IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 18, 2018, accepted October 28, 2018, date of publication November 5, 2018, date of current version November 30, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2879578

Parked Vehicle Edge Computing: Exploiting Opportunistic Resources for Distributed Mobile Applications

XUMIN HUANG[1], RONG YU[1], (Member, IEEE), JIANQI LIU[1,2], (Member, IEEE), AND LEI SHU[3,4], (Senior Member, IEEE)

[1]School of Automation, Guangdong University of Technology, Guangzhou 510006, China
[2]Guangdong Mechanical & Electrical College, Guangzhou 510006, China
[3]College of Engineering, Nanjing Agricultural University, Nanjing 210095, China
[4]School of Engineering, University of Lincoln, Lincoln LN6 7TS, U.K.

Corresponding author: Rong Yu (yurong@ieee.org)

The work was supported in part by the programs of NSFC under Grant 61422201, Grant 61370159, Grant 61503083, Grant 61701122, and Grant U1301255, in part by the Science and Technology Program of Guangdong Province under Grant 2015B010129001 and Grant 2016A030313734, and in part by the Science and Technology Program of Guangzhou, China, under Grant 201804010238.

ABSTRACT Vehicular Edge Computing (VEC) has been studied as an important application of mobile edge computing in vehicular networks. Usually, the generalization of VEC involves large-scale deployment of dedicated servers, which will cause tremendous economic expense. We also observe that the Parked Vehicles (PVs) in addition to mobile vehicles have rich and underutilized resources for task execution in vehicular networks. Thus, we consider scheduling PVs as available edge computing nodes to execute tasks, and this leads to a new computing paradigm, called by parked vehicle edge computing (PVEC). In this paper, we investigate PVEC and explore opportunistic resources from PVs to run distributed mobile applications. PVs coordinate with VEC servers for collective task execution. First, a system architecture with primary network entities is proposed for enabling PVEC. We also elaborately design an interactive protocol to support mutual communications among them with security guarantee. Moreover, we measure the availability of opportunistic resources and formulate a resource scheduling optimization problem by using Stackelberg game approach. A subgradient-based iterative algorithm is presented to determine workload allocation among PVs and minimize the overall cost of users. Numerical results indicate that compared with existing schemes, PVEC serves more vehicles and reduces service fee for users. We also demonstrate that the Stackelberg game approach is effective and efficient.

INDEX TERMS Edge computing, intelligent vehicles, computational efficiency, resource management.

I. INTRODUCTION

Nowadays, the global number of vehicles has reached 1.2 billion in 2014, and the number is predicted to be 2.0 billion in 2035 [1]. Numerous communication and computation resources are required for future service provision. Besides, an explosive growth of data traffic is generated with strict processing requirements. In compute-intensive applications, e.g., autonomous driving, raw sensor data is generated at 2 Gb/s and should be processed within 1 millisecond. To overcome the dilemma, mobile edge computing has been envisioned as a crucial technology to increase network resources and enable localized data processing by deploying pervasive and proximal cloud computing capability around users. The technology reduces latencies, mitigates communication jitter and realizes mobility support for users [2]. Recently, the technology is introduced into vehicular networks for extending computing environment to the vehicular network edge, leading to Vehicular Edge Computing (VEC), which is a great adoption of mobile edge computing in transportation domains [3].

Existing work have studied VEC in different application scenarios. On one hand, VEC enlarges resource capacity of vehicular networks and brings available computation resources at the network edge. In computation offloading, computation tasks can be directly executed by proximal VEC servers with fast response. With edge deployments, VEC servers have remarkable advantages in localized data

processing and can also undertake related tasks in network management.

However, there exist critical issues regarding VEC should be addressed for large-scale network implementation. The issues consist of the following aspects:

- VEC provides constrained network resources and this cannot adapt to the exponentially increasing number of connected vehicles. More available resources are still required, especially when compute-intensive applications have been developed rapidly.
- The number of dedicated servers in VEC can be promoted gradually. The establishment of VEC servers depends on high virtualization of physical resources in enhanced network routers [4], e.g., roadside unit. It is not feasible to deploy massive VEC servers all over the network for ubiquitous computing environment. This clearly causes large economic and time expense.
- Limited resource capacity always restricts the number of served users. Idle resources from existing network entities can be excavated to improve resource utilization and extend resource capacity of vehicular networks.

We observe that Parked Vehicles (PVs) always have rich embedded computation resources to execute tasks. So we aim to exploit underutilized resources from common PVs for on-demand resource provision.

PVs are of convenient opportunities to offer available resources for running distributed mobile applications. Similar to VEC servers, PVs become available edge computing nodes to serve mobile vehicles. Then we propose a new computing paradigm, called by Parked Vehicle Edge Computing (*PVEC*). In daily life, a high proportion of vehicles are parked on streets, roadways and parking lots. According to a real world urban parking report [5], about 70% of individual vehicles are parked for an average of more than 20 hours every day [6]. In addition, PVs stay in the parked state and they can be scheduled normally for service provision [7], [8]. Here, we are motivated to utilize PVs for task execution in computation offloading, as proposed in [9]. Most of existing work only pay attention to PV scheduling and the collaboration among PVs is optimized according to specified scenarios. But the coordination between dynamic PVs and fixed VEC servers is seldom researched by them. We consider that opportunistic resources extracted from PVs are complementary to regular resources offered by VEC. Two kinds of resources coexist with each other and further extend resource capacity at the vehicular network edge. Thus, PVs are employed to cooperate with VEC servers for collective task execution in PVEC. Owing to ubiquitous resource provision, mobile applications are run in a distributed manner.

In this paper, we investigate PVEC for rapid development of distributed mobile applications and try to answer the following questions: i) Why PVEC is proposed? ii) What is PVEC? iii) How to realize PVEC. PVEC schedules more available resources to facilitate distributed mobile applications with resource guarantee, ultimately serving more users. First, a system architecture with primary network entities is proposed. Second, we elaborately design an interactive protocol to support mutual communications among them for security consideration. Third, we measure the availability of opportunistic resources provisioned by PVs and formulate a resource scheduling optimization problem by using Stackelberg game. Theoretical Stackelberg equilibrium analysis is offered to determine workload allocation among PVs. The goal of the game is to minimize the overall cost for users in computation offloading. Finally, we present a subgradient-based iterative algorithm to achieve the goal.

The main contributions of the paper are summarized as follows.

- We exploit opportunistic resources from idle PVs to realize PVEC. A feasible system architecture including requesting vehicles, service providers with renting VEC servers, and PVs is introduced.
- We design an interactive protocol with basic request and response operations for service provision in PVEC. In particular, the considerate interactive protocol satisfies strict security and privacy requirements.
- We formulate and solve the resource scheduling optimization problem by using Stackelberg game approach. A service provider is a leader while PVs act as followers to response to it. Stackelberg equilibrium analysis with an iterative algorithm is provided to determine workload allocation among PVs, aiming to minimize service fee for a requesting vehicle.

The rest of this paper is organized as follows. Section II presents the related work. We offer the feasible system architecture of PVEC in Section III. The interactive protocol is also presented to support network communications. Section IV offers quantitive representation of opportunistic resource in PVEC. In Section V, the resource scheduling optimization problem is formulated by using Stackelberg game approach. The theoretical analysis with the iterative algorithm is proposed to reach the unique Stackelberg equilibrium. Performance evaluation of our scheme is provided in Section VI. Finally, Section VII concludes this paper.

II. RELATED WORK

A. VEHICULAR EDGE COMPUTING

VEC has drawn a lot of attentions in recent years. Owing to ubiquitous connections, mobile vehicles are convenient to directly upload computation tasks to proximal VEC servers for high-quality computation offloading. VEC servers serve mobile vehicles with overall performance improvements, e.g., lower service delay, reduced bandwidth consumption and enriched awareness support. Zhang *et al.* presented a series of computing offloading schemes to improve task scheduling in VEC environment. In [10], Stackelberg game approach was used to determine which VEC server is to serve a requesting vehicle and accordingly, how many computation resources are allocated to it. Furthermore, a predictive offloading scheme was introduced in [11].

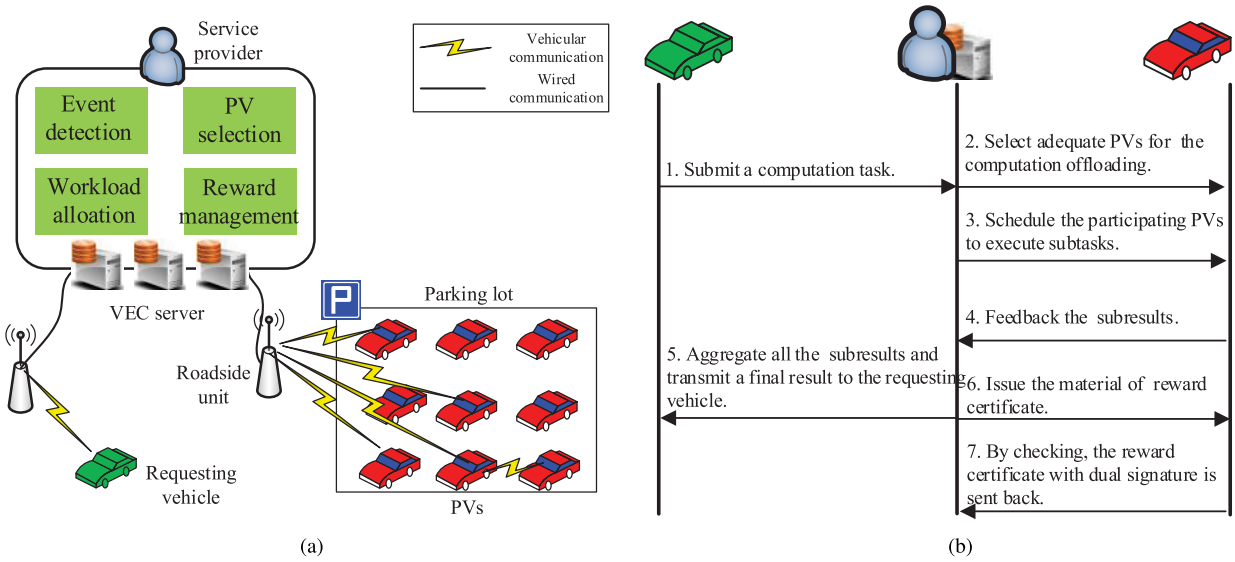


FIGURE 1. Architecture and protocol design for PVEC. (a) System architecture with primary network entities. (b) Interactive protocol for mutual communications.

