Joint Resource Allocation and Computation Offloading With Time-Varying Fading Channel in Vehicular Edge Computing

Shichao Li, Siyu Lin, *Member, IEEE*, Lin Cai, *Fellow, IEEE*, Wenjie Li, and Gang Zhu

***Abstract*—Vehicular edge computing (VEC) is considered as a novel paradigm to enhance the safety of automated vehicles and intelligent transportation systems (ITS). The computation offloading strategies are the key point of VEC, and the effect of time-varying channels cannot be ignored during the task transmission period. This paper investigates the utility maximization problem with task delay requirement constraints, in which the influence of time-varying channel on the task offloading strategies during the task offloading period is considered. The time-varying fading channel leads to the time-varying spectrum efficiency (SE), so the previous offloading strategies are questionable when the additional uncertain allocated bandwidth is taken into account. To deal with it, we first propose a linearization based Branch and Bound (LBB) algorithm to solve the fixed SE problem without considering the time-varying channel characteristics. Considering the complexity of the LBB algorithm, a closest rounding integer (CRI) algorithm is proposed to solve the fixed SE problem. Then, based on the resource allocation strategies of the fixed SE problem, we propose the LBB based computation offloading (LBBCO) algorithm and the CRI based computation offloading (CRICO) algorithm to solve the original problem for both the static tasks and dynamic tasks. The proposed LBBCO/CRICO algorithms are also applicable to multi-vehicle and multi-task scenarios. Furthermore, we analyze the effect of small-scale fading on the proposed offloading strategies. The simulation results show that the average utilities of LBBCO and CRICO algorithms have a small gap by 3.93% and 6.13% only to the upper bound, respectively. Meanwhile, the proposed LBBCO and CRICO algorithms can outperform the previous state-of-the-art solution by 4.52% and 2.38%, respectively.**

***Index Terms*—Edge computing, resource management, time-varying channels, wireless communication.**

I. Introduction

IN RECENT years, the Internet of Things (IoT) has attracted considerable attention [1]–[3]. As an important application of IoT, Internet of Vehicles (IoV) not only provides an environment that supports communication among vehicles, as well as between vehicles and roadside units (RSUs), but also realizes intelligent traffic management and intelligent vehicle control network integration [4]–[6].

IoV can provide many new services for people, such as autonomous driving, speech recognition, and online video [7], [8]. These applications and services require significant computation resources and strict delay constraints. However, the computation resources of the vehicles are limited, insufficient to support many applications [9], [10]. Vehicular edge computing (VEC), which can provide elastic computation resources on demand, is a promising solution. Compared with cloud computing, the VEC framework pushes the cloud service to the edge of the radio network. By utilizing the radio access network (RAN) in the VEC paradigm, some of the control and data plane functions are deployed to the VEC servers. Therefore, VEC can alleviate the computing and routing burdens significantly, and improve resource utilization. Moreover, since the VEC server is closer to the vehicles, the end-to-end delay can be reduced.

By using the VEC paradigm, resource utilization can be improved, and user experience of the computation resource-hungry applications can be enriched. However, compared with the traditional cloud servers, the VEC servers have limited resources [11], [12]. To improve the computation ability of VEC and utilize resources efficiently, the dynamic radio and computation resource allocation schemes should be carefully developed.

The joint radio and computation resource allocation play an important role in realizing high utility and low latency of the VEC system. In the current study, all the works assume that the channel fading is constant during the task offloading period. In practice, the coherence time of the vehicle channel and the task offloading time are not in the same time scale. For example, when the vehicle speed is 100 km/h, and the carrier frequency is 1.8 GHz, the coherence time of the channel is about 2.5 ms. However, the task offloading time is about tens of milliseconds to hundreds of milliseconds. For some delay-tolerant services, the task offloading time can achieve a few seconds [13]. If the time-varying fading is not considered, the offloading strategies will be inaccurate, the resource utilization will be reduced, and the task delay requirements cannot be guaranteed. In addition, when considering the influence of time-varying channels, the task transmission time is related to the location of the vehicle and the allocated bandwidth. Therefore, how to allocate bandwidth

Manuscript received October 31, 2018; revised March 18, 2019, September 12, 2019, and January 2, 2020; accepted January 16, 2020. Date of publication January 20, 2020; date of current version March 12, 2020. This work was partly supported by the National Natural Science Foundation of China under Grants 61861002 and 61971030. The review of this article was coordinated by Dr. Y. Guo. *(Corresponding author: Siyu Lin.)*

S. Li is with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China, and also with the School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China (e-mail: lishichao@bjtu.edu.cn).

S. Lin, W. Li, and G. Zhu are with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: sylin@bjtu.edu.cn; 18120099@bjtu.edu.cn; gzhu@bjtu.edu.cn).

L. Cai is with the Department of Electrical and Computer Engineering, University of Victoria, BC V8W3P6, Canada (e-mail: cai@ece.uvic.ca).

Digital Object Identifier 10.1109/TVT.2020.2967882

for the time-varying channels is an important and challengeable issue [13], [14].

In this paper, we first consider the scenario that a single vehicle offloads multi-task to a single VEC server. To study the influence of time-varying channels on the resource allocation and task offloading strategies during the task offloading period, a utility maximization problem with task delay requirement constraints in VEC is formulated. Considering the time-varying channels in the vehicular network, since the spectrum efficiency of transmission is time-varying, the amount of data offloading in each time slot also changes accordingly, resulting in the inability to accurately estimate the transmission delay. It will lead to the actual task offloading delay exceeding the task delay requirement. For the formulated problem, due to the characteristic of time-varying channel, the allocated bandwidth, the SE of channel and the location of the vehicle are coupled, which makes the problem difficult to solve. To solve this problem, we propose the linearization based Branch and Bound (LBB) algorithm and the closest rounding integer (CRI) algorithm to solve the fixed spectrum efficiency (SE) problem without considering the time-varying channel characteristics. Based on the resource allocation strategies of the fixed SE problem, the LBB/CRI based computation offloading (LBBCO/CRICO) algorithms are proposed to solve the original problem for both static and dynamic tasks, in which the bandwidth allocation is adjusted dynamically according to the comparison results between the time-varying SE and fixed SE. The proposed LBBCO/CRICO algorithms are also applicable to multi-vehicle and multi-task scenarios. Furthermore, the effects of small-scale fading on the utility performance of proposed offloading strategies are evaluated. Simulation results show that the average utilities of LBBCO and CRICO algorithms have a small gap of 3.93% and 6.13% only to the upper bound, respectively.

The rest of this paper is organized as follows. In Section II, the related work is summarized. In Section III, we introduce the system model, including network model, communication model, and computation model. The system utility maximization problem is formulated. In Section IV, LBB and CRI algorithms are proposed to solve the fixed SE problem. Section V presents LBBCO/CRICO algorithms for the time-varying SE problem. We analyze the multi-vehicle and multi-task scenarios in Section VI. The effects of small-scale fading on the utility performance of the proposed resource allocation strategies are evaluated in Section VII. Numerical results and discussions are presented in Section VIII. Finally, we conclude the paper in Section IX.

II. Related Work

In the VEC system, when considering vehicular mobility, there are mainly three aspects of objectives for the joint radio and computation resource allocation in VEC system. The first is to maximize the VEC system utility. To maximize the benefit of the VEC server while enhancing the utilities of the vehicles, an efficient offloading strategy was designed through a contract theoretic approach [15]. Considering the limited computation resources and delay requirements, the authors adopted a Stackelberg Game approach to design an optimal multilevel offloading scheme, which maximizes the utilities of both the vehicles and the computing servers [16]. In [11], the authors considered a problem of joint bandwidth and computation resources to maximize the utility of cloud service providers (SPs) in cloud-enabled vehicular networks, and proposed a coalition game model based on two-sided matching theory for cooperation among cloud SPs to share their idle resources. Cordeschi *et al.* studied the soft data fusion and cognitive radio in vehicular networks, and proposed a distributed resource management strategy to maximize the system utility [12]. Considering the movement of vehicles in different RSUs, Yu *et al.* studied radio resource allocation and virtual machine migration in the cloud-based vehicular network, and proposed a game-theoretical approach to maximize the system utility [17]. To provide better service to IoT users, a joint radio and computational resource allocation scheme was proposed to maximize the system utility and improve user satisfaction [18]. The second is to minimize the cost of offloading tasks. Zhang *et al.* presented a predictive combination mode relegation scheme to reduce the cost of offloading tasks in cloud-based VEC networks [7]. When the vehicles are randomly distributed along the road, a joint radio, caching and computation resource allocation strategy in VEC was proposed to minimize the system cost [19]. To minimize the cost of vehicular terminals and VEC server at the same time, a joint offloading decision and local computation resource allocation scheme on the side of vehicular terminals was proposed. Meanwhile, a joint radio and VEC server computation resource allocation scheme on the VEC server side was proposed [20]. Considering the IoT and vehicular computing has emerged due to smart cities, a joint resource allocation and computation resource allocation scheme was proposed to support delay-sensitive applications and to reduce the workload on the backend networks [21]. To maximize the overall communication-plus-computing energy efficiency, an energy-efficient adaptive resource scheduler for real-time vehicular network was proposed [22]. Zhou *et al.* studied the energy-efficient workload offloading problem in vehicular network and proposed a low-complexity distributed solution based on consensus alternating direction method of multipliers (ADMM) [23]. The third is to minimize the offloading delay of the tasks. By combining vehicular delay-tolerant networks and VEC paradigm, a store-and-carry forward offloading mechanism was proposed to reduce the delay requirement [24]. Wang *et al.* presented a content placement algorithm to minimize the latency of mobile users [25]. To minimize the application processing time in VEC system, an iteration algorithm based on Semi-Markov decision process (SMDP) approach was proposed [26].

However, the effect of fast time-varying channels during the task offloading period was not considered in previous work, which means the channel is time-varying during the task offloading. During the offloading period, due to the fast time-varying fading caused by the high-speed movement, the offloading delay of the task changes significantly [13], [14]. The previous offloading strategies should be re-investigated and enhanced for vehicular edge computing due to dynamic environments, which motivated this work.

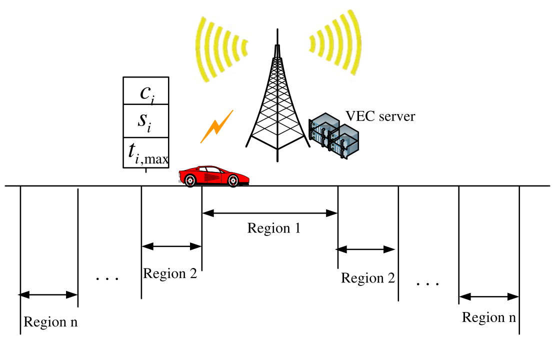


Fig. 1. System model.

III. System Model and Problem Formulation

In this section, the system model of VEC is described, including network model, communication model, and computation model. Then, we formulate the problem of joint radio, and computation resource allocation, with the objective to maximize the system utility while guaranteeing the delay requirement of the tasks.

A. Network Model

We consider a unidirectional road, each road side unit (RSU) equipped with a VEC server is deployed along the road to provide radio coverage as shown in Fig. 1. The coverage diameter of RSU is D. The vehicle speed is v and the period of the vehicle crossing one cell covered by RSU is $T = D/v$.

