AND PICKUP TIME WINDOW

The detour ratio and pickup time window reflect riders' requirements, and they could be tuned to fit the real scenarios by the system operator. In the experiment, we also varied their values to study their impact on the performance. As illustrated in Fig. 9, the service ratio and share factor increase with the threshold of the detour ratio. The service ratio increases from 0.25 to 0.46 when the detour ratio grows from 1.2 to 2.0. More requests are accepted without violating the detour constraints when the threshold is set higher, so both the service ratio and share factor increase. Fig. 10 shows the service ratio and waiting time increase with the size of the pickup time window. The service ratio increased from 0.31 to 0.44 when the size of the pickup window grows from 2 to 10 minutes. This also means the rider would have to wait longer before being picked up, which is also showed in the figure. So increasing the detour ratio threshold and the size of pickup time window result a higher service ratio, yet it is also at the cost of riders' comfort.

E. IMPACT OF NUMBER AND CAPACITY OF PVs

Fig. 11 shows the impact of the number of PVs. It is clear that when more PVs are deployed to the road network, more requests could be satisfied, and with smaller delay. We see

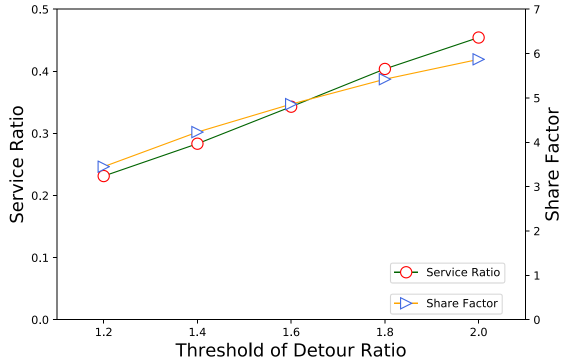


FIGURE 9. Impact of detour ratio threshold.

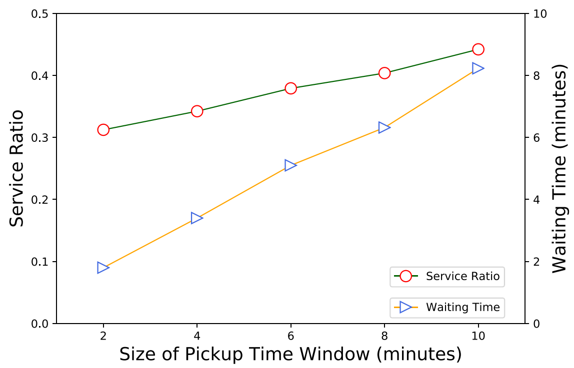


FIGURE 10. Impact of pickup time window.

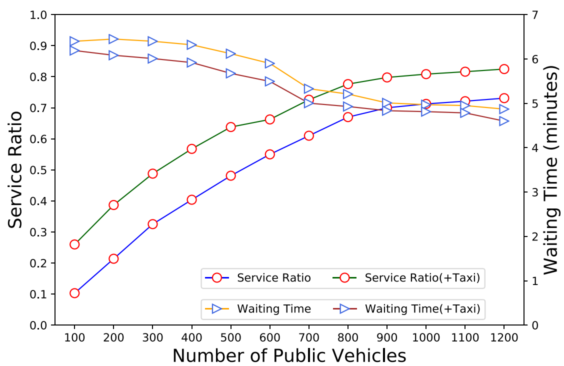


FIGURE 11. Impact of number of PVs.

a gradual increase of the service ratio and a decrease of the waiting time as the number of vehicles grows from 100 to around 900. However, more PVs, e.g. more than 900, have not much impact to increase the service ratio and waiting time. The service ratio is about 0.71 and the waiting time is about 4.9 minutes when more than 900 PVs are deployed. This is largely due to the the constraint time window and threshold of detour ratio, where requests that do not satisfy the constraints would be rejected, leading to failures of service. Besides the deployed PVs, we also add 800 taxis to analysize the overall performance. As showed in Fig. 11, FPVS plus taxis has better performance in both the service ratio and the waiting time. It achieves more than 80 percentage of service ratio when extra taxis are deployed and the number of PVs are more than 900. The waiting time also decreases about 0.4 minutes. This indicates that combining the PVs and taxis is an feasible solution to satisfy the demands of the riders. The proposed scheme achieves better transportation efficiency due to the sharing nature of PVs.