Both heterogeneous task requirements and mobility of vehicles are considered. For a requesting vehicle, it can offloaded a computation task to a VEC server located at the road segment ahead of the current position.

At the same time, VEC servers are deployed in close proximity to mobile vehicles. VEC servers are capable for localized data processing in vehicular networks. Current researches have proposed that VEC servers can be assigned to undertake different tasks for facilitating distributed network management. For example, VEC servers were utilized to ensure reliable vehicular communications [12], realize distributed reputation management [13] and promote distributed electric vehicle discharging/charging scheduling [14], [15].

Nevertheless, resource capacity of existing VEC still should be further extended to adapt to the exponentially increasing amount of global vehicles and rapid development of compute-intensive applications. In this paper, we consider that opportunistic resources from PVs can be scheduled well for service provision and enhancing resource capacity. Based on the utilization of PV, PVEC is to combine dynamic PVs and fixed VEC servers for collective task execution. As a result, applications are run with resource guarantee in the distributed paradigm and more users are served finally.

B. UTILIZATION OF PVS IN VEHICULAR NETWORKS

In previous work, PV is utilized as a key enabler for supporting diverse vehicular services. Due to ample and underutilized resources, PVs are regarded as a distributed computing, communication and storage asset. According to existing studies, PVs have been used as small-scale data centers [16], and thus computation tasks are offloaded to them for online and off-line execution. Toward inter-vehicles communications, PVs with on-board communication capabilities are relay nodes for packet forwarding [17] and content delivery [18]. Meanwhile, PVs can offer accessible storage capacity to cache vehicular content as scheduled [19]. In the work, PV scheduling with the collaboration among PVs is optimized accordingly.

We are motivated to employ common PVs for assisting in task execution and cooperating with VEC servers in computation offloading. In this way, PVEC realizes that opportunistic resources are formulated and gathered to contribute for extending the resource capacity and improves resource utilization simultaneously. Compared with the above work, our work mainly study the coordination between PVs and VEC servers. Opportunistic resources from PVs are greatly complementary to regular resources residing in VEC. Then we focus on the resource scheduling optimization in PVEC when the opportunistic resources are explored and allocated for satisfying resource requirements of requesting vehicles.

III. ARCHITECTURE AND PROTOCOL OF PVEC

A. SYSTEM ARCHITECTURE OF PVEC

As illustrated in Fig. 1(a), a feasible system architecture with all related network entities is proposed for the implementation of PVEC. In the architecture, the coordination between PVs and VEC servers is particularly considered. This ensures that a service provider can directly host the service at the vehicular network edge. Different workloads from mobile applications are allocated to both PVs and VEC servers. Then mobile applications are run in a distributed manner, and also with reliable resource provision.

With the utilization of VEC servers, decision-making capability of a service provider is shifted to the close proximity of users. A request from a mobile vehicle can be realized and processed promptly in computation offloading. The service provider also schedules VEC servers to

undertake surveillance tasks in a parking lot. PVs are detected and there exists a roadside unit in the parking lot. VEC servers communicate with the roadside unit, which communicates with PVs via existing vehicular communications, i.e., vehicle-to-vehicle and vehicle-to-infrastructure communications. By using the roadside unit as a relay, the service provider interacts with the PVs in real time. The computation task from the requesting vehicle is offloaded to appropriate PVs for execution. Computation offloading is swiftly performed by proximal PVs. Then computation result is finally transmitted to the requesting vehicle.

Here, the requesting vehicle moves on the road and the changing locations are monitored by local roadside units, which collect status information including locations of passing vehicles [20]. By identifying the closest roadside unit of the requesting vehicle at that time, the service provider utilizes it as a relay to feedback the computation result. In short, extra resource provision is offered by PVs in addition to current VEC servers, significantly resulting in extended resource capacity. Ultimately, more available resources are scheduled on demand to support running applications and serve more users.

For service provision, the service provider rents several VEC servers to conduct routine operations. The operations consist of event detection, PV selection, workload allocation and reward management. More specifically, in each resource scheduling time period, the service provider assigns VEC servers to observe PVs with dynamic arrivals and departures in the parking lot. By predicting following parking behavior of the PVs, the service provider handles VEC servers to determine which PVs with opportunistic resources exhibit great serviceability for task execution at this time. Appropriate PVs are selected and employed to serve the requesting vehicle. In particular, the service provider acts as a broker of the requesting vehicle in computation offloading. To improve user satisfaction, the service provider determines how to allocate the total workloads among the selected PVs in an economic way. For stimulations, the service provider gives the participating PVs certain rewards. The participating PVs will receive reward certificates, which record crucial information about task execution, e.g., the amount of workloads undertaken by the PVs and the corresponding rewards. After that, the PVs can exhibit the reward certificates to cash the rewards at any time. More details of the network entities and their functionalities in the architecture of PVEC are described as follows.

- *Requesting vehicle:* Owing to current vehicular communication technologies, the requesting vehicle is convenient to upload its computation task to the nearest roadside unit on the road. The VEC servers wiredly connect to the roadside unit. The roadside unit helps transmit the request to the VEC servers belonging to a service provider. For secure communications, as requesters and participators, both requesting vehicles and PVs interact with the service provider by using anonymous pseudonyms.
- *Service provider:* By renting the VEC servers with edge deployments, the service provider directly process the submitted request for the requesting vehicle in vicinity. The service provider receives the computation task, splits the task into multiple subtasks and employs adequate PVs for execution. When subresults of all the subtasks are gathered, the service provider is also responsible for aggregating the subresults to form a final result. The final result is sent back to the requesting vehicle.
- *VEC server:* VEC servers offer the service provider with managing capability for detecting and selecting PVs, allocating workloads and giving rewards to PVs. In addition, participating PVs could not undertake the whole workloads or continue the allocated workloads. The service provider also assigns VEC servers to undertake the residual workloads when necessary.
- *Parked vehicle:* PVs become main performers in computation offloading. According to previous work, PVs with parked state still contribute their own computing capability and storage space for task execution. By considering the acquired utility, each PV responds to the service provider whether to participate in task execution and the amount of undertaking workloads. With valid efforts, PVs get authentic reward certificates regarding task execution in PVEC and can cash the rewards afterwards.

B. INTERACTIVE PROTOCOL FOR PVEC

During service provision, regular communications among different network entities are necessary. PVs are employed by the service provider to execute tasks in computation offloading. In the meantime, PVs should communicate with the service provider in a considerate way to satisfy security and privacy requirements. Firstly, PVs need to ensure anonymous communications when interacting with the service provider. Thus, others including the service provider cannot know which PVs are actually recruited and given how many rewards for task execution. At the same time, the location privacy of PVs is protected. Secondly, transmitted messages should be encrypted and signed well for secure communications. This aims to defend against tampering attacks and camouflage attacks. Otherwise, a malicious service provider may pretend to recruit several PVs in PVEC but refuse to pay for the PVs afterwards. Last but not least, after the interactive communications, both the service provider and PVs should agree on the fact of task execution regarding workloads, rewards and so on, to defend against reward repudiation of reception and giving.

Based on the requirements, we propose an interactive protocol to support basic request and response operations in PVEC. As shown in Fig. 1(b), there exist some key operations for a requesting vehicle, a service provider and several PVs in the interactive protocol. To summarize, all the related operations are mainly performed in the following phases:

system initialization, request submission, task execution and reward management.

1) SYSTEM INITIALIZATION

For anonymous communications in vehicular networks, each vehicle V is registered to a trusted certification authority and given a set of pseudonyms, which can be regarded as several public keys for signatures and encryption. The pseudonyms are certified by the trusted authority and cannot reveal any information about the identity of the vehicle. For the l-th pseudonym of vehicle V, the certification authority generate a set of key pairs $(PK_V^l, SK_V^l, Cert_V^l)$. PK_V^l and SK_V^l indicate the public key and private key corresponding to the l-th pseudonym while $Cert_V^l$ is a signature of the certification authority on the public key PK_V^l. Here, the vehicle can use the private key SK_V^l to digitally sign messages m when maintaining the l-th pseudonym, namely, carrying out the operation $Sig(SK_V^l, m)$. The vehicle can also use the public key PK_V^l to encrypt the transmitted message m for vehicle V: $E_{PK_V^l}(m)$.