The influence of the time-varying channel on the system cannot be ignored [27], [28]. The channel condition, which mainly includes path loss and fading, cannot be regarded as a constant when the tasks are offloaded. Broadband wireless systems typically support adaptive modulation and coding techniques to adjust the data rate according to the received signal-to-noise ratio (SNR). Further, the wireless channel quality between the RSU and the moving vehicle highly depends on the path loss, which is a function of the communication distance d. As the communication distance increases, SE decreases. For example, when the vehicle is located on the edge of the cell, SE is minimal. Here, we focus on the path loss and divide the small cell into n region according to the distance between the vehicle and the RSU. The SE for each region can be written as $\mathcal{R} = \{r_1, \ldots, r_n\}$ [29], [30]. The statistical information of path loss in each region can be regarded as the expected value of SNR in each region. We first utilize statistical information for resource management. And then, the influence of random fading on task offloading strategies is studied in Section VI.

B. Communication Model

We consider a vehicle with I independent tasks and the tasks set is denoted as $\mathcal{I} = \{1, \ldots, I\}$. For each computation task, it can be accomplished either on the vehicle locally or on the VEC server by vehicle offloading. We denote $\alpha_i \in \{0, 1\}$ as the computation offloading decision of task i. Specifically, if task i is determined to offload to the VEC server, $\alpha_i = 1$. Otherwise, task i is chosen to compute locally, $\alpha_i = 0$. We denote the offloading vector as $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_i, \ldots, \alpha_I\}$.

To avoid interference, the spectrum within one RSU is orthogonally assigned to each task [31]. We denote b_i as the bandwidth allocated to task i. $\boldsymbol{b} = \{b_1, \ldots, b_i, \ldots, b_I\}$ is the bandwidth vector. The whole available spectrum bandwidth is B. Therefore, $\sum_{i=1}^{I} b_i \leq B$.

C. Computation Model

For the computation model, we use a tuple $\{c_i, s_i, t_{i,\max}\}$ to represent the task i, where c_i is the number of CPU-cycle required to accomplish task i. s_i is the computation file size of task i, and $t_{i,\max}$ is the delay tolerance of task i [7].

1) Local Computing: We denote $f^l_{\max}$ as the maximal CPU-cycle frequency of the vehicle, and define f^l_i as the CPU-cycle frequency allocated to task i. If task i is computed locally by the vehicle, the computation time t_{i0} can be expressed as

$$t_{i0} = \frac{c_i}{f^l_i}. \tag{1}$$

2) VEC Server Computing: For the VEC server computing, a task is offloaded to the VEC server through the wireless link first, then the VEC server executes the computation task. The time consumption for task offloading includes two parts: transmission time by the vehicle and computation time on the VEC server.

According to the communication model, when the vehicle offloads task i to the VEC server, it should satisfy

$$\sum_{j=1}^{y_i} b_i r_{i,j} \Delta t = s_i, \tag{2}$$

where y_i is the end time slot of task i offloading to the VEC server. $r_{i,j}$ is the SE of task i in the task transmission region of time slot j. Δt is the time slot duration. Because the start time slot of offloading is known, the transmission time is

$$t^{\text{trans}}_i = (y_i - 1)\Delta t. \tag{3}$$

The VEC server will execute the computation task after transmission. We define $f^s_{\max}$ as the maximal CPU-cycle frequency of the VEC server, and define f^s_i as the CPU-cycle frequency allocated to task i by VEC server. The CPU-cycle frequency allocation vector is $\boldsymbol{f^s} = \{f^s_1, \ldots, f^s_i, \ldots, f^s_I\}$. The computation time of VEC server on task i is

$$t^{com}_i = \frac{c_i}{f^s_i}. \tag{4}$$

Therefore, the offloading time of task i can be expressed as

$$t^{\text{off}}_i = (y_i - 1)\Delta t + \frac{c_i}{f^s_i}. \tag{5}$$

In this paper, because of the size of computation outcome data is much smaller than that of the computation input data, we ignore the time consumption of task computation outcome transmission from the VEC server to the vehicle. In addition, for the task offloading strategies, the vehicle can offload the task basic information (such as the number of CPU-cycle required, computation file size and delay tolerance) to the VEC server firstly. And then, the VEC server executes the algorithm and returns the results to the vehicle. Due to the VEC server has powerful computation capability, the size of task basic information and computation outcome are very small, the computation latency can be ignored in this paper [7].

D. Utility Maximization Problem Formulation

The network utility can be viewed as the offloading task number to the VEC server. The tasks can be computed either locally or by the VEC server. However, for autonomous driving of intelligent transportation systems (ITS) application, offloading some tasks to the VEC server can be combined with historical data for better results, such as map navigation and automatic vehicle tracking, etc [32], [33]. The VEC servers can exchange the processing results of these tasks and achieve vehicle scheduling for the whole network. The network utility maximization problem can be formulated as

$$(\mathbf{P1}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{f^s},\boldsymbol{b},\{y_i\}} \sum_{i=1}^{I} \alpha_i \tag{6a}$$

$$\text{s.t.} \quad \alpha_i\left((y_i-1)\Delta t + \frac{c_i}{f_i^s}\right) + (1-\alpha_i)t_{i0} \le t_{i,\max}, \forall i \in \mathcal{I}, \tag{6b}$$

$$0 \le f_i^s \le \alpha_i f_{\max}^s, \quad \forall i \in \mathcal{I}, \tag{6c}$$

$$\sum_{i=1}^{I} f_i^s \le f_{\max}^s, \tag{6d}$$

$$0 \le b_i \le \alpha_i B, \quad \forall i \in \mathcal{I}, \tag{6e}$$

$$\sum_{i=1}^{I} b_i \le B, \tag{6f}$$

$$\sum_{j=1}^{y_i} b_i r_{i,j}\Delta t = s_i, \quad \forall i \in \mathcal{I}, \tag{6g}$$

$$\alpha_i \in \{0,1\}, \quad \forall i \in \mathcal{I}. \tag{6h}$$

Constraint (6b) gives the delay requirement of the task. Constraints (6c) and (6d) are the computation resources constraints. Constraints (6e) and (6f) denote the bandwidth constraints. Constraint (6g) gives offloading data size requirement. Constraint (6h) is the offloading indicator constraint.

Considering the limited radio and computation resources, when there are a large number of tasks to be computed, some task requests can be rejected by utilizing admission control, so that the admission tasks can be computed efficiently.

For constraint (6g), it can be expressed as

$$\sum_{j=1}^{y_i} r_{i,j} = \frac{s_i}{b_i \Delta t}. \tag{7}$$

In this paper, we assume that the bandwidth allocated to the tasks remains constant throughout the transmission process. Considering that the variables y_i and b_i are coupled, and α_i is a binary variable, problem (**P1**) is a mixed-integer nonlinear programming problem.

IV. Problem Solution With Fixed SE

In order to solve problem (**P1**), we propose LBBCO algorithm which includes two steps. The first step is to propose the LBB algorithm to solve the fixed SE problem without considering the time-varying channel characteristics. The second step is to propose LBBCO algorithm by utilizing the results of LBB algorithm. Considering the complexity of LBB algorithm is high, the CRI algorithm is proposed. By utilizing the results of CRI algorithm, CRICO algorithms are proposed.

A. Fixed SE Problem

Because the variables y_i and b_i are coupled, which makes the problem difficult to solve. We need to simplify Eq. (6g) firstly. Since the moving track of the vehicle is certain, the location of the vehicle can be predicted. We utilize the fixed SE of the vehicle in each region instead of the time-varying SE.

We denote $r_{i,n}^{fix}$ as the SE in the region n of task i at the beginning offloading time slot, and utilize $r_{i,n}^{fix}$ to replace the time-varying r_i. The problem can be reformulated as

$$(\mathbf{P2}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{f^s},\boldsymbol{b}} \sum_{i=1}^{I} \alpha_i \tag{8a}$$

$$\text{s.t.} \quad \alpha_i\left(\frac{s_i}{b_i r_{i,n}^{fix}} + \frac{c_i}{f_i^s}\right) + (1-\alpha_i)\frac{c_i}{f_i^l} \le t_{i,\max}, \forall i \in \mathcal{I}, \tag{8b}$$

$$(6c)-(6f), (6h). \tag{8c}$$

B. Constraint Linearization

Due to the constraints of problem of (**P2**) are nonconvex constraints, we first linearize them. To avoid the denominator to be zero, we introduce two microscales ε_1 and ε_2 for (8b), the problem can be reformulated as

$$(\mathbf{P3}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{f^s},\boldsymbol{b}} \sum_{i=1}^{I} \alpha_i \tag{9a}$$

$$\text{s.t.} \quad \alpha_i\left(\frac{s_i}{(b_i+\varepsilon_1) r_{i,n}^{fix}} + \frac{c_i}{f_i^s+\varepsilon_2}\right) + (1-\alpha_i)\frac{c_i}{f_i^l} \le t_{i,\max}, \quad \forall i \in \mathcal{I}, \tag{9b}$$

$$(6c)-(6f), (6h). \tag{9c}$$

Lemma 1: The solution of problem (**P3**) is the upper bound of problem (**P2**).

Proof: The proof of Lemma1 is in the Appendix A. ■

Problem (**P3**) is sensible to ε_1 and ε_2 for obtaining an upper bound of problem (**P2**). We define two auxiliary variables, $\beta_i = \frac{1}{b_i+\varepsilon_1}$, $\gamma_i = \frac{1}{f_i^s+\varepsilon_2}$. Problem (**P3**) can be reformulated as

$$(\mathbf{P4}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma}} \sum_{i=1}^{I} \alpha_i \tag{10a}$$

$$\text{s.t.} \quad \alpha_i\left(\frac{s_i}{r_{i,n}^{fix}}\beta_i + c_i\gamma_i\right) + (1-\alpha_i)\frac{c_i}{f_i^l} \le t_{i,\max}, \quad \forall i \in \mathcal{I}, \tag{10b}$$

$$\frac{1}{\alpha_i f_{\max}^s + \varepsilon_2} \le \gamma_i \le \frac{1}{\varepsilon_2}, \quad \forall i \in \mathcal{I}, \tag{10c}$$

$$\sum_{i=1}^{I}\frac{1}{\gamma_i}\le f^s_{\max}+I\varepsilon_2, \tag{10d}$$

$$\frac{1}{\alpha_i B+\varepsilon_1}\le\beta_i\le\frac{1}{\varepsilon_1},\quad\forall i\in\mathcal{I}, \tag{10e}$$

$$\sum_{i=1}^{I}\frac{1}{\beta_i}\le B+I\varepsilon_1, \tag{10f}$$

$$\alpha_i\in\{0,1\}. \tag{10g}$$

Problem (**P4**) is not a convex problem, because, first, the α_i is a binary variable, and second, there are $\alpha_i\beta_i$ and $\alpha_i\gamma_i$ in problem (**P4**). In order to solve this problem, for $\alpha_i\in\{0,1\}$, we relax α_i as $0\le\alpha_i\le1$. For $\alpha_i\beta_i$, we introduce $\mu_i=\alpha_i\beta_i$, due to $0\le\alpha_i\le1$, considering $\frac{1}{\alpha_i B+\varepsilon_1}\le\beta_i\le\frac{1}{\varepsilon_1}$, the bound of β_i is $\frac{1}{B+\varepsilon_1}\le\beta_i\le\frac{1}{\varepsilon_1}$ [34], [35]. We can get the linearization bound factor product constraints of μ_i

$$\mu_i-\frac{1}{B+\varepsilon_1}\alpha_i\ge0,\quad\forall i\in\mathcal{I}, \tag{11a}$$

$$\beta_i-\frac{1}{B+\varepsilon_1}-\mu_i+\frac{1}{B+\varepsilon_1}\alpha_i\ge0,\quad\forall i\in\mathcal{I}, \tag{11b}$$

$$\frac{1}{\varepsilon_1}\alpha_i-\mu_i\ge0,\quad\forall i\in\mathcal{I}, \tag{11c}$$

$$\frac{1}{\varepsilon_1}-\beta_i-\frac{1}{\varepsilon_1}\alpha_i+\mu_i\ge0,\quad\forall i\in\mathcal{I}. \tag{11d}$$