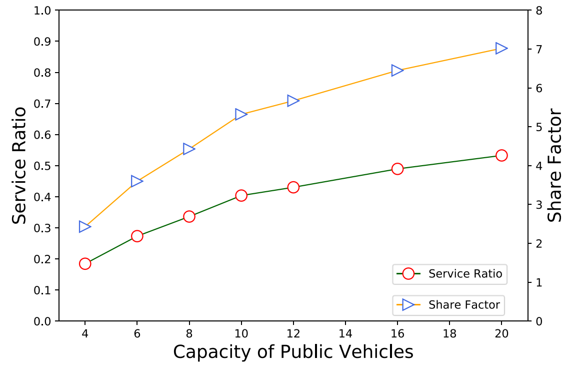


FIGURE 12. Impact of capacity of PVs.

Fig. 12 shows the impact of the capacity of PVs. It is clear that larger capacity means holding more riders at the same time, so both the service ratio and share factor will grow with the capacity of vehicles. Specially, the service factor increases from 0.18 to 0.53, the share factor increases from 2.4 to 7.0 when the capacity increases from 4 to 20. Yet when the capacity is relatively large, e.g. greater than 10, the increase becomes smooth. The share factor is about 6.45 when the capacity is 18, and 7.0 when the capacity is 20. This indicates there are some limits when using larger PVs to pickup the riders. On the contrary, PVs that have small or median capacity are more flexible to pickup riders and it would achieve higher service ratio if more vehicles are deployed.

VIII. CONCLUSIONS

In this paper, we have proposed a distributed public vehicle scheduling system that integrates fog nodes and vehicular sensing. The proposed FPVS includes components of metadata gathering, cost estimation, request answering and PVs scheduling. Fog nodes act as intermediate nodes to store and gather metadata, which are sensed and extracted by vehicular nodes. And the system adopts a heuristic request insertion algorithm, as well as a cooperative strategy among vehicles nodes, fog nodes, and the cloud to dispatch requests and to schedule routes for PVs. Experimental studies in real-world datasets demonstrate that FPVS achieves higher service ratio of requests and better efficiency than other transportation methods, and the distributed vehicular sensing is capable of collecting feasible metadata for scheduling applications. To the best of our knowledge, this research is the first step of integrating fog nodes and vehicular sensing for request responsive scheduling systems.

The integration between the RDRS system and VANETs is attracting more and more attention in the research community. For future work, we are going to further investigate the cooperation strategies among the cloud, fog, and the vehicular nodes; and we plan to study the optimized placement and coverage of fog nodes within the proposed FPVS framework.

REFERENCES

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

[2] 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.

[3] A. Attanasio, J.-F. Cordeau, G. Ghiani, and G. Laporte, "Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem," *Parallel Comput.*, vol. 30, no. 3, pp. 377–387, Mar. 2004.

[4] P. Cheng, H. Xin, and L. Chen, "Utility-aware ridesharing on road networks," in *Proc. ACM Int. Conf. Manage. Data*, 2017, pp. 1197–1210.

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

[6] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, "A comprehensive survey on vehicular Ad Hoc network," *J. Netw. Comput. Appl.*, vol. 37, pp. 380–392, Jan. 2014.

[7] A. Dua, N. Kumar, and S. Bawa, "A systematic review on routing protocols for vehicular ad hoc networks," *Veh. Commun.*, vol. 1, no. 1, pp. 33–52, Jan. 2014.