2) REQUEST SUBMISSION

Considering there exists a requesting vehicle i with maintaining its g-th pseudonym, it would like to upload a computation task with input data *data* and task requirements *req*. The vehicle prefers to be served by a service provider *SP*. Then the request is encrypted and sent to the nearest roadside. The operation can be expressed as follows: $i \rightarrow RSU : request = E_{PK_{RSU}}(Sig(SK_i^g, data\,|req)\,|Cert_i^g\,|SP\,|timestamp)$, where PK_{RSU} is the public key of *RSU* and *timestamp* indicates the current time slot.

3) TASK EXECUTION

The request is decrypted by the roadside unit and forwarded to the service provider *SP*, which manipulates PVs in a parking lot. In this time period, *SP* exploits VEC servers to select the PVs with higher serviceability for task execution. The service provider records the request with an identification number *rid* and inquires one selected PV j using h-th pseudonym : $SP \rightarrow j : inquiry = E_{PK_j^h}(\mathrm{Sig}(SK_{SP}, rid\,|par)\,|Cert_{SP}\,|timestamp)$, where *par* is the corresponding parameters of executing the computation task, e.g., the reward of undertaking per workload for any participating PV. Based on the reward parameter and individual workload state, each PV replies with the response *res* about whether to join the task execution and how many workloads for undertaking : $j \rightarrow SP : reply = E_{SP}(\mathrm{Sig}(SK_{PK_j^h}, rid\,|res)\,|Cert_j^h\,|timestamp)$.

By collecting the responses, the service provider divides the whole computation task into a set of subtasks and sends them to the participating PVs (including PV j) for execution: $SP \rightarrow j : execution = E_{PK_j^h}(\mathrm{Sig}(SK_{SP}, rid\,|pid\,|subtask)\,|Cert_{SP}\,|timestamp)$. *pid* is the participation identification number in this task. After a while, PV j outputs the subresult (*sre*) and sends it back: $j \rightarrow SP : output = E_{SP}(\mathrm{Sig}(SK_{PK_j^h}, rid\,|pid\,|sre)\,|Cert_j^h|\,timestamp)$.

4) REWARD MANAGEMENT

While receiving subresults of all the subtasks, the service provider aggregates them to form a final result for the requesting vehicle. After that, the service provider issues the material of the reward certificates to the participating PVs. The utilization of reward certificates simplifies reward cashing for the PVs and avoids repetitive operations about reward management for the service provider. PVs may continuously undertake computation tasks as scheduled by the service provider. They acquire several reward certificates after participating in many times of task execution in computation offloading. They can cash all the rewards simultaneously when leaving the parking lot. The cumulative rewards may compensate for their parking fee. Moreover, authentic reward certificates can be regarded as valid evidences to indicate the cooperative behavior of both the service provider and PVs in computation offloading.

We take PV j as an example: $SP \rightarrow j : material = E_{PK_j^h}(\mathrm{Sig}(SK_{SP}, rid\,|pid\,|rew))\,|Cert_{SP}\,|timestamp)$. *rew* is the rewards for the PV as scheduled. PV j decrypts the message, acquires $\mathrm{Sig}(SK_{SP}, rid\,|pid\,|rew)$ as the issued material, and particularly checks whether *rew* is right according to the presetting reward policy. After checking, the PV is required to sign the issued material to form a consentual reward certificate, and feed back it to the service provider: $j \rightarrow SP : reward\ certificate = E_{SP}(\mathrm{Sig}(SK_{PK_j^h}\,|SK_{SP}, rid\,|pid\,|rew)\,|Cert_j^h\,|timestamp)$. Ultimately, both the service provider and PV agree on the reward certificate with dual signature. The PV can cash the rewards by exhibiting the reward certificate to the service provider when necessary.

C. SECURITY ANALYSIS

In this paper, we elaborately design an interactive protocol wherein the security and privacy requirements are guaranteed for promoting PVEC. Next, we offer necessary security analysis to demonstrate that the proposed interactive protocol can achieve the following performances.

- *Message confidentiality*: Standard cryptographic primitives including asymmetric/symmetric key-based encryption and digital signatures are utilized in the protocol. Without the symmetric keys and private keys of entities, an adversary cannot open the encrypted messages. In particular, a vehicle utilizes public/private keys regarding a temporary pseudonym as session keys to communicate with a service provider. The session keys are totally generated by the certification authority, which also issues long-term master keys to the vehicle. Toward forward secrecy, the certification authority designs the session keys well to achieve that the adversary cannot compute the session keys even if the master keys are compromised. Moreover, the vehicle

changes the pseudonym after interacting with the service provider. The previous pseudonym will not be reused and be revoked after a while. Ultimately, it is very difficult for the adversary to recover the session keys and further access to messages incurred in the previous sessions. This avoids information leakage for vehicles and guarantees forward secrecy.

- *Identity authentication*: Each message is signed and attached with legal certificates. This avoids that the entities are counterfeited. The legal identities of vehicles and service providers are authenticated by verifying the signatures. We also use timestamp in all the messages to prevent replay attack caused by compromised entities.
- *Anonymity maintenance*: All the vehicles interact with service provider by using pseudonyms instead of real identities. Only the certification authority can acquire the true identities. During the task execution, participating PVs can still change their pseudonyms and use *pid* to communicate with corresponding service providers. In fact, a compromised service provider may link a limited amount of pseudonyms to a specified PV during the procedure. A restricted linkability caused by pseudonym usage is achieved for privacy preservation. Therefore, when participating in computation offloading, the identity and location privacy of individual vehicles can be protected well as a whole.
- *Reward integrity*: A participating PV cannot launch reward repudiation of reception, because the final reward certificate is also bound with its signature. This means that the PV should have approved the rewards. The participating PV cannot tamper and regenerate the reward certificate (e.g., modifying the vital parameter *rew*) due to the unforgery of the signature signed by the service provider. As for the service provider, it should pay for the participating PV according to the predesigned reward policy. Otherwise, the participating PV can disclose its cheating behavior by submitting the handshake messages to the certification authority for fair judgement.

IV. QUANTITIVE REPRESENTATION OF OPPORTUNISTIC RESOURCES

To exploit opportunistic resources well, we should ensure that PVs with dynamic mobility competent to reliable resource provision in PVEC. Hence, the availability of opportunistic resources from PVs is measured to distinguish those target PVs with great serviceability. The adequate PVs are selected for task execution in PVEC. In this section, we also offer quantitive representation about reward function, cost function of each PV and a requesting vehicle in computation offloading when they offer/occupy opportunistic resources in PVEC.

A. AVAILABILITY OF OPPORTUNISTIC RESOURCES

In PVEC, the availability of opportunistic resources from PVs is studied for reliable resource provision. In particular, the availability is exactly influenced by parking behavior. At the beginning of every resource scheduling time period, VEC servers seek target PVs for on-demand resource provision. Before the exploitation of opportunistic resources in PVEC, VEC servers evaluate the availability of idle resources owned by current PVs in resource scheduling. When a PV is predicated to stay in the specified time interval with a higher probability, the PV shows greater reliability to provide idle resources for task execution in computation offloading. This means that if tasks are allocated to the PV, the probability of re-offloading caused by the sudden departure of the PV is significantly lessened. This avoids incurring extra workloads to migrate the tasks. Hence, we use the probability that a PV will stay continuously in the entire time period, as a key metric to formulate the availability of opportunistic resources provisioned by the PV.

We consider that parking behavior of PVs can be detected, recorded and analysed by advanced technologies, e.g., big data mining [21]. The data analysis methods are supported by VEC servers. With the everlasting records and complex analysis, the probability density function of parking duration of PVs is acquired. The function is expressed by $f(t)$, where t is the parking duration and has a considerable range, $t \in [0, t^{\max}]$. Clearly, the cumulative distribution function of $f(t)$ is $F(t) = \int_0^t f(t)dt, t \leq t^{\max}$. For simplicity, we consider that there exists a service provider that schedules local VEC servers to manipulate a lot of PVs in a parking lot. At the beginning of the j-th time period, we denote a PV i in the parking lot as v_i^j. For v_i^j, the accumulative parking durations up to now is detected and denoted as at_i^j. We try to estimate the probability of staying continuously in the j_{th} time period, which can be calculated by the following conditional probability

$$p_i^j = P(t \geq at_i^j + T | t \geq at_i^j), \tag{1}$$

where T is time span of the time period. The conditional probability is to represent the probability that the PV will continue to stay parked for at least T time slots once the PV has been parked for specified time slots.

To calculate the above probability, we need to acquire the probability density function conditioned on $t > at_i^j, f(t|t > at_i^j)$. To this end, we firstly get the cumulative distribution function under the condition $t > at_i^j$,

$$F(t, t > at_i^j) = \begin{cases} F(t), & t > at_i^j \\ 0, & t \leq at_i^j \end{cases} \tag{2}$$

According to the Bayesian formula, we calculate the cumulative distribution function conditioned on $t > at_i^j, F(t|t > at_i^j)$, by

$$F(t|t > at_i^j) = \frac{F(t, t > at_i^j)}{1 - F(at_i^j)} = \frac{F(t)}{1 - F(at_i^j)}, \quad t > at_i^j. \tag{3}$$

We further obtain $f(t|t > at_i^j)$ via the derivation of $F(t|t > at_i^j)$,

$$f(t|t > at_i^j) = \frac{f(t)}{1 - F(at_i^j)}, \quad t > at_i^j. \tag{4}$$

Finally, the conditional probability p_i^j is calculated by

$$p_i^j = \int_{at_i^j+T}^{t^{\max}} \frac{f(t)}{1-F(at_i^j)} dt = \frac{1-F(at_i^j+T)}{1-F(at_i^j)}. \quad (5)$$

The value of the probability indicates whether the stability of resource provision undertaken by the PV is high enough to facilitate PVEC. Thus, we are motivated to use the value as a evaluation index when exploring the availability of opportunistic resources provisioned by the PV.