For $\alpha_i\gamma_i$, we define $\omega_i=\alpha_i\gamma_i$. Due to $0\le\alpha_i\le1$ and $\frac{1}{f^s_{\max}+\varepsilon_2}\le\gamma_i\le\frac{1}{\varepsilon_2}$ similar as (11), the bound factor product of ω_i can be written as

$$\omega_i-\frac{1}{f^s_{\max}+\varepsilon_2}\alpha_i\ge0,\quad\forall i\in\mathcal{I}, \tag{12a}$$

$$\gamma_i-\frac{1}{f^s_{\max}+\varepsilon_2}-\omega_i+\frac{1}{f^s_{\max}+\varepsilon_2}\alpha_i\ge0,\quad\forall i\in\mathcal{I}, \tag{12b}$$

$$\frac{1}{\varepsilon_2}\alpha_i-\omega_i\ge0,\quad\forall i\in\mathcal{I}, \tag{12c}$$

$$\frac{1}{\varepsilon_2}-\gamma_i-\frac{1}{\varepsilon_2}\alpha_i+\omega_i\ge0,\quad\forall i\in\mathcal{I}. \tag{12d}$$

By substituting μ_i and ω_i into problem (**P4**), which can be written as

$$(\mathbf{P5})\quad\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\mu},\boldsymbol{\omega}}\sum_{i=1}^{I}\alpha_i \tag{13a}$$

$$\text{s.t.}\quad\left(\frac{s_i}{r^{fix}_{i,n}}\mu_i+c_i\omega_i\right)+(1-\alpha_i)\frac{c_i}{f^l_i}\le t_{i,\max},\quad\forall i\in\mathcal{I}, \tag{13b}$$

$$0\le\alpha_i\le1\quad\forall i\in\mathcal{I}, \tag{13c}$$

$$(10c)-(10f),(11),(12). \tag{13d}$$

Therefore, the problem is relaxed as a convex problem. The optimal value of problem (**P5**) is $\bar{A}$, which is the upper bound of problem (**P2**).

We define $\mathcal{I}_1=\{i|i\in\mathcal{I},\alpha_i=1\}$ and $\mathcal{I}_0=\{i|i\in\mathcal{I},\alpha_i=0\}$. When $\boldsymbol{\alpha}$ is determined, problem (**P2**) can be rewritten as

$$(\mathbf{P6})\quad\max_{\boldsymbol{f^s},\boldsymbol{b}}\|\mathcal{I}_1\|_1 \tag{14a}$$

$$\text{s.t.}\quad\frac{s_i}{b_i r^{fix}_{i,n}}+\frac{c_i}{f^s_i}\le t_{i,\max},\quad\forall i\in\mathcal{I}_1, \tag{14b}$$

$$(6c)-(6f). \tag{14c}$$

The optimal value of problem (**P6**) is $\underline{A}$, which is the lower bound of problem (**P2**).

Problem (**P5**) and problem (**P6**) are convex problems. Therefore, we can use standard packages such as CVX to solve them.

C. Problem Solution Based on Branch and Bound Method

By constraint linearization, problem (**P5**) becomes a linear problem. In this subsection, we propose a linearization based Branch and Bound (LBB) algorithm to solve problem (**P2**) [36], [37].

We define A^* as the optimal value of problem (**P2**) and corresponding resource allocation strategy is $\{\boldsymbol{\alpha}^*,\boldsymbol{f}^{s*},\boldsymbol{b}^*\}$. The optimal value of problem (**P5**) is $\bar{A}$, and corresponding resource allocation strategy is $\{\bar{\boldsymbol{\alpha}},\bar{\boldsymbol{\beta}},\bar{\boldsymbol{\gamma}},\bar{\boldsymbol{\mu}},\bar{\boldsymbol{\omega}}\}$. We denote the upper bound of A^* as $U_1=\bar{A}$. In order to obtain the lower bound of offloading vector $\underline{\boldsymbol{\alpha}}$, we can utilize the following method

$$\underline{\boldsymbol{\alpha}}=\left\{\underline{\alpha}_i\mid\underline{\alpha}_i=\begin{cases}1 & \bar{\alpha}_i>0.5\\0 & \bar{\alpha}_i\le0.5\end{cases}\forall\bar{\alpha}_i\in\bar{\boldsymbol{\alpha}}\right\}. \tag{15}$$

Based on (15), we can determine $\mathcal{I}_1$ and $\mathcal{I}_0$, and obtain the optimal value $\underline{A}$ of problem (**P6**). We denote a lower bound of A^* as $L_1=\underline{A}$. The resource allocation strategy corresponding to L_1 is $SL_1=\{\underline{\boldsymbol{\alpha}},\underline{\boldsymbol{f^s}},\underline{\boldsymbol{b}}\}$. If $U_1-L_1\le\epsilon$, where ϵ is the tolerance, the search can be terminated. Otherwise, we select another leaf node for further branching.

After that, we need to go to branch. We assume that the branching process is the lth branching. When we pick the node k with the maximal depth, the left and right leaf node problems can be formed. The first problem is

$$(\mathbf{P7})\quad\max_{\boldsymbol{\alpha},\boldsymbol{f^s},\boldsymbol{b}}\sum_{i=1}^{I}\alpha_i \tag{16a}$$

$$\text{s.t.}\quad(6b)-(6f),(6h) \tag{16b}$$

$$\{\alpha_1,\alpha_2,\ldots,\alpha_{|d(k)|},\alpha_{|d(k)|+1}\}=\{d(k),1\}, \tag{16c}$$

and the second problem is

$$(\mathbf{P8})\quad\max_{\boldsymbol{\alpha},\boldsymbol{f^s},\boldsymbol{b}}\sum_{i=1}^{I}\alpha_i \tag{17a}$$

$$\text{s.t.}\quad(6b)-(6f),(6h), \tag{17b}$$

$$\{\alpha_1,\alpha_2,\ldots,\alpha_{|d(k)|},\alpha_{|d(k)|+1}\}=\{d(k),0\}, \tag{17c}$$

where $d(k)$ is the determined computation offloading strategy of node k. $\mid d(k)\mid$ is the element number of $d(k)$. Problem (**P7**) and (**P8**) are the left and right leaf node of node k, respectively. We denote $\mathcal{I}_d$ as the determined computation offloading strategies set, $\mathcal{I}_{d0} = \{i \mid i \in \mathcal{I}_d, \alpha_i = 0\}$ and $\mathcal{I}_{d1} = \{i \mid i \in \mathcal{I}_d, \alpha_i = 1\}$. Based on linearization and $\boldsymbol{\alpha}$ relaxation in the last section, the corresponding convex problems (**P7**) and (**P8**) are

$$(\mathbf{P9}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\mu},\boldsymbol{\omega}} \|\mathcal{I}_{d1}\|_1 + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \alpha_g \tag{18a}$$

$$\text{s.t.} \quad (6c), (6e), \quad \forall i \in \mathcal{I}_d. \tag{18b}$$

$$(10c), (10e), (11), (12), (13b), (13c), \quad \forall i \in \mathcal{I} \setminus \mathcal{I}_d, \tag{18c}$$

$$(14b), \quad \forall i \in \mathcal{I}_{d1}, \tag{18d}$$

$$\sum_{i\in\mathcal{I}_{d1}} b_i + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \left(\frac{1}{\beta_g} - \varepsilon_1\right) \le B, \tag{18e}$$

$$\sum_{i\in\mathcal{I}_{d1}} f_i^s + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \left(\frac{1}{\gamma_g} - \varepsilon_2\right) \le f_{\max}^s, \tag{18f}$$

$$\mathcal{I}_d = \{d(k), 0\}. \tag{18g}$$

$$(\mathbf{P10}) \quad \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\mu},\boldsymbol{\omega}} \|\mathcal{I}_{d1}\|_1 + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \alpha_g \tag{19a}$$

$$\text{s.t.} \quad (6c), (6e), \quad \forall i \in \mathcal{I}_d. \tag{19b}$$

$$(10c), (10e), (11), (12), (13b), (13c), \quad \forall i \in \mathcal{I} \setminus \mathcal{I}_d, \tag{19c}$$

$$(14b), \quad \forall i \in \mathcal{I}_{d1}, \tag{19d}$$

$$\sum_{i\in\mathcal{I}_{d1}} b_i + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \left(\frac{1}{\beta_g} - \varepsilon_1\right) \le B, \tag{19e}$$

$$\sum_{i\in\mathcal{I}_{d1}} f_i^s + \sum_{g\in\mathcal{I}\setminus\mathcal{I}_d} \left(\frac{1}{\gamma_g} - \varepsilon_2\right) \le f_{\max}^s, \tag{19f}$$

$$\mathcal{I}_d = \{d(k), 1\}. \tag{19g}$$

These two problems are the upper bound of the left and right leaf node. We can solve these two problems and obtain the optimal values $\bar{A}_l$ and $\bar{A}_{l+1}$, respectively. The corresponding offloading computation indicators are $\bar{\boldsymbol{\alpha}}_l$ and $\bar{\boldsymbol{\alpha}}_{l+1}$, respectively. Based on (15), we can obtain $\underline{\boldsymbol{\alpha}}_l$ and $\underline{\boldsymbol{\alpha}}_{l+1}$. And then, we can obtain the lower bound $\underline{A}_l$ of left node problem, and the corresponding resource allocation strategy is $\underline{SL}_l = \{\underline{\boldsymbol{\alpha}}_l, \underline{\boldsymbol{f}}_l^s, \underline{\boldsymbol{b}}_l\}$. In the same way, we can obtain the lower bound $\underline{A}_{l+1}$ of right node problem, and the corresponding resource allocation strategy is $\underline{SL}_{l+1} = \{\underline{\boldsymbol{\alpha}}_{l+1}, \underline{\boldsymbol{f}}_{l+1}^s, \underline{\boldsymbol{b}}_{l+1}\}$.