[8] Y. Xu, X. Chen, A. Liu, and C. Hu, "A latency and coverage optimized data collection scheme for smart cities based on vehicular ad-hoc networks," *Sensors*, vol. 17, no. 4, p. 888, 2017.

[9] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an international standard for wireless access in vehicular environments," in *Proc. IEEE Veh. Technol. Conf. (VTC Spring)*, May 2008, pp. 2036–2040.

[10] Y. Lai *et al.*, "Fog-based two-phase event monitoring and data gathering in vehicular sensor networks," *Sensors*, vol. 18, no. 1, p. 82, 2018.

[11] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, Aug. 2012, pp. 13–16.

[12] J. Zeng, T. Wang, Y. Lai, J. Liang, and H. Chen, "Data delivery from WSNs to cloud based on a fog structure," in *Proc. 4th IEEE Int. Conf. Adv. Cloud Big Data*, vol. 3. Aug. 2016, pp. 959–973.

[13] Intel. (2017). *Next Unit of Computing*. [Online]. Available: https://www.intel.com/content/www/us/en /products/boards-kits/nuc.html

[14] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: Models and algorithms," *Ann. Oper. Res.*, vol. 153, no. 1, pp. 29–46, 2007.

[15] Z. Hao, E. Novak, S. Yi, and Q. Li, "Challenges and software architecture for fog computing," *IEEE Internet Comput.*, vol. 21, no. 2, pp. 44–53, Mar./Apr. 2017.

[16] J. Xu, X. Liu, M. Ma, A. Liu, T. Wang, and C. Huang, "Intelligent aggregation based on content routing scheme for cloud computing," *Symmetry*, vol. 9, no. 1, p. 221, 2017.

[17] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proc. Nat. Acad. Sci. USA*, vol. 111, no. 37, pp. 13290–13294, 2014.

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

[19] M. Zhu, X.-Y. Liu, and X. Wang. (2017). "An online ride-sharing path planning strategy for public vehicle systems." [Online]. Available: https://arxiv.org/abs/1712.09356

[20] M. Zhu, X.-Y. Liu, and X. Wang. (2017). "Joint transportation and charging scheduling in public vehicle systems—A game theoretic approach." [Online]. Available: https://arxiv.org/abs/1712.07947

[21] U. Lee, B. Zhou, M. Gerla, E. Magistretti, P. Bellavista, and A. Corradi, "Mobeyes: Smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Commun.*, vol. 13, no. 5, pp. 52–57, Oct. 2006.

[22] B. Hull *et al.*, "CarTel: A distributed mobile sensor computing system," in *Proc. 4th Int. Conf. Embedded Netw. Sensor Syst.*, 2006, pp. 125–138.

[23] T. Delot, N. Mitton, S. Ilarri, and T. Hien, "Decentralized pull-based information gathering in vehicular networks using GeoVanet," in *Proc. IEEE 12th Int. Conf. Mobile Data Manage.*, vol. 1. Jun. 2011, pp. 174–183.

[24] B. Płaczek, "Selective data collection in vehicular networks for traffic control applications," *Transp. Res. C, Emerg. Technol.*, vol. 23, pp. 14–28, Aug. 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0968090X1100180X

[25] A. Skordylis and N. Trigoni, "Efficient data propagation in traffic-monitoring vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 680–694, Sep. 2011.

[26] X. Li, H. Huang, X. Yu, W. Shu, M. Li, and M.-Y. Wu, "A new paradigm for urban surveillance with vehicular sensor networks," *Comput. Commun.*, vol. 34, no. 10, pp. 1159–1168, Jul. 2011.

[27] S. K. Sharma and X. Wang, "Live data analytics with collaborative edge and cloud processing in wireless IoT networks," *IEEE Access*, vol. 5, pp. 4621–4635, 2017.