In PVEC, a service provider regards a PV as a qualified performer in computation offloading by estimating its predicted probability of staying continuously in the whole time period. With the higher probability, the PV is of better serviceability for being recruited to execute tasks. After calculating current probabilities of the existing PVs, the service provider schedules VEC servers to find the target PVs that satisfies $p_i^j \geq P_{th}$. The target PVs are employed to undertake workloads for executing tasks in computation offloading. As for P_{th}, it is a predefined threshold value for regulation in PV selection. There is a tradeoff problem when setting the value of P_{th}. Clearly, with a lower threshold value, the number of selected PVs are finally increased. But the stability of task execution becomes more difficult to be guaranteed over time.

B. REWARD AND COST IN UTILIZATION OF OPPORTUNISTIC RESOURCES

There exists a vehicle V sending a request of computation offloading to a service provider. In PVEC, the computation task can be offloaded to PVs. If the input data size uploaded by vehicle V is D_V^{in}, the total workloads of accomplishing the computation task $W = H_V D_V^{in}$, which represents the amount of executed CPU cycles. In previous work, e.g., [22] and [23], the workloads are expressed as a linear function of the input data size. H_V is an application-centric parameter related to the type of running application. The computation task with total workloads W is split as a set of subtasks and forwarded to a group of selected PVs (denoted as $\mathcal{K}$) in the resource scheduling time period. PVs choose to execute various subtasks with different workloads. In computation offloading, a part of VEC servers are also assigned to undertake workloads if the total workloads are too large. Thus, the procedure of computation offloading in PVEC is shown in Fig. 2.

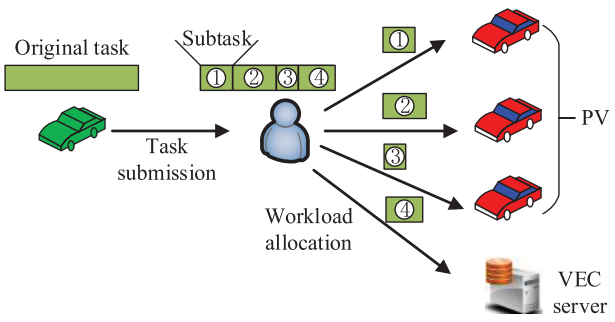


FIGURE 2. Procedure of computation offloading in PVEC.

PVs will acquire different utilities when undertaking various workloads in computation offloading. For a PV $v_k, k \in \mathcal{K}$, the service provider will ask PV v_k whether to undertake a part of the workloads. Based on individual rationality, the PV responses to the service provider according to a utility function, wherein both reward policy and workload state are considered. Let x_k^V represent a decision variable to determine the amount of undertaking workloads for serving the requesting vehicle. r_k^V and c_k are the given rewards by the requesting vehicle and consumed cost for undertaking per workload, respectively. Then the final economic benefits are equal to $(r_k^V - c_k)x_k^V$. Nevertheless, similarly to the work in [24], when the higher value of x_k^V, larger workloads lead to more negative effects for the PV. Task execution gives rise to extra workloads for the PV, and causes inconvenience to private service of the PV. The negative effects resulted by a temporary participation in computation offloading are modeled as $n_k(x_k^V)^2$. n_k is an inconvenience parameter to formulate the negative effects and related to the workload state,

$$n_k = \frac{y_k}{y_k^{\max}}. \quad (6)$$

y_k is current workloads and $y_k^{\max}$ is the maximal workloads that can be tolerated by the PV. As a result, the utility function of v_k is express as the acquired revenue minus caused negative effects:

$$U_k^V = (r_k^V - c_k)x_k^V - \omega_k n_k (x_k^V)^2, \quad (7)$$

where ω_k is a tradeoff weight parameter.

As for the requesting vehicle, its cost function is related with service fee. When a part of the workloads are allocated to the PVs, the residual part should be executed by VEC servers. The VEC servers are employed to process per workload with the required service fee f_0. Then the total payment for the VEC servers is $f_0(W - \sum_{k\in\mathcal{K}} x_k^V)$. So the overall cost of the requesting vehicle in computation offloading is expressed by

$$C_V = f_0(W - \sum_{k\in\mathcal{K}} x_k^V) + \sum_{k\in\mathcal{K}} r_k^V x_k^V. \quad (8)$$

For the requesting vehicle, it aims to optimize the reward policy for all the PVs to minimize the overall cost with satisfying several constraints. The cost minimization problem is introduced in (9) and we also offer more details about the constraints as follows. $\sum_{k\in\mathcal{K}} x_k^V \leq W$ should be satisfied to avoid excessive workloads, as shown in constraint (10). Besides, we use an individual preference parameter, ρ_V $(0 \leq \rho_V \leq 1)$, to represent the minimal ratio of the requesting workloads allocated to the PVs. Here, we consider that when the computation task is totally executed by VEC servers, the requesting vehicle pays more than that incurs when the computation task is cooperatively executed by multiple PVs. By setting the pricing policy, the usage of underutilized resources from PVs are encouraged and PVEC can be promoted. Thus, from this viewpoint, the requesting

vehicle puts forward a customized requirement about the ratio of the workloads allocated to the PVs in computation offloading. There exists a constraint: $\sum_{k\in\mathcal{K}} x_k^j \geq \rho_V W$, as shown in constraint (11). Finally, the reward policy should be positive. Based on the overall considerations, we illustrate the cost minimization problem with feasible constraints as follows.

$$\min_{r_k^V} \ C_V \tag{9}$$

$$\text{s.t.} \ \sum_{k\in\mathcal{K}} x_k^V \leq W \tag{10}$$

$$\sum_{k\in\mathcal{K}} x_k^V \geq \rho_V W \tag{11}$$

$$r_k^V \geq 0 \tag{12}$$

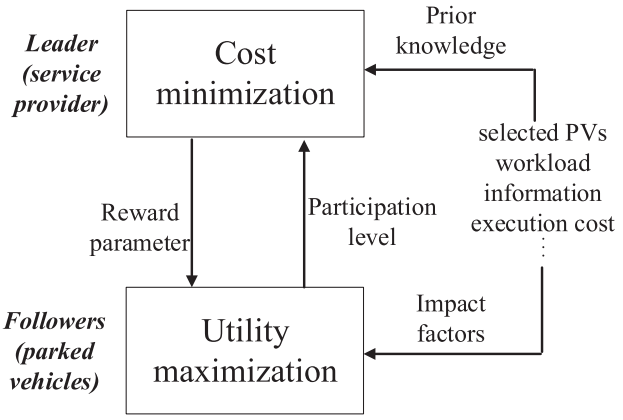


FIGURE 3. Stackelberge game model between a service provider and PVs.

V. OPTIMAL RESOURCE SCHEDULING

In this section, we pay attention to a critical problem about how a service provider schedules opportunistic resources to serve a requesting vehicle for maximizing user satisfaction in computation offloading.

A. STACKELBERG GAME MODEL

To improve user satisfaction, the service provider takes the overall cost of the requesting vehicle into consideration. Then the service provider represents the requesting vehicle to negotiate with PVs when employing them for task execution. To recruit PVs, the interaction between the service provider and PVs is modeled as a typical leader-follower game. More specifically, for encouraging the PVs to undertake workloads, the service provider acts as the broker of the requesting vehicle to reward the PVs with incentives indicated by r_k^V. Based on given rewards, the participating PVs make optimal decisions about the amount of undertaking workloads x_k^V for maximizing the utilities. After that, their responses are collected to the service provider and it optimizes the reward policy for the PVs to minimize the overall service cost. This means that in the incentive mechanism, the service provider is naturally fit for acting a leader to determine the final reward policy while the PVs become followers responding to the service provider with respect to given rewards. According to the descriptions in [25], a convenient analytical model to study the above scenario is provided by Stackelberg game.

After determining players and their utility functions, the Stackelberg game between the service provider and PVs is defined by

$$\Gamma = \{(V \cup \{v_k\}_{k\in\mathcal{K}}), (\{x_k^V\}_{k\in\mathcal{K}}, \{p_k^V\}_{k\in\mathcal{K}}), (C_V, \{U_k^V\}_{k\in\mathcal{K}})\}.$$

The strategic form consists of three aspects: a player set, strategy space and corresponding utility functions. In Fig. 3, a Stackelberg game model is proposed to formulate the above incentive mechanism for optimizing resource scheduling in PVEC. It is noted that the service provider is a leader while all the selected PVs are followers. The service provider finally determines the optimal reward parameter p_k^{V*} based on prior knowledge about the impacts of the decision on behavior of the PVs. In particular, we consider that the service provider can realize the amount of workloads required by the requesting vehicle, and acquire workload state and task execution information from the PVs. Due to the attractive incentives, the PVs would like to upload the information in a secure way. As for each PV, the best participation level x_k^{V*} is put forward for utility maximization with the condition of p_k^{V*}. Next, we utilize the theoretic approach of Stackelberg game to analyse their best responses accordingly.