And then, we need to calculate the upper bound U_{l+1} and lower bound L_{l+1} of A^*. U_{l+1} is the maximal upper bound of all unpruned leaf nodes. Meanwhile, L_{l+1} is the maximal lower bound of all unpruned leaf nodes. The leaf nodes whose upper bounds are smaller than L_{l+1} need to be pruned. After that, we update $l = l + 1$ and continue the process until $U_l - L_l \le \epsilon$. The result is $A^* = L_l$, and the corresponding resource allocation strategy is $SL_l = \{\boldsymbol{\alpha}^*, \boldsymbol{f}^{s*}, \boldsymbol{b}^*\}$. The proposed LBB method is shown in Algorithm 1.

Algorithm 1: Proposed LBB Algorithm.

```
1:  Initialize all the system parameters;
2:  Initialize $l = 1$, $U_1 = \bar{A}$, $L_1 = \underline{A}$;
3:  while $U_l - L_l > \epsilon$ do
4:     Select $k = \arg\max_{i\in\mathcal{U}_l} \bar{A}_i$, and split the leaf node
       into two subproblems;
5:     Obtain $\bar{A}_l$, $\bar{A}_{l+1}$, $\bar{\boldsymbol{\alpha}}_l$ and $\bar{\boldsymbol{\alpha}}_{l+1}$ by solving the
       problem (P9) and problem (P10), respectively;
6:     Obtain $\underline{\boldsymbol{\alpha}}_l$ and $\underline{\boldsymbol{\alpha}}_{l+1}$ according to (15), and calculate
       $\underline{A}_l$, $\underline{A}_{l+1}$, $\underline{SL}_l$ and $\underline{SL}_{l+1}$;
7:     $\mathcal{U}_{l+1}^p = \{i | i \in \mathcal{U}_{l+1}^{up}, \bar{A}_i < L_l\}$;
8:     $\mathcal{U}_{l+1}^{up} = \mathcal{U}_{l+1}^{up} \setminus \mathcal{U}_{l+1}^p$;
9:     $U_{l+1} = \max_{i\in\mathcal{U}_{l+1}^{up}} \bar{A}_i$;
10:    $L_{l+1} = \max_{i\in\mathcal{U}_{l+1}^{up}} \underline{A}_i$, $SL_{l+1} = \underline{SL}_{\arg\max_{i\in\mathcal{U}_{l+1}^{up}} \underline{A}_i}$;
11:    $l = l + 1$;
12: end while
13: $A^* = L_l$;
14: $\{\boldsymbol{\alpha}^*, \boldsymbol{b}^*, \boldsymbol{f}^{s*}\} = SL_l$;
```

For the Branch and Bound algorithm, the complexity grows exponentially in the worst case [38]. Because problem (**P5**), problem (**P6**), problem (**P9**) and problem (**P10**) are all convex optimization problems, we denote the complexity of these problems by $\mathcal{O}(\mathcal{C}1)$, $\mathcal{O}(\mathcal{C}2)$, $\mathcal{O}(\mathcal{C}3)$ and $\mathcal{O}(\mathcal{C}4)$, respectively. They are all polynomials. For line 2, the complexity of calculating the initial value is $\mathcal{O}(\mathcal{C}1 + \mathcal{C}2)$. The while loop has 2^I iterations in the worst case. For lines 4-6, the complexity is $\mathcal{O}(\mathcal{C}3 + \mathcal{C}4 + 2\mathcal{C}2)$. For lines 7-10, the complexity is $\mathcal{O}(4|\mathcal{U}_{l+1}^{up}|)$, where $|\mathcal{U}_{l+1}^{up}| = l + 1$ in the worst case. Therefore, the complexity of the proposed Algorithm 1 is

$$\begin{aligned} &\mathcal{O}(\mathcal{C}1 + \mathcal{C}2 + 2^I(\mathcal{O}(\mathcal{C}3 + \mathcal{C}4 + 2\mathcal{C}2) + \mathcal{O}(1 + 2^I))) \\ &= \mathcal{O}(\mathcal{C}1 + 2^I \max\{\mathcal{C}2, \mathcal{C}3, \mathcal{C}4\} + 2^{2I}) \\ &= \mathcal{O}(2^{2I}). \end{aligned} \tag{20}$$

D. CRI Algorithm

According to the last subsection, we can find that although LBB algorithm can solve problem (**P2**) accurately, the time complexity is high. In order to reduce the complexity, we propose a closest rounding integer (CRI) algorithm, which can obtain the resource allocation and task offloading strategies with low complexity. The idea of the proposed CRI algorithm is as follows: First, we solve problem (**P5**) and obtain the optimal offloading strategy $\bar{\boldsymbol{\alpha}}$, which is the upper bound of problem (**P2**). Then, we utilize the CRI algorithm to obtain $\boldsymbol{\alpha}$. If $\bar{\alpha}_i > 0.5$, we set $\alpha_i = 1$. Otherwise, $\alpha_i = 0$. In order to guarantee the feasibility of the offloading strategies, we substitute $\boldsymbol{\alpha}$ into problem (**P6**). If problem (**P6**) is feasible, $\boldsymbol{\alpha}$ are feasible offloading strategies. Otherwise, there is insufficient resources to support so many tasks offloading, so some tasks should be computed locally. We sort the elements of $\bar{\boldsymbol{\alpha}}$ which satisfy $\bar{\alpha}_i > 0.5$ in the ascending order, set the least $\bar{\alpha}_i$ as $\alpha_i = 0$. After that, substitute the offloading strategies into problem (**P6**), until problem (**P6**) is feasible.

Algorithm 2: Proposed CRI Algorithm.

```
1: Initialize all the system parameters;
2: Obtain $\boldsymbol{\alpha}$ by utilizing (21);
3: Substitute (21) into problem (**P6**);
4: **if** problem (**P6**) is feasible **then**
5:    We can obtain feasible resource allocation and task offloading strategies;
6: **else**
7:    Update $\boldsymbol{\alpha}$ until problem (**P6**) is feasible;
8: **end if**
```

The CRI algorithm is elaborated as follows:

Step 1: The optimal offloading strategies $\bar{\boldsymbol{\alpha}}$ of problem (**P5**) is the upper bound of problem (**P2**), and $0 \leq \bar{\alpha}_i \leq 1$. We can simply utilize the following method to obtain $\boldsymbol{\alpha}$.

$$\boldsymbol{\alpha} = \left\{ \alpha_i \mid \alpha_i = \begin{cases} 1 & \bar{\alpha}_i > 0.5 \\ 0 & \bar{\alpha}_i \leq 0.5 \end{cases} \forall \bar{\alpha}_i \in \bar{\boldsymbol{\alpha}} \right\}. \tag{21}$$

Step 2: In order to guarantee the feasibility of the offloading strategies. Substitute (21) into problem (**P6**).

1) If problem (**P6**) is feasible, we can obtain feasible resource allocation and task offloading strategies.
2) If problem (**P6**) is infeasible, it means that there is insufficient resources to support so many tasks offloading, some tasks should be computed locally. We sort the elements of $\bar{\boldsymbol{\alpha}}$ which satisfy $\bar{\alpha}_i > 0.5$ in the descending order, set the least $\bar{\alpha}_i$ as $\alpha_i = 0$. After that, substitute the offloading strategies into problem (**P6**), until problem (**P6**) is feasible.

The proposed CRI algorithm is shown in Algorithm 2.

V. Heuristic Algorithms for Time-Varying SE Problem

In this section, basing on the resource allocation strategies of the fixed SE, we propose the LBB based computation offloading (LBBCO) algorithm and the CRI based computation offloading (CRICO) algorithm to solve time-varying SE problem for both the static tasks and dynamic tasks.

A. Static Task Offloading Strategies

In this subsection, we only consider the static task offloading strategies, so the total number of tasks is constant when the vehicle crosses one cell covered by a RSU. From Section IV, we can obtain the offloading time of the tasks when the tasks are decided to offload. The offloading time includes task transmission time t_i^{trans} and computation time of VEC server t_i^{com}. t_i^{com} is independent of the channel condition. However, the task transmission time t_i^{trans} changes when considering the time-varying channel characteristics. The main idea of the heuristic algorithms is as follows: we utilize the average SE of the vehicle passing through the region during the task transmission time compared with the fixed SE. If the average SE is larger than the fixed SE, it means that the resource allocation policy of the fixed SE makes bandwidth surplus. We need to calculate whether more tasks could be offloaded. Otherwise, it means that the resource allocation policy of the fixed SE makes bandwidth insufficient. Some tasks cannot be offloaded. The proposed heuristic algorithms are elaborated as follows:

Step 1: Because the vehicle speed v and the task offloading start position P_{strat} are certain, the RSU can accurately predict how long the vehicle will stay in the transmission region. We denote the time of vehicle staying in the transmission region as t_{stay}. If $t_{\text{stay}} \geq t_i^{\text{trans}}$, which means that task i is offloaded to the VEC server when the vehicle stays in one transmission region. Therefore, the resource allocation strategies can be obtained by the LBB/CRI algorithms.

Step 2: If $t_{\text{stay}} < t_i^{\text{trans}}$, it means that task i is offloaded to the VEC server when the vehicle moves to other transmission regions. Because the distance of each region is certain, by utilizing t_i^{trans}, the RSU can predict the position of task i to complete transmission. Therefore, the average SE $\bar{r}_i$ in this period of time t_i^{trans} can be calculated.

Step 3: For some tasks, if $\bar{r}_i \geq r_{i,n}^{fix}$, $\frac{s_i}{b_i \bar{r}_i} \leq \frac{s_i}{b_i r_{i,n}^{fix}}$, it means that the resource allocation policy of the fixed SE makes bandwidth surplus. We denote $b_{i,+} = \frac{s_i}{t_i^{\text{trans}} \bar{r}_i}$ as the bandwidth allocation for the task i which satisfies $\bar{r}_i \geq r_{i,n}^{fix}$. The total surplus bandwidth is $b_+ = \sum_{i \in \mathcal{I}_+} (b_i - b_{i,+})$, where $\mathcal{I}_+$ is the task set which satisfies $\bar{r}_i \geq r_{i,n}^{fix}$.

Step 4: For other tasks, if $\bar{r}_i < r_{i,n}^{fix}$, $\frac{s_i}{b_i \bar{r}_i} > \frac{s_i}{b_i r_{i,n}^{fix}}$, it means that the resource allocation policy of the fixed SE makes bandwidth insufficient. We denote $b_{i,-} = \frac{s_i}{t_i^{\text{trans}} \bar{r}_i}$ as the bandwidth allocation for the task i which satisfies $\bar{r}_i < r_{i,n}^{fix}$. The total insufficient bandwidth is $b_- = \sum_{i \in \mathcal{I}_-} (b_{i,-} - b_i)$, where $\mathcal{I}_-$ is the task set which satisfies $\bar{r}_i < r_{i,n}^{fix}$.

Step 5: We denote $b_{sur} = b_+ - b_-$. If $b_{sur} \geq 0$, it means that there is surplus bandwidth. If VEC server has surplus computation resources, we utilize the proposed LBB/CRI algorithms to calculate whether more tasks can be offloaded. If VEC server does not have surplus computation resources, no more tasks can be computed.

Step 6: If $b_{sur} < 0$, it means that some tasks cannot be offloaded to the VEC server, due to the lack of bandwidth. We sort the tasks with insufficient bandwidth in the ascending order. Remove the task with the highest deficit in bandwidth from the list to offload, the surplus bandwidth can be allocated to other tasks.