[28] M. Eltoweissy, S. Olariu, and M. Younis, "Towards autonomous vehicular clouds," in *Proc. Int. Conf. Ad Hoc Netw.*, 2010, pp. 1–16.

[29] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.

[30] M. T. Asif *et al.*, "Spatiotemporal patterns in large-scale traffic speed prediction," *IEEE Transp. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 794–804, Feb. 2014.

[31] Y. Li, Y. Zheng, H. Zhang, and L. Chen, "Traffic prediction in a bike-sharing system," in *Proc. 23rd SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2015, p. 33.

[32] Y. Tong *et al.*, "The simpler the better: A unified approach to predicting original taxi demands based on large-scale online platforms," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1653–1662.

[33] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "MaxProp: Routing for vehicle-based disruption-tolerant networks," in *Proc. INFOCOM*, Apr. 2006, pp. 1–11.

[34] H. Füßler, J. Widmer, M. Käsemann, M. Mauve, and H. Hartenstein, "Contention-based forwarding for mobile ad hoc networks," *Ad Hoc Netw.*, vol. 1, no. 4, pp. 351–369, Nov. 2003.

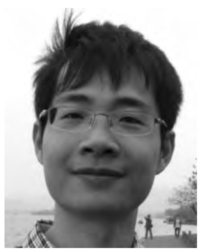
[35] K. C. Lee, U. Lee, and M. Gerla, "Geo-opportunistic routing for vehicular networks [topics in automotive networking]," *IEEE Commun. Mag.*, vol. 48, no. 5, pp. 164–170, May 2010.

[36] J. Choi and H. Lee, "Supporting handover in an IEEE 802.11p-based wireless access system," in *Proc. 7th ACM Int. Workshop Veh. InterNETworking (VANET)*, New York, NY, USA, 2010, pp. 75–80. [Online]. Available: http://doi.acm.org/10.1145/1860058.1860073

[37] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE simulator for DTN protocol evaluation," in *Proc. 2nd Int. Conf. Simulation Tools Techn. (SIMUTools)*, New York, NY, USA, 2009, Art. no. 55.

[38] Z. Lyu, Y. Lai, K.-C. Li, F. Yang, M. Liao, and X. Gao, "Taxi route recommendation based on urban traffic Coulomb's law," in *Proc. Int. Conf. Web Inf. Syst. Eng.*, Oct. 2017, pp. 376–390.

[39] C. D. Bazelon and L. Figurelli, "The economic costs and benefits of a federal mandate that all light vehicles employ 5.9 GHz DSRC technology," Nat. Cable Telecommun. Assoc., Tech. Rep. 7292, 2016. [Online]. Available: http://files.brattle.com/files/7292_brattle_costs_benefits_of_v2v_mandate_may_2_2016.pdf



YONGXUAN LAI received the Ph.D. degree in computer science from the Renmin University of China in 2009. He is currently an Associate Professor with the Software School, Shenzhen Research Institute, Xiamen University, China. He is an Academic Visiting Scholar with the Data and Knowledge Engineering Group, University of Queensland, Australia, from 2017 to 2018. His research interests include network data management, vehicular ad hoc networks, big data management and analysis.

FAN YANG received the Ph.D. degree in control theory and control engineering from Xiamen University, Xiamen, China, in 2009. He is currently an Associate Professor with the Department of Automation, Institute of Pattern Recognition and Intelligent Systems, Xiamen University. He has authored over 50 journal articles and conference papers. His current research interests include mobile computing, pattern recognition, data mining, and bioinformatics.

LU ZHANG received the B.S. degree from the Tianjin University of Finance and Economics, China, in 2016. She is currently pursuing the master's degree with the Software School, Xiamen University, China. Her research interests include wireless sensor networks and fog computing.

ZIYU LIN received the Ph.D. degree in computer science from Peking University, China, in 2009. He is currently an Assistant Professor with the Computer Science Department, Xiamen University, China. His research interests include mobile computing, bigdata, and database.