According to game theory, an optimal solution for such a Stackelberg game is the Stackelberg equilibrium at which a leader obtains the optimal utility given followers' best responses. For the Stackelberg game Γ, we define the Stackelberg equilibrium as follows:

Definition 1: (r_k^{V*}, x_k^{V*}) can be achieved as the proposed Stackelberge equilibrium if and only if it satisfies the following set of inequalities:

$$\forall r_k^V, C_V(r_k^{V*}, x_k^{V*}) \leq C_V(r_k^V, x_k^{V*})$$
$$\forall x_k^V, U_k^V(r_k^{V*}, x_k^{V*}) \geq U_k^V(r_k^{V*}, x_k^V).$$

The objective of the proposed Stackelberg game Γ is to find the unique Stackelberg equilibrium where both the requesting vehicle and PVs have no motivations to change their decisions. At this equilibrium, both the leader and any follower cannot benefit, in terms of overall cost and individual utility, respectively, by unilaterally changing the strategies. Hence, when all the players are at the Stackelberg equilibrium, the service provider cannot help the requesting vehicle reduce the overall cost by decreasing the reward parameter from the Stackelberg equilibrium value r_k^{V*}. Similarly, no PV is able to improve the utility by changing different workloads in addition to the Stackelberg equilibrium value x_k^{V*} for undertaking.

B. STACKELBERG EQUILIBRIUM ANALYSIS

Firstly, we analyse the best response of a follower. U_k^V can be converted into an optimal utility function in terms of x_k^V. We take the second derivatives of U_k^V with respect to x_k^V, and easily find that $\partial^2 U_k^V / \partial x_k^{V2} < 0$. The utility function U_k^{V*} is concave and this indicates that the maximal value of the

function exists. By using $\partial U_k^V / \partial x_k^V = 0$, we obtain

$$x_k^{V*} = \frac{r_k^V - c_k}{2\omega_k n_k}. \tag{13}$$

x_k^{V*} is called the best response of PV v_k in determining the participation level for serving requesting vehicle V, which maximizes the utility on the condition of given rewards r_k^V. Clearly, the best response is positive only when $r_k^V \geq c_k$ and the PV is recruited for task execution.

By substituting x_k^{V*} into the cost function C_V, the service provider reformulates the optimization problem as follows

$$\begin{aligned} &\min_{r_k^V} f_0 W + \sum_{k\in\mathcal{K}} \frac{(r_k^V)^2 - (f_0 + c_k) r_k^V + f_0 c_k}{2\omega_k n_k} \\ &\text{s.t. } \rho_V W \leq \sum_{k\in\mathcal{K}} \frac{r_k^V + c_k}{2\omega_k n_k} \leq W \\ &\quad r_k^V \geq c_k \end{aligned} \tag{14}$$

For the above problem, we simplify it as follows:

$$\begin{aligned} &\min_{r_k^V} \sum_{k\in\mathcal{K}} [a_k (r_k^V)^2 - b_k r_k^V] \\ &\text{s.t. } d \leq \sum_{k\in\mathcal{K}} a_k r_k^V \leq e \\ &\quad r_k^V \geq c_k \end{aligned} \tag{15}$$

where $a_k = \frac{1}{2\omega_k n_k}$, $b_k = a_k(f_0 + c_k)$, $d = \rho_V W + \sum_{k\in\mathcal{K}} \frac{c_k}{2\omega_k n_k}$ and $e = d + (1 - \rho_V)W$.

By proposing the Lagrange multipliers α, β and γ_k for the constraints accordingly, the Lagrange function with respect to the problem in (14), L, is expressed by

$$\begin{aligned} L = \sum_{k\in\mathcal{K}} [a_k (r_k^V)^2 - b_k r_k^V] - \alpha(\sum_{k\in\mathcal{K}} a_k r_k^V - d) \\ + \beta(\sum_{k\in\mathcal{K}} a_k r_k^V - e) - \gamma_k (r_k^V - c_k) \end{aligned} \tag{16}$$

We also find that $\partial^2 L / \partial r_k^{V2} = 2\, a_k \geq 0$. So the optimization problem in (14) is convex with linear constraints. There exists a unique r_k^{V*} to be solved as the best response of the service provider under the given condition of x_k^{V*}.

C. SUBGRADIENT-BASED ITERATIVE ALGORITHM

In this paper, we use subgradient method to solve the typical convex optimization problem and acquire x_k^{V*}. According to the method, the optimization problem in (14) is transformed as

$$\begin{aligned} &\min_{r_k^V} L(r_k^V, \alpha, \beta, \gamma_k) \\ &\text{s.t. } \alpha, \beta, \gamma_k \geq 0 \end{aligned} \tag{17}$$

We update the above multipliers round by round as follows:

$$\begin{cases} \alpha^{\ell+1} = \left| \alpha^{\ell} - \kappa(\sum_{k\in\mathcal{K}} a_k r_k^V - d) \right|^+ \\ \beta^{\ell+1} = \left| \beta^{\ell} + \eta((\sum_{k\in\mathcal{K}} a_k r_k^V - e) \right|^+ \\ \gamma_k^{\ell+1} = \left| \gamma_k^{\ell} - \lambda_k (r_k^V - c_k) \right|^+ \end{cases} \tag{18}$$

Here, ℓ is the index of the round while κ, η and λ_k are three presetting step sizes. And $|x|^+ = \max(x, 0)$. As for updating the solution of $r_k^{V,\ell+1}$, we calculate it based on the KKT condition, $\partial L / \partial r_k^V = 0$. Thus, we have

$$r_k^{V,\ell+1} = 0.5 * (f_0 + c_k - \alpha^{\ell} + \beta^{\ell}) + \omega_k n_k \gamma_k^{\ell}. \tag{19}$$

According to the parameter update law, an iterative algorithm based on subgradient method is presented to reach the Stackelberg equilibrium. In practice, the service provider realizes the request and acts as the broker of the requesting vehicle to negotiate with the PVs in a one-round Stackelberg game. In particular, the communications among them are significantly supported by the interactive protocol proposed in Section III-B.

The algorithm is executed by VEC servers belonging to the service provider. The service provider is convenient to hold prior knowledge of the computation task and related parameters of the PVs via the incentive mechanism. At First, by knowing impacts of the decision on the behavior of the PVs, the service provider formulates the optimization problem in (14). Then the subgradient method is utilized for solving the optimal solution of r_k^V round by round. The process about parameter update is repeated until the iteratively updating $r_k^{V,\ell}$ is changed within a definitively smaller range. ε is a small positive constant to make the algorithm converge with specified accuracy. The value of ε can be adjusted to influence the number of executed rounds when necessary. Finally, within the limited rounds, r_k^{V*} is acquired and submitted to the PVs. Each PV replies with the best response x_k^{V*} for maximizing the utility. Requesting workloads are allocated by the service provider to all the PVs and VEC severs, respectively. More details about the algorithm can be found in **Algorithm 1**.

Theorem 1: A unique Stackelberg equilibrium exists between the service provider and all the PVs in our proposed Stackelberg game.

Proof: With the given reward policy, each PV acts as a follower and always has its own best response x_k^{V*} due to the concave character of the utility function. Based on the precognition of x_k^{V*}, the optimization problem in (14) is formulated. The problem is a convex optimization problem and we can obtain the optimal solution, r_k^{V*}, by using **Algorithm 1**. At the same time, for the leader, the service provider has a unique optimal strategy r_k^{V*} under given the best strategies of all the PVs. Ultimately, both the leader and followers are fully satisfied. Their decisions (r_k^{V*}, x_k^{V*}) make that their utilities have been maximized simultaneously. Moreover, when all the

Algorithm 1 Subgradient-Based Iterative Algorithm for Reaching the Stackelberg Equilibrium

Input: The knowledge of the computation task and PVs: $f_0, \{h_V, D_V^{in}, \rho_V\}$ and $\{c_k, \omega_k, y_k, y_k^{max}\}, k \in \mathcal{K}$.
Output: The final workload allocation results and reward policy $\{x_k^{V*}, r_k^{V*}\}, k \in \mathcal{K}$.

Initialization: $\ell = 0$, $r_k^{V,0}$, α^0, β^0, γ^0 and fixed step sizes: κ, η and λ.
Calculate the workloads and inconvenience parameter: $W = h_V D_V^{in}$ and $n_k = y_k / y_k^{max}$
repeat
  Based on Eqn. (18), update the multipliers: $\alpha^{\ell+1}$, $\beta^{\ell+1}$ and $\gamma_k^{\ell+1}$.
  Based on the KKT condition, update $r_k^{V,\ell+1}$ by using Eqn. (19).
  The next round is continued, $\ell = \ell + 1$.
until $(\left| r_k^{V,\ell} - r_k^{V,\ell-1} \right| \leq \varepsilon)$;
The final reward policy is determined, $r_k^{V*} = r_k^{V,\ell}$.
for *PVs in the set* $\mathcal{K}$ **do**
  Based on Eqn. (13), the best response x_k^{V*} is replied.
end
The residual amount of workloads indicated by $W - \sum_{k \in \mathcal{K}} x_k^{V*}$ are allocated to VEC servers.
final;
return $\{x_k^{V*}, r_k^{V*}\}$

players, including each PV and the requesting vehicle, have their optimized payoff and cost, respectively, considering the strategies chosen by other players in the game. They have no incentives to change the decisions and take other actions. Thus, the unique Stackelberg equilibrium indicated by (r_k^{V*}, x_k^{V*}) is reached in the game. ■

VI. NUMERICAL RESULTS

We evaluate the performance of the proposed PVEC by extensive simulations. For simplicity, we consider that there exists a requesting vehicle sending a computation task to a service provider. The service provider rents multiple VEC servers and manipulates a lot of PVs in a parking lot for task execution in computation offloading.