The proposed LBBCO/CRICO algorithms are shown in Algorithm 3.

B. Dynamic Task Offloading Strategies

In the previous subsection, we only considered static task offloading strategies. However, due to the change of road traffic flow, the tasks will be randomly generated. In this subsection, we propose the dynamic task offloading strategies.

The tasks follow the Poisson process with average generation rate λ [39]. Compared with static tasks, due to the random generation of tasks, the total bandwidth and computation resources of VEC server are changing at each time slot. Therefore, for the dynamic task offloading strategies, we first update the total resources at each time slot. And then, utilize the LBBCO/CRICO

Algorithm 3: Proposed LBBCO/CRICO Algorithms.

```
1:  Obtain $t_i^{\text{trans}}$, $b_i$ by utilizing LBB/CRI algorithms,
    calculate $t_{\text{stay}}$, and give the task offloading start
    position $P_{strat}$;
2:  if $t_{\text{stay}} \geq t_i^{\text{trans}}$ then
3:    The resource allocation strategies can be obtained by
      LBB/CRI algorithms;
4:  else
5:    Calculate the average SE $\bar{r}_i$;
6:    if $\bar{r}_i \geq r_{i,n}^{fix}$ then
7:      Calculate $b_{i,+}$ and $b_+$;
8:    else
9:      Calculate $b_{i,-}$ and $b_-$;
10:   end if
11:   if $b_{sur} \geq 0$ then
12:     if there is surplus computation resources in VEC
        server then
13:       Calculate whether more tasks can be offloaded;
14:     else
15:       The resource allocation strategies can be
          obtained by LBB/CRI algorithms;
16:     end if
17:   else
18:     Let the task which lack of the most bandwidth
        does not offload, the surplus bandwidth can be
        allocated to the other tasks;
19:   end if
20: end if
```

Algorithm 4: Dynamic Task Offloading Strategies.

```
1: Initialize all the system parameters;
2: while $t \in [0, T]$ do
3:   if there is surplus resources then
4:     Utilize LBBCO/CRICO algorithms to allocate
       resources to the tasks;
5:   else
6:     The tasks are calculated by the vehicle;
7:   end if
8:   Update the total resources at each time slot;
9: end while
```

algorithms to allocate resources to the tasks. The proposed dynamic task offloading strategies are shown in Algorithm 4.

C. The Convergence of the Proposed Algorithms

We analyze the convergence of the proposed algorithms. The convergence of the proposed LBBCO and CRICO algorithms are shown in Fig. 2. From this figure, the utility of proposed LBBCO algorithm increases quickly in the first 20 iterations and then enter a stable status within the first 65 iterations. For the proposed CRICO algorithm, 10 iterations lead to convergence.

VI. Multi-Vehicle and Multi-Task Scenarios

The proposed LBBCO/CRICO algorithms are also applicable to multi-vehicle and multi-task scenarios.

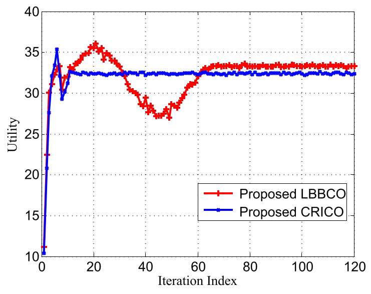


Fig. 2. Convergence progresses of the proposed algorithms.

A. Network Model

The proposed algorithms are also applicable to multi-vehicle and multi-task scenarios. There are K vehicles in this system and each vehicle has I independent tasks. We denote the set of vehicle as $\mathcal{K} = \{1, \ldots, K\}$, and the set of task in each vehicle as $\mathcal{I} = \{1, \ldots, I\}$.

B. Communication Model

We denote $\alpha_{k,i} \in \{0, 1\}$ as the computation offloading decision of the task. Specifically, if task i in vehicle k is determined to offload to the VEC server, $\alpha_{k,i} = 1$. Otherwise, task i in vehicle k is chosen to be computed locally, $\alpha_{k,i} = 0$. We denote the offloading vector as $\boldsymbol{\alpha} = \{\alpha_{1,1}, \ldots, \alpha_{k,i}, \ldots, \alpha_{K,I}\}$. In order to avoid interference, the spectrum is orthogonally assigned to each task. We denote $b_{k,i}$ as the bandwidth of VEC allocated to task i in vehicle k. $\boldsymbol{b} = \{b_{1,1}, \ldots, b_{k,i}, \ldots, b_{K,I}\}$ is the bandwidth vector. The whole available spectrum bandwidth is B. Therefore, we can get $\sum_{k=1}^{K} \sum_{i=1}^{I} b_{k,i} \leq B$.

C. Computation Model

We use a tuple $\{c_{k,i}, s_{k,i}, t_{k,i,\max}\}$ to represent the task i in vehicle k, where $c_{k,i}$ is the number of CPU-cycle required to accomplish task i in vehicle k. $s_{k,i}$ is the computation file size of task i in vehicle k, and $t_{k,i,\max}$ is the delay tolerance of task i in vehicle k.

1) Local Computing: We denote $f_{k,max}^{l}$ as the maximal CPU-cycle frequency of the vehicle k, and define $f_{k,i}^{l}$ as the CPU-cycle frequency allocated to task i in vehicle k. If task i in vehicle k is computed locally by the vehicle, the computation time $t_{k,i0}$ can be expressed as

$$t_{k,i0} = \frac{c_{k,i}}{f_{k,i}^{l}}. \tag{22}$$

2) VEC Server Computing: For the VEC server computing, a task is offloaded to the VEC server through the wireless link first, and then the VEC server executes the computation task. The time consumption of task offloading includes two parts: transmission time by the vehicle and computation time on the VEC server.

According to the communication model, when the vehicle k offloads task i to the VEC server, it should satisfy

$$\sum_{j=1}^{y_{k,i}} b_{k,i} r_{k,i,j} \Delta t = s_{k,i}, \tag{23}$$

where $y_{k,i}$ is the end time slot of task i in vehicle k offloading to the VEC server. $r_{k,i,j}$ is the SE of task i in vehicle k in the task transmission region of time slot j. Δt is the time slot duration. Because the start time slot of offloading is known, the transmission time is

$$t_{k,i}^{\text{trans}} = (y_{k,i} - 1)\Delta t. \tag{24}$$

The VEC server will execute the computation task after transmission. We define $f_{\max}^s$ as the maximal CPU-cycle frequency of the VEC server, and define $f_{k,i}^s$ as the CPU-cycle frequency allocated to task i in vehicle k by VEC server. The CPU-cycle frequency allocation vector is $\boldsymbol{f}^s = \{f_{1,1}^s, \ldots, f_{k,i}^s, \ldots, f_{K,I}^s\}$. The computation time of VEC server on task i in vehicle k is

$$t_{k,i}^{com} = \frac{c_{k,i}}{f_{k,i}^s}. \tag{25}$$

Therefore, the offloading time of task i in vehicle k can be expressed as

$$t_{k,i}^{off} = (y_{k,i} - 1)\Delta t + \frac{c_{k,i}}{f_{k,i}^s}. \tag{26}$$

D. Problem Formulation

The network utility maximization problem can be formulated as

$$\max_{\boldsymbol{\alpha}, \boldsymbol{f}^s, \boldsymbol{b}, \{y_{k,i}\}} \sum_{k=1}^{K} \sum_{i=1}^{I} \alpha_{k,i} \tag{27a}$$

$$\text{s.t.} \quad \alpha_{k,i} t_{k,i}^{off} + (1 - \alpha_{k,i}) t_{k,i0} \leq t_{k,i,max}, \forall k \in \mathcal{K}, \quad \forall i \in \mathcal{I}, \tag{27b}$$

$$0 \leq f_{k,i}^s \leq \alpha_{k,i} f_{\max}^s, \quad \forall k \in \mathcal{K}, \quad \forall i \in \mathcal{I}, \tag{27c}$$

$$\sum_{k=1}^{K} \sum_{i=1}^{I} f_{k,i}^s \leq f_{\max}^s, \tag{27d}$$

$$0 \leq b_{k,i} \leq \alpha_{k,i} B, \quad \forall k \in \mathcal{K}, \quad \forall i \in \mathcal{I}, \tag{27e}$$

$$\sum_{k=1}^{K} \sum_{i=1}^{I} b_{k,i} \leq B, \tag{27f}$$

$$\sum_{j=1}^{y_{k,i}} b_{k,i} r_{k,i,j} \Delta t = s_{k,i}, \quad \forall k \in \mathcal{K}, \quad \forall i \in \mathcal{I}, \tag{27g}$$

$$\alpha_{k,i} \in \{0, 1\}, \quad \forall k \in \mathcal{K}, \quad \forall i \in \mathcal{I}. \tag{27h}$$

Constraint (27b) gives the delay requirements of the task. Constraints (27c) and (27d) are the computation resources constraints. Constraints (27e) and (27f) denote the bandwidth constraints. Constraint (27g) gives offloading data size requirement. Constraint (27h) is the offloading indicator constraint.

For the problem, if the target VEC server of each task is determined, we can also utilize the proposed LBBCO/CRICO algorithms to solve it.

VII. Offloading Strategies Considering Small-Scale Fading

In the previous sections, we simply divided the small cell into n regions according to the distance between the vehicle and the RSU, and only considered the path loss effect on the offloading strategies. However, the small-scale fading also has a great impact on the wireless channel [40]. The small-scale fading cannot be predicted accurately, but the vehicle can obtain the statistics information of small-scale fading which remains unchanged for a long time. Because the small-scale fading can reduce SE, more bandwidth is needed when considering the small-scale fading. In this section, we evaluate the effect of small-scale fading on the proposed computation offloading algorithm.

In vehicular networks, we usually utilize Nakagami-m model to describe the small-scale fading, where m is the fading factor [41], [42]. The m factor can increase from $\frac{1}{2}$ to ∞, when $m = 1$, the fading becomes Rayleigh.

Considering flat fading, and for narrowband signals, the task transmission SE at region n can be expressed as

$$r(n) = \log_2(1 + h_s(n) h(n) P(n)), \tag{28}$$

where $\sqrt{h_s(n)}$ is small-scale fading with Nakagami-m distribution. $h(n)$ is the path loss at region n. $P(n)$ is the transmission power of the vehicle at region n.

In vehicle scenario, we are interested in satisfying small delay constraints and large data offloading rates. The coverage distance and the vertical distance are small. In this case, the SNR of the vehicle would be high even when the vehicle is on the edge of a cell [30], [42], [43]. Therefore, the task transmission SE is

$$\begin{aligned} r(n) &\approx \log_2(h_s(n) h(n) P(n)) \\ &= \log_2(h(n) P(n)) + \log_2(h_s(n)). \end{aligned} \tag{29}$$

From Eq. (29), we notice that the task transmission SE would be reduced due to the factor $\log_2(h_s(n))$. Because the period that the vehicle passes through one cell is much longer than the channel coherence time, the small-scale fading is ergodic. Therefore, we get

$$\log_2(h_s(n)) = E(\log_2(h_s(n))). \tag{30}$$

$\sqrt{h_s(n)}$ is Nakagami-m distribution with parameters Ω and m, where Ω is the average SNR and m is the fading factor. Thus, we obtain

$$E(\log_2(h_s(n))) = \left[\psi(m) + \ln \frac{\Omega}{m}\right] \log_2 e, \tag{31}$$

where ψ is Digamma function [30].