A. PV SELECTION IN PVEC

In the simulations, PV behavior is formulated based on a real dataset from ACT Government Open Data Portal dataACT. The real dataset uses the SmartParking app to collect 180295 parking records in the Manuka shopping precinct within a month [26]. We select about 60,000 PVs for an observation, and make data statistic and analysis on their parking behavior. To summarize, parking duration of a PV ranges from 8 to 240 minutes. Based on the statistic data, we draw the frequency distribution histogram about parking duration of all the PVs in the dataset, as shown in Fig. 4. Here, the class

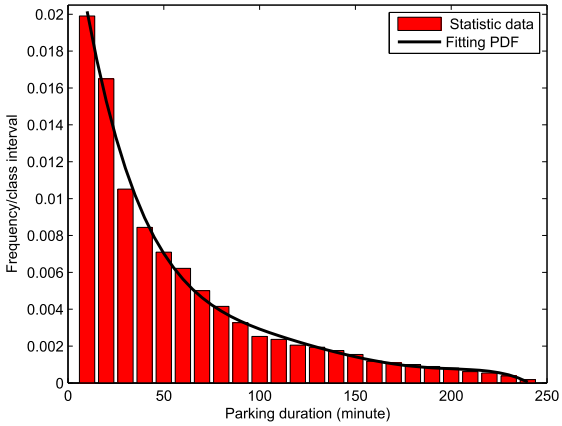


FIGURE 4. Frequency distribution histogram about parking duration of one PV in the dataset.

interval for grouping statistics is 10 minutes. Furthermore, we use the professional curve fitting tool provided by Matlab software, cftool, to fit and get the probability density function (PDF) of parking duration of these PVs. The fitting PDF is also illustrated by the black curve in the figure.

According to the fitting PDF, we can measure the availability of opportunistic resource offered by PVs in PVEC. At the same time, we calculate the serviceability of a PV for task execution in a resource scheduling time period. T is to indicate the time span of the time period. Based on Eqn. (5), the serviceability of a PV is equal to the predicted probability of staying continuously within T minutes. As shown in Fig.5, we find that the serviceability is decreased with the accumulative parking durations. As time goes by, those PVs that have been parked with longer time durations are exactly inadequate for being selected for task execution. This is because that for them, the probability of a sudden departure is increased and thus, this leads to lower serviceability in PVEC. Besides, the serviceability of a PV is decreased with the expanding time window, T. The increasing value of T is

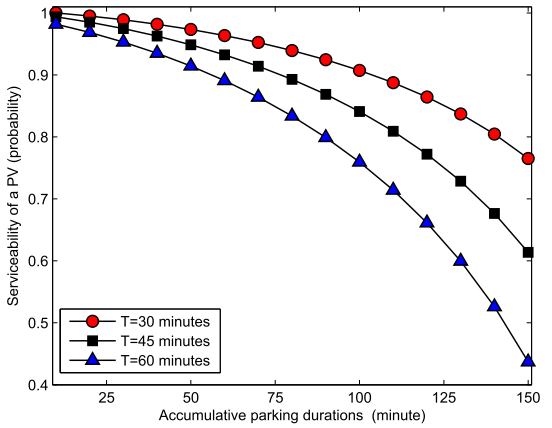


FIGURE 5. Comparison of serviceability of a PV with respect to different accumulative parking durations and T.

to add a strict restriction for estimating PV behavior when selecting adequate PVs. As a consequence, a PV is estimated with a lower probability of staying continuously within the entire time period. So the PV is difficult to be evaluated with the valid serviceability in an extended resource scheduling time period. For example, the evaluated serviceability of the PV that has been parked with 100 minutes is reduced more than 16% when the time period is extended from $T = 30$ to $T = 60$ minutes.

In practical scenarios, the value of T needs to be consider well. When the time period is set with shorter span, more PVs can be selected but frequent update of routing table is caused to the service provider in the communications with the PVs. In turn, less PVs are selected for task execution and the corresponding reliability of resource provision cannot be also guaranteed at the end of the time period when the time period is set with longer span. Based on the overall considerations, we set the time period as 30 minutes in the following simulations.

B. PERFORMANCE EVALUATION OF RESOURCE SCHEDULING

In a 30-minute resource scheduling time period, the arrival of PVs follows Poisson distribution and their parking durations are distributed to follow the fitting probability density function. In PV selection, the threshold value P_{th} is set as 0.9. For the PVs, the consumed cost for executing per giga CPU cycles c ranges randomly from 0.1 to 2. The maximal workload is 1000 giga CPU cycles and the current workload is distributed uniformly over [100, 500] giga CPU cycles. The value of the tradeoff weight ω changes from 0.5 to 1.5 for every PV. As for the parameters in the subgradient-based iterative algorithm, crucial step sizes κ, η and λ_k are valued by 0.03, 0.02 and 0.05. The small positive constant ε is set by 0.01 to make the algorithm converge.

To evaluate our scheme, we compare the performances between our proposed PVEC and existing work that only introduce the proposal of VEC, e.g., [10]. Firstly, we demonstrate that PVEC combines idle resources from PVs with current resources in VEC to collectively extend resource capacity of vehicular networks. Workloads required by mobile applications are allocated and executed in a optimized way. Finally, sufficient resources are scheduled on demand to support a variety of services. Network capacity can be improved and more vehicles are served at the network edge.

For example, we observe a scenario wherein a service provider rents 2 VEC servers whose computing capability is 8 GHz. In PVEC, the service provider also manipulates multiple PVs in the parking lot for task execution. In general, computing capability of a PV ranges from 0.5 to 1.5 GHz. Besides, the arrival of uploading tasks follows Poisson distribution. For simplicity, the arrival rate is 1 per minute. For the requesting tasks, the input data size for computation offloading is distributed over [300, 1000] KB and application-centric parameter H range from 2 to 5 mega CPU cycles per byte. As shown in Fig. 6, total amount of served vehicles belonging

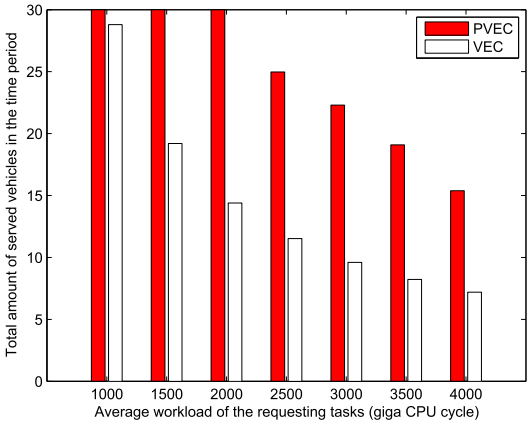


FIGURE 6. Comparison of total amount of served vehicles with respect to different schemes.

to the service provider in the time period is clearly decreased with the increasing average workload of the requesting tasks. But compared with the existing scheme, our scheme (i.e., PVEC) achieves more served vehicles, owing to the extended resource capacity. When the average workload of all the requesting workloads is 2500 giga CPU cycles, the total amount of served vehicles in PVEC is about twice that in the existing scheme. This means that PVEC outperforms the general VEC and has a great advantage in offering sufficient network resources to serve more vehicles.

Next, we pay attention to resource scheduling optimization in PVEC and consider how to minimize the overall cost for a user. We take a requesting vehicle V running an application studied in [27] as an example. The input data size for computation offloading is 400 KB while the application-centric parameter $H_V = 2.5$ mega CPU cycles per byte. The individual preference ρ_V is 0.2. For the requesting vehicle, its utility is directly related with overall cost for offloading the computation task, C_V. In the previous work, the computation task is entirely executed by VEC servers, whose service fee is f_0 for undertaking per giga-cycle workloads. Thus, C_V is linearly increased with the increasing value of service fee f_0, as shown by the dash line with diamond markers in Fig. 7. But in PVEC, C_V is jointly optimized by considering f_0 and execution cost of the PVs indicated by c.

Fig. 7 demonstrates that the overall cost of the requesting vehicle in PVEC is greatly lower than that in VEC. PVEC offers additional ways for task execution in computation offloading. The computation task can be offloaded to both idle PVs and VEC servers as scheduled. In our scheme, the service provider seeks an optimal strategy about workload allocation for minimizing the overall cost of the requesting vehicle. When f_0 is increased, the service provider puts more workloads to the PVs with lower execution cost, as shown in Fig. 8. When necessary (i.e., f_0 is too large), the service provider puts all the workloads to the PVs. The workloads allocation between VEC servers and the PVs are dynamically determined under different conditions of f_0 and execution

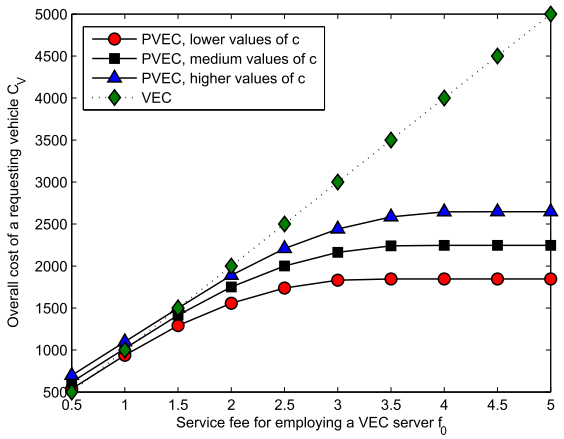


FIGURE 7. Comparison of overall cost of a requesting vehicle with respect to different schemes.

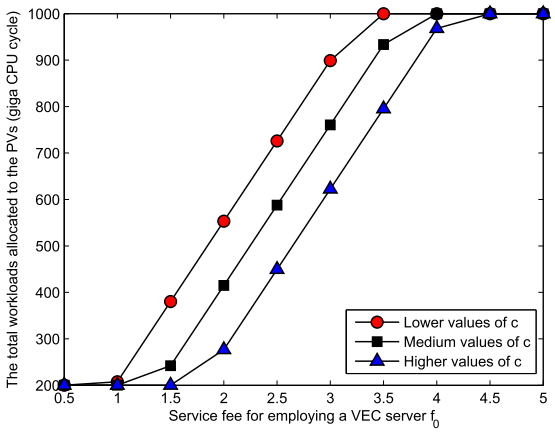
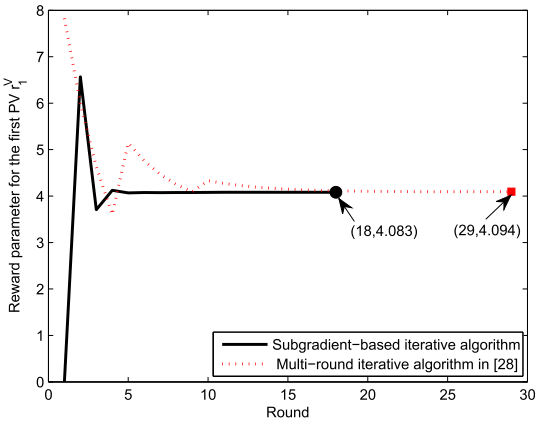
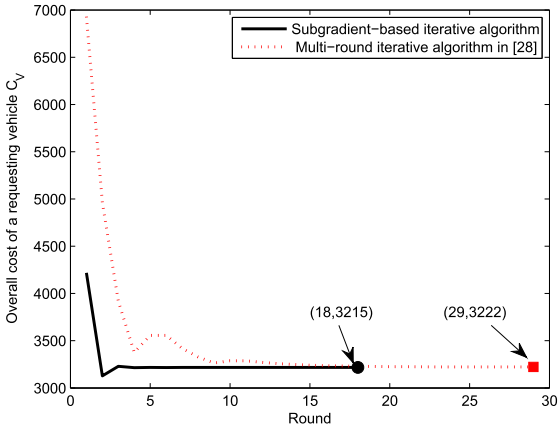


FIGURE 8. Comparison of total workloads allocated to the PVs with respect to different values of *c*.



(a)



(b)

FIGURE 9. Performance comparison of two algorithms for reaching the Stackelberg equilibrium. (a) Comparison of two algorithms regarding convergence rates. (b) Comparison of two algorithms regarding approximation effect.

cost of the PVs. Hence, the proposed PVEC is superior to the existing work in reducing service cost for a vehicle and improving user satisfaction finally.

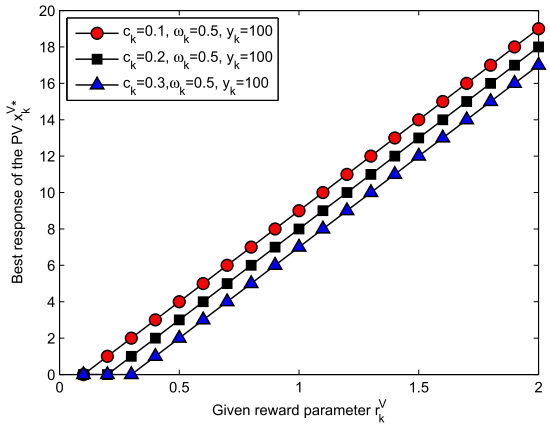
Fig. 7 also shows that the overall cost of the requesting vehicle is increased with the increasing execution cost of all the PVs. The values of c regarding all the PVs are divided into three kinds: lower, medium and higher. Accordingly, the average value of c is set as $[0.4, 0.8, 1.2]$. Higher values of c means that the PVs should consume more expenditure in executing the computation task. For ensuring positive incentive for the PVs, this also results in higher incentives from the service provider. For example, when $f_0 = 3$, the overall cost of the requesting vehicle is increased about 33% when the values of c are changed from lower level to higher level. But at this time (i.e., higher values of c), the overall cost in PVEC is still only 81% of that in the existing work. In Fig. 8, we can also find that the total workloads allocated to the PVs are clearly decreased with the increasing values of c under the same condition of f_0. Based on the rationality requirement, when the execution cost of all the PVs increasing, the service provider would like to reduce the workloads assigned to them.

C. STACKELBERG GAME APPROACH

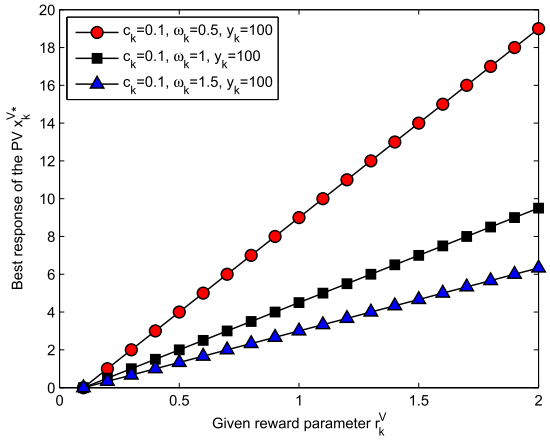
1) ALGORITHM VALIDATION FOR THE STACKELBERG GAME

We validate **Algorithm 1** to verify the effectiveness of the algorithm. In this paper, we present a subgradient-based iterative algorithm for reaching the Stackelberg equilibrium. In previous work, there existed a multi-round iterative algorithm wherein both the leader and followers go through multiple rounds to reach the Stackelberg equilibrium. The algorithm is particularly studied for a Stackelberg game based incentive mechanism in [28]. In each round, all the followers need to respond with the strategies two times, based on the different strategies of the leader. Compared with the algorithm, our algorithm only updates fixed 4 parameters in each round.

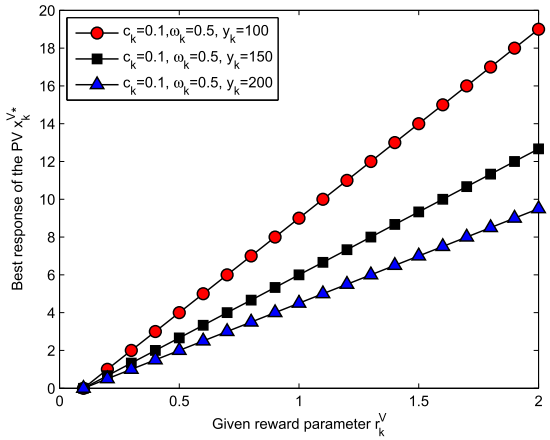
For algorithm validation, we compare the performance among the two algorithms in terms of convergence and accuracy. We set the same termination condition for the multi-round iterative algorithm. Ultimately, both of the two algorithms are able to converge within limited rounds.



(a)



(b)



(c)

FIGURE 10. Performance comparison of the best response x_k^{V*} with respect to different p_k^V, c_k, θ_k and y_k. (a) Different values of execution cost c_k. (b) Different values of tradeoff weight ω_k. (c) Different values of current workloads y_k.

In Fig. 9(a), dynamically changing processes of one reward parameter, e.g., r_1^V, in the two algorithms are shown with details. In our proposed algorithm, r_1^V stops changing and acquires a final value after 18 rounds while 29 rounds are executed to obtain the changeless value of r_1^V in the multi-round iterative algorithm. This means that the proposed algorithm converges faster than the existing algorithm. Moreover, the optimization objective C_V can also be achieved with a smaller value in the subgradient-based iterative algorithm, as shown in Fig. 9(b). To summarize, the proposed algorithm is effective and efficient to reach the Stackelberg equilibrium. The proposed algorithm exhibits great advantages in the faster convergence rate and better approximation effect.

2) IMPACTS TO THE STACKELBERG GAME

Next, the impacts of different system parameters for the best response of a PV in the Stackelberg game are studied and illustrated by Fig. 10. We randomly choose a PV k for observations. Based on Eqn. (13), the best response of PV k for executing the task from the requesting vehicle, x_k^{V*}, is influenced the following parameters: given reward parameter r_k^V, its execution cost for undertaking per workload c_k, the individual tradeoff weight ω_k and current workloads y_k. Clearly, with the higher value of the reward parameter, the PV would like to undertake more workloads for earing more benefits. So x_k^{V*} is increased with the higher value of r_k^V.

In turn, both c_k and y_k play a negative effect on x_k^{V*} due to the increasing execution cost and incurred inconvenience for the PV. Based on the rationality, the PV chooses to reduce its undertaking workloads. Similar negative effect can be resulted by increasing value of the tradeoff weight ω_k. We take that ω_k is changed from 0.5 to 1.5 when $c_k = 0.1$, $y_k = 100$ and $r_k^V = 1$ as an example. Under the circumstance, the increment of ω_k directly gives rise to that x_k^{V*} suffers from 66.7% percentage declines. Higher value of ω_k means that the PV pays less attention to earn benefits in task execution. So less workloads are the better choice to undertake and x_k^{V*} is decreased finally. To summarize, the above numerical results demonstrate that the Stackelberg game approach is effective and efficient for both the service provider and PVs.