VIII. Simulation Results and Futher Discussions

In this section, we provide extensive simulation results to show the performance of the two proposed algorithms. We first consider the scenario a vehicle offloads multi-task to a VEC server, the tasks can be computed either locally or by the VEC server. In a unidirectional road, a RSU is deployed along the road every 500 m, the vehicle speed is 80 km/h. The bandwidth of the system is 10 MHz. Because the wireless SE is mainly

TABLE I
SE OF REGION

Distance region (m)	SE (b/s/Hz)
[0,50],[450,500]	0.5
[50,100],[400,450]	1
[100,150],[350,400]	1.5
[150,200],[300,350]	2
[200,250],[250,300]	2.5

determined by the relative distance, we divided the small cell into 10 regions. The wireless configuration is shown in Table I. The maximal CPU-cycle frequency of VEC server is 20 GHz/s. There are 50 tasks to be processed. The data size of the tasks and the number of CPU-cycle requirements are Gaussian distributions, $s_i \sim \mathcal{N}(1000, 100)$ and $c_i \sim \mathcal{N}(1000, 100)$, respectively. The data size is measured in KB and the number of CPU-cycle is measured in Megacycles, respectively. The delay constraint for each task is 4 s. In the simulation section, we simulate the performance by averaging 500 randomly generated tasks.

We compare the following algorithms with our proposed algorithms:

- Admission Control for Joint Communication and Computation (ACJCC) algorithm: The objective of this algorithm is to support more tasks in the edge server. We choose this algorithm for comparison since it has the same objective function as our work. This algorithm consists of two steps, the first step is to allocate communication and computation resources without considering the total resources constraints. The second step is to design the admission control mechanism when considering the total resources constraints [44].
- Minimum Overhead Offloading Algorithm (MOOA): In this algorithm, an overhead function which includes energy consumption of task transmission and task offloading delay is formulated. If the task overhead on locally is larger than that on the VEC server, the task is offloaded to the VEC server. Otherwise, the task is decided to compute locally. The channel time-varying characteristics are not taken into account in this algorithm [45].
- Minimum Data Size First Offloading Algorithm (MDSFOA): We sort the data size of tasks in the ascending order. The vehicle offloads the minimum data size firstly.
- Minimum CPU-cycle Requirement First Offloading Algorithm (MCRFOA): We sort the CPU-cycle requirement of tasks in the ascending order. The vehicle offloads the minimum CPU-cycle requirement firstly.
- Upper bound: α_i is relaxed as (0,1). First, by solving the convex problem (**P5**), we obtain the upper bound of problem (**P2**). And then, by utilizing the idea of LBBCO/CRICO algorithms and solving the convex problem (**P5**) to adjust the resource allocation strategies, we can obtain the upper bound of problem (**P1**).

A. Static Task Offloading Strategies

In this subsection, we evaluate the performance of the proposed offloading strategies for static tasks. The number of tasks is 50, and the vehicle can obtain the information of all the tasks.

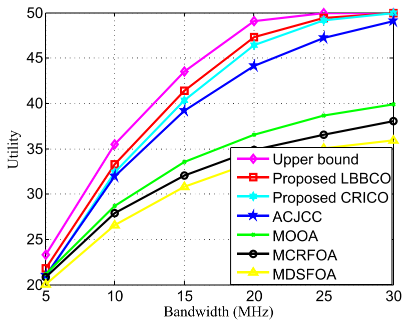


Fig. 3. The bandwidth versus system utility.

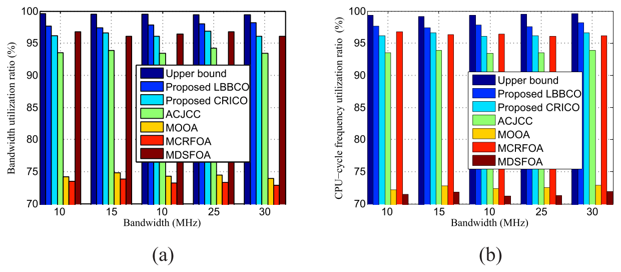


Fig. 4. Resource utilization ratio of each algorithm. (a) The bandwidth utilization ratio of each algorithm. (b) The CPU-cycle frequency utilization ratio of each algorithm.

Fig. 3 illustrates the bandwidth versus system utility. The system utility increases with respect to the bandwidth, as higher bandwidth leads to lower task transmission time and thus more tasks can be offloaded to the VEC server. From Fig. 3, when the bandwidth is 5 MHz, the utilities of LBBCO and CRICO algorithms are about 6.44% and 10.73% below the upper bound, respectively. However, when the bandwidth is 25 MHz, the gaps to the upper bound are as low as 1.2% and 1.4%, respectively. When the bandwidth reaches about 30 MHz, all the tasks can be offloaded to the VEC server. It means that radio resources are sufficient to maximize system utility. Compared to the ACJCC algorithm, when the bandwidth grows from 5 MHz to 30 MHz, the utilities of LBBCO and CRICO algorithms have performance gains of about 4.52% and 2.38%, respectively. From the simulation results, the proposed algorithms considering time-varying fading channel can improve resource utilization with better task offloading strategies.

Further results are shown in Fig. 3, we measured the resource utilization of each algorithm. Fig. 4(a) compares the bandwidth utilization ratio and Fig. 4(b) compares the CPU-cycle frequency utilization ratio. From Figs. 4(a) and 4(b), for the proposed LBBCO algorithm, the bandwidth and CPU-cycle frequency utilization ratio are about 98% and 97%, respectively. For the proposed CRICO algorithm, the bandwidth and CPU-cycle frequency utilization ratio are about 97% and 96%, respectively. If more tasks need to be offloaded, resources must be fully utilized. The resources of LBBCO and CRICO algorithms are almost fully utilized, and their performance is better than other algorithms.

Fig. 5 illustrates the CPU-cycle frequency of the VEC server versus system utility. The system utility is higher with a higher

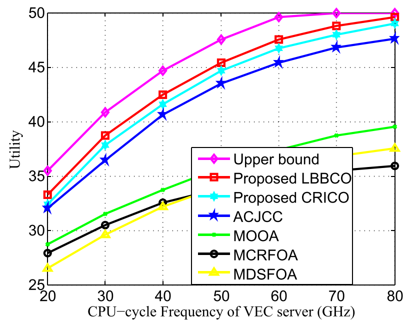


Fig. 5. The CPU-cycle frequency of VEC server versus system utility.

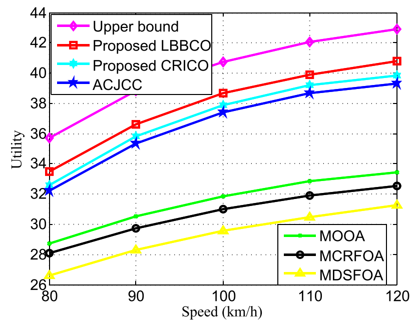


Fig. 7. The vehicle speed versus system utility when the vehicle moves from the cell edge to the center.

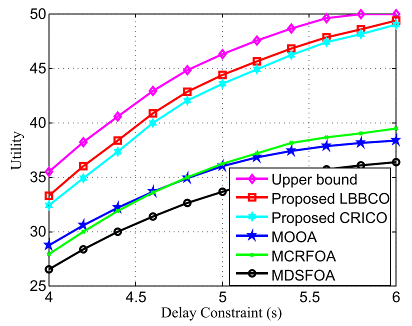


Fig. 6. The delay constraint versus system utility.

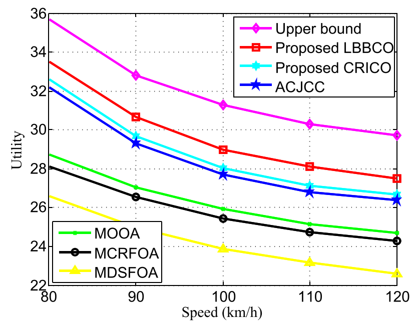


Fig. 8. The vehicle speed versus system utility when the vehicle moves from the cell center to the edge.

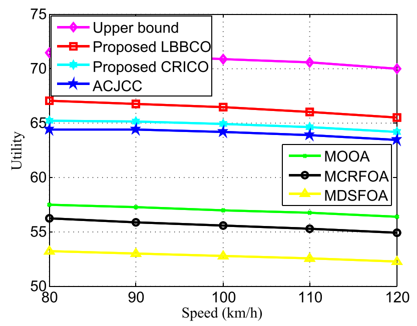


Fig. 9. The vehicle speed versus system utility.

CPU-cycle frequency of the VEC server. With the increasing of CPU-cycle frequency, the task computation time can be reduced, and more tasks can be offloaded to the VEC server. For the proposed LBBCO algorithm, when the CPU-cycle frequency reaches about 80 GHz, about 99.2% tasks can be offloaded to the VEC server, for the CRICO algorithm, about 98% of tasks can be offloaded. This figure also shows that when computation resources are sufficient, the maximization of system utility can be guaranteed.

Fig. 6 illustrates the delay constraint versus system utility, which increases with a longer delay constraint. Given a larger maximum tolerable delay, fewer bandwidth and CPU-cycle frequency resources can be allocated to a task to satisfy the delay constraint. Therefore, more tasks can be offloaded to the VEC server. For the proposed LBBCO algorithm, we can see that when the delay constraint reaches 6 s, the bandwidth and CPU-cycle frequency resources can satisfy the offloading delay requirements of about 98.7% tasks. For the CRICO algorithm, about 98% tasks can be offloaded.

Fig. 7 illustrates the vehicle speed versus system utility when the vehicle moves from the cell edge to the center. For all the algorithms, the system utility increases with the increase of the vehicle speed, as the vehicle drives faster when it is closer to the cell center. The average SE will increase in this time period. From step 3 of Algorithm 3, there will be surplus bandwidth resources in this case. Therefore, more tasks can be offloaded to the VEC server when the vehicle is close to the center.

Fig. 8 illustrates the vehicle speed versus system utility when the vehicle moves from the cell center to the edge, where we can see the opposite trend.

Fig. 9 illustrates the vehicle speed versus system utility in one cell. From this figure, we can see that the system utility decreases with the vehicle speed increasing. This is because with the increasing of vehicle speed, vehicle spends less time in each region. It will lead to a lower amount of task transmission.

Fig. 10 illustrates the utility of proposed two algorithms with different m factors. In the previous figures, we only considered the effect of path loss on offloading schemes. However, the small-scale fading cannot be eliminated. Therefore, we must evaluate the effect of small-scale fading on the offloading schemes. From this figure, it can be seen that more serious small-scale fading leads to worse system utility. That is because the more serious small-scale fading, the more SE is reduced, which leads to less tasks can be offloaded to the VEC server. It also can be seen that when bandwidth grows from 10 MHz to 30 MHz and $m = 5$, the utilities of proposed LBBCO and

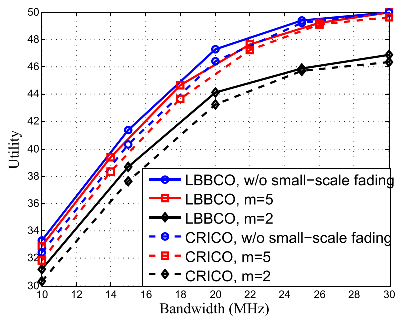


Fig. 10. The utility of proposed two algorithms with different m factors.

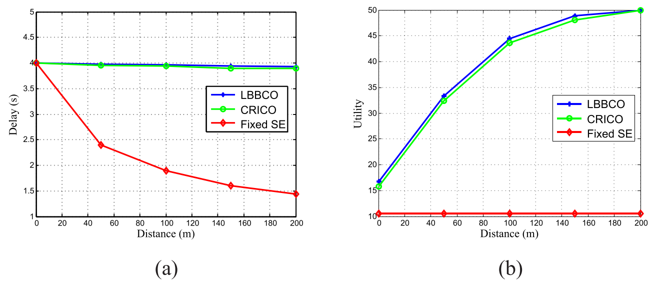


Fig. 11. Scenario: the vehicle moves from the cell edge to the center. (a) The distance versus task offloading delay under different algorithms. (b) The distance versus utility under different algorithms.

CRICO algorithms only decrease by an average of 0.9% and 1.02%, respectively. And when $m = 2$, the utilities of proposed LBBCO and CRICO algorithms decrease by an average of 7.05% and 7.35%, respectively. Therefore, the proposed algorithms are still effective in the view of the small-scale fading.

To verify the validity of the proposed algorithms, we analyze the effect of resource allocation strategies on the task offloading delay and utility. We compare the fixed SE algorithm with our proposed algorithms:

- Fixed SE Algorithm: This algorithm can be obtained in Section IV, which does not consider the time-varying channel.

Fig. 11 shows the performance when the vehicle moves from the cell edge to the center. Fig. 11(a) illustrates the distance versus task offloading delay under different algorithms. For the fixed SE algorithm, the vehicle decides the resource allocation strategies when it is at the cell edge. And it utilizes the resource allocation strategies all the time. From this figure, it can be seen that the task offloading delay decreases when the vehicle moves from the cell edge to the center. We can explain as follows: the SE increases when the vehicle moves from the cell edge to the center. The fixed SE algorithm utilizes the resource allocation strategies when the vehicle is at the cell edge, which allocates more resources to the tasks. Therefore, the task offloading delay is less than the delay requirements. It means that the resources of the fixed SE algorithm are wasted. Fig. 11(b) can verify this conclusion. Fig. 11(b) illustrates the distance versus utility under different algorithms. For the fixed SE algorithm, the system utility is about 10.1 all the time. However, when the vehicle is at the cell center, for the proposed LBBCO and CRICO algorithms, all the tasks can be offloaded to the VEC server.

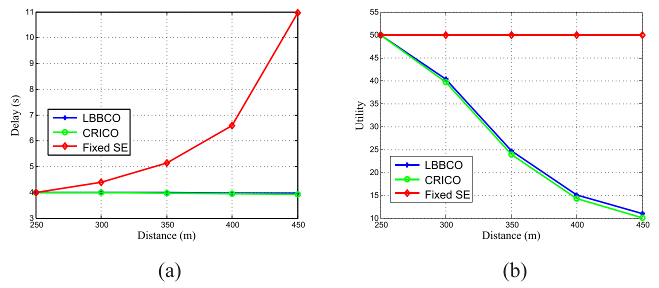


Fig. 12. Scenario: the vehicle moves from the cell center to the edge. (a) The distance versus task offloading delay under different algorithms. (b) The distance versus utility under different algorithms.

From Fig. 11, it can also be seen that the proposed LBBCO algorithm and CRICO algorithm can satisfy the task offloading delay requirements, and there are fewer resources wasted.

Fig. 12 shows the performance when the vehicle moves from the cell center to the edge. Fig. 12(a) illustrates the distance versus task offloading delay under different algorithms. For the fixed SE algorithm, the vehicle decides the resource allocation strategies when it is at the cell center. And it utilizes the resource allocation strategies all the time. From this figure, it can be seen that the task offloading delay increases when the vehicle moves from the cell center to the edge. We can explain as follows: the SE decreases when the vehicle moves from the cell center to the edge. The fixed SE algorithm utilizes the resource allocation strategies when the vehicle is at the cell center, which allocates fewer resources to the tasks. Therefore, the fixed SE algorithm cannot guarantee the task offloading delay requirements. Fig. 12(b) can verify this conclusion. Fig. 12(b) illustrates the distance versus utility under different algorithms. For the fixed SE algorithm, all the tasks can be offloaded to the VEC server. However, the task offloading delay is about 11 s when the vehicle is at the cell edge. Therefore, the resource allocation strategies of the fixed SE algorithm are infeasible. In fact, for the proposed LBBCO and CRICO algorithms, there are only about 11 tasks can be offloaded when the vehicle is at the cell edge. From this figure, it can also be seen that, the proposed LBBCO algorithm and CRICO algorithm can satisfy the task offloading delay requirements.

B. *Dynamic Task Offloading Strategies*

In this subsection, we evaluate the performance of the proposed offloading strategies for the dynamic tasks in one RSU coverage area. The tasks follow the Poisson process with average generation rate λ, and the duration of each time slot is 200 ms.

Fig. 13 illustrates the task average generation rate versus total system utility. From this figure, the total system utility first increases with the task average generation rate and then flats off. That is because the bandwidth resources and computation resources of VEC server are limited.

Fig. 14 illustrates the bandwidth versus total system utility with different m factors. The task average generation rate is 8. From this figure, it can be seen that the total system utility increases with the bandwidth increasing. However, the total system utility improvement diminishes with increase of bandwidth. That

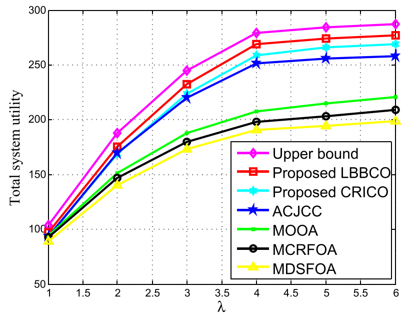


Fig. 13. The task average generation rate λ versus total system utility.

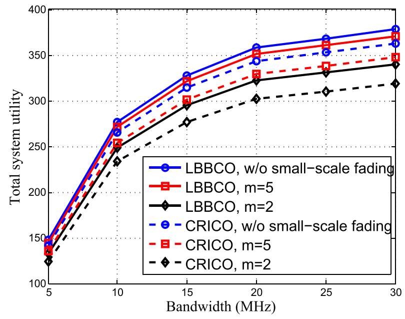


Fig. 14. The bandwidth versus total system utility with different m factors.

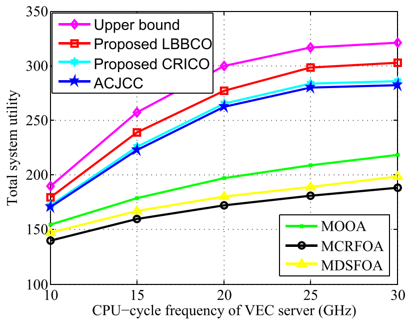


Fig. 15. The CPU-cycle frequency of VEC server versus total system utility.

is because the computation resources of VEC server are limited. It also can be seen that, when bandwidth grows from 5 MHz to 30 MHz and $m = 5$, the total system utilities of proposed LBBCO and CRICO algorithms only decrease by an average of 1.9% and 4.03%, respectively. And when $m = 2$, the total system utilities of proposed LBBCO and CRICO algorithms decrease by an average of 10.02% and 12.11%, respectively.

Fig. 15 illustrates the CPU-cycle frequency of VEC server versus total system utility. The task average generation rate is 8. From this figure, the total system utility first increases with the computation resources increasing, and then reaches saturation, as for the bandwidth resources are limited. Compared with the upper bound, when the CPU-cycle frequency of VEC server grows from 10 GHz to 30 GHz, the performance of LBBCO and CRICO algorithms get closer to the upper bound with small gaps of 6.38% and 10.88%, respectively.

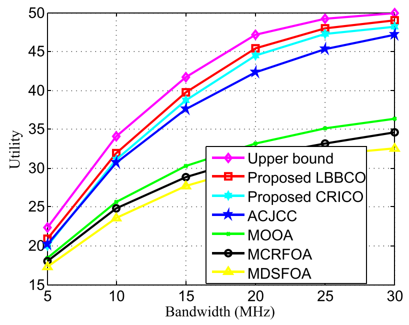


Fig. 16. The bandwidth versus total system utility in multi-vehicle and multi-task scenarios.

C. *Multi-Vehicle and Multi-Task Scenarios*

In this subsection, we evaluate the performance of the proposed offloading strategies for multi-vehicle and multi-task scenarios. There are 3 vehicles in this system, and each vehicle has 20 tasks to be processed in the RSU coverage area.

Fig. 16 illustrates the bandwidth versus system utility in multi-vehicle and multi-task scenarios. The system utility increases with the bandwidth increasing. When the bandwidth is 30 MHz, for the proposed LBBCO and CRICO algorithms, all the tasks can be offloaded to the VEC server. Compared with the ACJCC algorithm, when the bandwidth grows from 5 MHz to 30 MHz, the utilities of LBBCO and CRICO algorithms increase by an average of 4.64% and 2.36%, respectively.

IX. Conclusion and Future Works

In this paper, we investigated the joint resource allocation and computation offloading with time-varying channels in VEC. We formulated a problem of joint radio and computation resource allocation, with the objective to maximize the VEC system utility while guaranteeing the delay requirement of tasks. Because of the task transmission time and the allocated bandwidth are coupled, the problem is a nonconvex problem. In order to solve this problem, we first utilized the fixed SE instead of the time-varying SE, and then proposed LBB algorithm to solve the fixed SE problem. Considering the complexity of LBB algorithm cannot be guaranteed, CRI algorithm was proposed. After that, based on the resource allocation strategies of the fixed SE, LBBCO/CRICO algorithms were proposed to solve the time-varying SE problem for both the static tasks and dynamic tasks. The proposed LBBCO/CRICO algorithms are also applicable to multi-vehicle and multi-task scenarios. Finally, the effect of small-scale fading on the proposed offloading strategies was analyzed. Simulation results have demonstrated that the proposed two algorithms can achieve good performance. The system architecture provides a new entry point for computation offloading of time-varying channels in vehicle networks. The proposed joint resource allocation and computation offloading schemes can be used conveniently in the future IoV system.

For this paper, there are three aspects can be extended. (i) The time-varying bandwidth allocation strategies may cause some different results. We will study the offloading strategies when considering the time-varying bandwidth allocation.

(ii) There are rich D2D resources in vehicular networks, D2D communication enables multi-vehicles cooperative task offloading. Based on the first work, we will focus on the resource allocation of multi-vehicles and multi-VEC servers system by D2D communication. (iii) For the multi-vehicle and multi-server scenarios, the server selection problem should be considered. A vehicle can access the server with sufficient resources to reduce the task offloading delay and improve system utility. Furthermore, the task offloading delay should be reformulated considering three parts: the transportation time of the vehicle moving to the selected RSU, the task transmission time, and the task computation time. The interference among different RSUs should also be considered when allocating resources.