VII. CONCLUSIONS

In this paper, we propose a new computing paradigm named by PVEC, where common PVs with available computing capability and storage space are utilized as edge computing nodes at the vehicular network edge. In PVEC, distributed mobile applications are achieved via the coordination between PVs and VEC servers, and supported with reliable resource guarantee. To facilitate distributed mobile applications, we focus on the high-efficiency scheduling of opportunistic resources in PVEC. More specifically, we present a system architecture to introduce primary network entities in PVEC, and an interactive protocol to support communications among them. Besides, we identify appropriate PVs for reliable task execution by measuring the availability of opportunistic resources in PVEC. The PVs are selected to undertake workloads for serving a requesting vehicle in computation offloading. Stackelberg game approach is leveraged

to formulate and solve the the resource scheduling optimization problem. The goal of the game is to minimize the overall cost of users. After that, to reach the Stackelberg equilibrium, a subgradient-based iterative algorithm is proposed to determine workload allocation among the PVs and ensures the economy optimality of service provision. Through extensive simulations based on a real dataset, we demonstrate that compared with existing work, PVEC has superior performances in improving network capacity and reducing service fee in computation offloading. Numerical results also demonstrate that the Stackelberg game approach is effective and efficient in PVEC.

REFERENCES

[1] J. Moorhead and T. Nixon, *Carbon Pricing on the Horizon*. New York, NY, USA: Thomson Reuters, 2015.
[2] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—A key technology towards 5G," ETSI, Sophia Antipolis, France, White Paper 11, 2015, pp. 1–16.
[3] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018.
[4] K. Hong, D. Lillethun, U. Ramachandran, B. Ottenwälder, and B. Koldehofe, "Mobile fog: A programming model for large-scale applications on the Internet of Things," in *Proc. 2nd ACM SIGCOMM Workshop Mobile cloud Comput.*, 2013, pp. 15–20.
[5] C. Morency and M. Trépanier, "Characterizing parking spaces using travel survey data," CIRRELT, Montreal, QC, Canada, Tech. Rep. TR 2008-15, May 2008.
[6] T. Litman, *Parking Management: Strategies, Evaluation and Planning*. Victoria, BC, Canada: Victoria Transport Policy Institute, 2006.
[7] A. Aliyu *et al.*, "Cloud computing in VANETs: Architecture, taxonomy, and challenges," *IETE Tech. Rev.*, vol. 35, no. 5, pp. 523–547, 2017.
[8] S. Abdelhamid, H. Hassanein, and G. Takahara, "Vehicle as a resource (VaaR)," *IEEE Netw.*, vol. 29, no. 1, pp. 12–17, Jan. 2015.
[9] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans. Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.
[10] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proc. IEEE 17th Int. Conf. Communs. (ICC)*, Paris, France, May 2017, pp. 1–6.
[11] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive offloading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Jun. 2017.
[12] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 94–100, Jul. 2017.
[13] X. Huang, R. Yu, J. Kang, and Y. Zhang, "Distributed reputation management for secure and efficient vehicular edge computing and networks," *IEEE Access*, vol. 5, pp. 25408–25420, 2017.
[14] Y. Cao *et al.*, "Mobile edge computing for big-data-enabled electric vehicle charging," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 150–156, Mar. 2018.
[15] N. Kumar, S. Zeadally, and J. J. Rodrigues, "Vehicular delay-tolerant networks for smart grid data management using mobile edge computing," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 60–66, Oct. 2016.
[16] S. Arif, S. Olariu, J. Wang, G. Yan, W. Yang, and I. Khalil, "Datacenter at the airport: Reasoning about time-dependent parking lot occupancy," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 11, pp. 2067–2080, Nov. 2012.
[17] N. Liu, M. Liu, W. Lou, G. Chen, and J. Cao, "PVA in VANETs: Stopped cars are not silent," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 431–435.
[18] F. Malandrino, C. Casetti, C.-F. Chiasserini, C. Sommer, and F. Dressler, "The role of parked cars in content downloading for vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 9, pp. 4606–4617, Nov. 2014.
[19] Z. Su, Q. Xu, Y. Hui, M. Wen, and S. Guo, "A game theoretic approach to parked vehicle assisted content delivery in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6461–6474, Jul. 2017.
[20] L. Le, A. Festag, R. Baldessari, and W. Zhang, "Vehicular wireless short-range communication for improving intersection safety," *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 104–110, Nov. 2009.
[21] A. Nara, M.-H. Tsou, J.-A. Yang, and C.-C. Huang, *The Opportunities and Challenges with Social Media and Big Data for Research in Human Dynamics*. Cham, Switzerland: Springer, 2018, pp. 223–234.
[22] J. Song, Y. Cui, M. Li, J. Qiu, and R. Buyya, "Energy-traffic tradeoff cooperative offloading for mobile cloud computing," in *Proc. IEEE 22nd Int. Symp. Qual. Service (IWQoS)*, May 2014, pp. 284–289.
[23] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," *HotCloud*, vol. 10, pp. 4–10, Jun. 2010.
[24] X. Wang, X. Chen, W. Wu, N. An, and L. Wang, "Cooperative application execution in mobile cloud computing: A Stackelberg game approach," *IEEE Commun. Lett.*, vol. 20, pp. 946–949, May 2016.
[25] J. Cruz, Jr., "Survey of nash and stackelberg equilibrim strategies in dynamic games," *Ann. Econ. Social Meas.*, vol. 4, no. 4, pp. 339–344, 1975.
[26] *ACT Government Open Data Portal dataACT*. Accessed: 2017. [Online]. Available: https://www.data.act.gov.au/Transport/SmartParking-History/grth-myzs
[27] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: Real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proc. 17th IEEE Symp. Comput. Commun. (ISCC)*, Cappadocia, Turkey, Jul. 2012, pp. 59–66.
[28] Y. Liu, C. Xu, Y. Zhan, Z. Liu, J. Guan, and H. Zhang, "Incentive mechanism for computation offloading using edge computing: A Stackelberg game approach," *Comput. Netw.*, vol. 129, pp. 399–409, Dec. 2017.

XUMIN HUANG is currently pursuing the Ph.D. degree in networked control systems with the Guangdong University of Technology, China. His research interests mainly focus on network performance analysis, simulation, and enhancement in wireless communications and networking.

RONG YU (M'08) received the Ph.D. degree from Tsinghua University, China, in 2007. He worked in the School of Electronic and Information Engineering, South China University of Technology. In 2010, he joined the Institute of Intelligent Information Processing, Guangdong University of Technology, where he is currently a Full Professor. His research interests include wireless networking and mobile computing in featured environments such as edge cloud, connected vehicles, smart grid, and Internet of Things. He is the co-inventor of over 30 patents and author or co-author of over 100 international journal and conference papers. He was a member of the Home Networking Standard Committee in China, where he led the standardization work of three standards.

JIANQI LIU (M'14) received the Ph.D. degree (majoring in control science and engineering) and the M.S. degree (majoring in computer software and theory) from the Guangdong University of Technology, GuangZhou, China, the B.S degree in computer science and technology from Nanchang University, Nanchang, China. He is currently an Associate Professor with the School of Automation, Guandong University of Technology, China. His current research interests are Internet of Vehicle, Internet of Things, and Cyber-Physical Systems.

LEI SHU (M'07–SM'15) received the B.Sc. degree in computer science from South Central University for Nationalities, China, in 2002, the M.Sc. degree in computer engineering from Kyung Hee University, South Korea, in 2005, and the Ph.D. degree from the Digital Enterprise Research Institute, National University of Ireland, Galway, Ireland, in 2010. Until 2012, he was a Specially Assigned Researcher with the Department of Multimedia Engineering, Graduate School of Information Science and Technology, Osaka University, Japan. He is currently a Distinguished Professor with Nanjing Agricultural University, China, and a Lincoln Professor with the University of Lincoln, U.K. He is also the Director of the NAU-Lincoln Joint Research Center of Intelligent Engineering. His main research fields are wireless sensor networks and Internet of Things. He has published over 380 papers in related conferences, journals, and books in the areas of sensor networks. His current H-index is 42 and i10-index is 153 in Google Scholar Citation. He was a recipient of GLOBECOM 2010, ICC 2013, ComManTel 2014, the 2014 Top Level Talents in Sailing Plan of Guangdong Province, China, the 2015 Outstanding Young Professor of Guangdong Province, WICON 2016, and the SigTelCom 2017 Best Paper Awards, the 2017 and 2018 IEEE Systems Journal Best Paper Awards, and the Outstanding Associate Editor Award of 2017 IEEE ACCESS. He has served over 50 various Co-Chair for international conferences/workshops, such as IWCMC, ICC, ISCC, ICNC, and Chinacom, especially the Symposium Co-Chair for the IWCMC 2012 and ICC 2012, the General Co-Chair for Chinacom 2014, Qshine 2015, Collaboratecom 2017, DependSys 2018, and SCI 2019, the TPC Chair for InisCom 2015, NCCA 2015, WICON 2016, NCCA 2016, Chinacom 2017, InisCom 2017, WMNC 2017, and NCCA 2018, a TPC member of over 150 conferences, such as ICDCS, DCOSS, MASS, ICC, GLOBECOM, ICCCN, WCNC, and ISCC. He has been serving as an Associate Editor for the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the *IEEE Communications Magazine*, the *IEEE Network Magazine*, the IEEE SYSTEMS JOURNAL, the IEEE ACCESS, the IEEE/CAA JOURNAL OF AUTOMATIC SINICA, and *Sensors*.