Appendix

Proof of Lemma 1: We denote the feasible solutions of problem (**P2**) and problem (**P3**) as $\mathcal{S}_2$ and $\mathcal{S}_3$, respectively. Problem (**P2**) and problem (**P3**) have the same objective function and constraints except (8b) and (9b). And constraint (8b) is stricter than (9b).

$$\alpha_i\left(\frac{s_i}{b_i r_{i,n}^{fix}}+\frac{c_i}{f_i^s}\right)+(1-\alpha_i)\frac{c_i}{f_i^l} \geq \alpha_i\left(\frac{s_i}{(b_i+\varepsilon_1)r_{i,n}^{fix}}+\frac{c_i}{f_i^s+\varepsilon_2}\right)+(1-\alpha_i)\frac{c_i}{f_i^l}. \quad (32)$$

Therefore, for any feasible solution $\mathcal{S}^{\forall}$ belonging to $\mathcal{S}_2$, it also belongs to $\mathcal{S}_3$. The optimal solution of problem (**P2**) is smaller than or equal to the optimal solution of problem (**P3**). Therefore, the solution of problem (**P3**) is the upper bound of problem (**P2**). ■

References

[1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surv. Tut.*, vol. 17, no. 4, pp. 2347–2376, Oct.–Dec. 2015.

[3] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of Vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018.

[4] H. T. Wu and G. J. Horng, "Establishing an intelligent transportation system with a network security mechanism in an Internet of Vehicle environment," *IEEE Access*, vol. 5, pp. 19239–19247, Sep. 2017.

[5] S. Bitam, A. Mellouk, and S. Zeadally, "VANET-cloud: A generic cloud computing model for vehicular Ad Hoc networks," *IEEE Wireless Commun.*, vol. 22, no. 1, pp. 96–102, Feb. 2015.

[6] Q. Gao, S. Lin, and G. Zhu, "Joint vehicular and static users multiplexing transmission with hierarchical modulation for throughput maximization in vehicular networks," *IEEE Trans. Intell. Transp. Syst.* pp. 1–13, Aug. 2019.

[7] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive off-loading," *IEEE Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Jun. 2017.

[8] H. Zhou, W. Xu, Y. Bi, J. Chen, Q. Yu, and X. S. Shen, "Toward 5G spectrum sharing for immersive-experience-driven vehicular communications," *IEEE Wireless Commun.*, vol. 24, no. 6, pp. 30–37, Dec. 2017.

[9] S. Li *et al.*, "Joint admission control and resource allocation in edge computing for Internet of Things," *IEEE Netw.*, vol. 32, no. 1, pp. 72–79, Jan.-Feb. 2018.

[10] S. Lin *et al.*, "Advanced dynamic channel access strategy in spectrum sharing 5G systems," *IEEE Wireless Commun.*, vol. 24, no. 5, pp. 74–80, Oct. 2017.

[11] R. Yu *et al.*, "Cooperative resource management in cloud-enabled vehicular networks," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7938–7951, Dec. 2015.

[12] N. Cordeschi, D. Amendola, and E. Baccarelli, "Reliable adaptive resource management for cognitive cloud vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 64, no. 6, pp. 2528–2537, Jun. 2015.

[13] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2322–2358, Oct.–Dec. 2017.

[14] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surv. Tut.*, vol. 19, no. 3, pp. 1628–1656, Jul.–Sep. 2017.

[15] K. Zhang, Y. Mao, S. Leng, A. Vinel, and Y. Zhang, "Delay constrained offloading for mobile edge computing in cloud-enabled vehicular networks," in *Proc. 8th Int. Workshop Resilient Netw. Des. Model.*, 2016, pp. 288–294.

[16] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.

[17] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management," *IEEE Netw.*, vol. 27, no. 5, pp. 48–55, Sep.-Oct. 2013.

[18] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, "Joint radio and computational resource allocation in IoT fog computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7475–7484, Aug. 2018.

[19] M. Li, P. Si, and Y. Zhang, "Delay-tolerant data traffic to software-defined vehicular networks with mobile edge computing in smart city," *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9073–9086, Oct. 2018.

[20] J. Du, R. Yu, X. Chu, J. Feng, and G. Lu, "Computation offloading and resource allocation in vehicular networks based on dual-side cost minimization," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1079–1092, Feb. 2019.

[21] S. S. Shah, M. Ali, A. W. Malik, M. A. Khan, and S. D. Ravana, "vFog: A vehicle-assisted computing framework for delay-sensitive applications in smart cities," *IEEE Access*, vol. 7, pp. 34900–34909, Mar. 2019.

[22] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Trans. Cloud Comput.*, vol. 7, no. 1, pp. 196–209, Jan.-Mar. 2019.

[23] Z. Zhou, J. Feng, Z. Chang, and X. S. Shen, "Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, May 2019.

[24] 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.

[25] S. Wang, Z. Zhang, R. Yu, and Y. Zhang, "Low-latency caching with auction game in vehicular edge computing," in *Proc. IEEE/CIC Int. Conf. Commun. China*, Oct. 2017, pp. 1–6.

[26] K. Zheng, H. Meng, P. Chatzimisios, L. Lei, and X. Shen, "An SMDP-based resource allocation in vehicular cloud computing systems," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7920–7928, Dec. 2015.

[27] H. Zhou, N. Cheng, Q. Yu, X. S. Shen, D. Shan, and F. Bai, "Toward multi-radio vehicular data piping for dynamic DSRC/TVWS spectrum sharing," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 10, pp. 2575–2588, Oct. 2016.

[28] H. Gao, C. Yuen, Y. Ren, T. Lv, and W. Long, "Distributed user scheduling for MIMO-Y channel," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 7123–7139, Dec. 2015.

[29] M. Xing, J. He, and L. Cai, "Maximum-utility scheduling for multimedia transmission in drive-thru Internet," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2649–2658, Apr. 2016.

[30] S. Li *et al.*, "Energy efficiency and capacity tradeoff in cloud radio access network of high-speed railways," *Mobile Inf. Syst.*, vol. 2017, pp. 1–12, 2017.

[31] H. Gao, M. Wang, and T. Lv, "Energy efficiency and spectrum efficiency tradeoff in the D2D-enabled HetNet," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10 583–10 587, Nov. 2017.

[32] H. Zhou *et al.*, "WhiteFi infostation: Engineering vehicular media streaming with geolocation database," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 8, pp. 2260–2274, Aug. 2016.

[33] H. Zhou *et al.*, "TV white space enabled connected vehicle networks: Challenges and solutions," *IEEE Netw.*, vol. 31, no. 3, pp. 6–13, May/Jun. 2017.

[34] H. D. Sherali and W. P. Adams, *Reformulation-Linearization Technique for Solving Discrete*. NJ, USA: John Wiley & Sons, Inc., 2011.

[35] Y. Niu, C. Gao, Y. Li, D. Jin, L. Su, and D. Wu, "Boosting spatial reuse via multiple-path multihop scheduling for directional mmwave WPANs," *IEEE Trans. Veh. Technol.*, vol. 65, no. 8, pp. 6614–6627, Aug. 2016.

[36] J. Clausen, "Branch and bound algorithms—Principles and examples," *Parallel Process. Lett.*, vol. 22, no. 5, pp. 658–663, 1999.

[37] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960.

[38] J. M. Stephen Boyd, *Branch and Bound Methods*. Berlin Heidelberg: Springer, 1987.

[39] J. He, Y. Ni, L. Cai, J. Pan, and C. Chen, "Optimal dropbox deployment algorithm for data dissemination in vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 3, pp. 632–645, Mar. 2018.

[40] X. Ma, H. Dong, X. Liu, L. Jia, G. Xie, and Z. Bian, "An optimal communications protocol for maximizing lifetime of railway infrastructure wireless monitoring network," *IEEE Trans. Ind. Informat.*, vol. 14, no. 8, pp. 3347–3357, Aug. 2018.

[41] C. Zhang, P. Fan, K. Xiong, and P. Fan, "Optimal power allocation with delay constraint for signal transmission from moving train to base stations in high-speed railway scenarios," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5775–5788, Dec. 2015.

[42] S. Lin *et al.*, "Finite-state Markov modeling for high-speed railway fading channels," *IEEE Antennas Wireless Propag. Lett.*, vol. 14, pp. 954–957, Jan. 2015.

[43] S. Lin, H. Zhang, J. Ding, H. Wang, B. Ai, and Z. Zhong, "Fast simulation of vehicular channels using finite-state markov models," *IEEE Wireless Commun. Lett.*, vol. 8, no. 4, pp. 1056–1059, Aug. 2019.

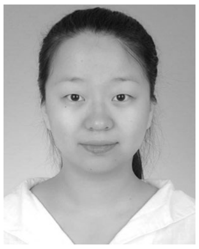
[44] X. Chen, W. Li, S. Lu, Z. Zhou, and X. Fu, "Efficient resource allocation for on-demand mobile-edge cloud computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8769–8780, Sep. 2018.

[45] J. Zhang *et al.*, "Energy-latency trade-off for energy-aware offloading in mobile edge computing networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2633–2645, Aug. 2018.

Shichao Li received the Ph.D. degree in communication and information systems from Beijing Jiaotong University, Beijing, China, in 2019. He is currently an Assistant Professor with the School of Information and Communication, Guilin University of Electronic Technology, Guilin, China. His main research interests include mobile edge computing, vehicular networks, high-mobility broadband wireless communications, wireless resource allocation, and cloud radio access networks.

Siyu Lin (Member, IEEE) received the B.E. and Ph.D. degrees in electronics engineering from Beijing Jiaotong University, Beijing, China, in 2007 and 2013, respectively. From 2009 to 2010, he was an Exchange Student with the Universidad Politenica de Madrid, Madrid, Spain. From 2011 to 2012, he was a Visiting Student with the University of Victoria, Victoria, BC, Canada. Since 2016, he has been with Beijing Jiaotong University, where he is currently an Associate Professor. His main research interests include wireless communication networks and railway mobile communications. He has served as a Track Co-Chair for IEEE VTC2020-Fall. He was the recipient of the First Class Award of Science and Technology in Railway in 2017.

Lin Cai (Fellow, IEEE) received the M.A.Sc. and Ph.D. degrees in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2002 and 2005, respectively. Since 2005, she has been with the Department of Electrical and Computer Engineering, University of Victoria, and she is currently a Professor. Her research interests span several areas in communications and networking, with a focus on network protocol and architecture design supporting emerging multimedia traffic and the Internet of Things. She is an NSERC E.W.R. Steacie Memorial Fellow. She was the recipient of the NSERC Discovery Accelerator Supplement (DAS) Grants in 2010 and 2015, respectively, and the Best Paper Awards of IEEE ICC 2008 and IEEE WCNC 2011. She has cofounded and chaired the IEEE Victoria Section Vehicular Technology and Communications Joint Societies Chapter. She has been elected to serve the IEEE Vehicular Technology Society Board of Governors, 2019–2021. She has served as an Area Editor for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, a member of the Steering Committee of the IEEE TRANSACTIONS ON BIG DATA and IEEE TRANSACTIONS ON CLOUD COMPUTING, an Associate Editor for the IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON COMMUNICATIONS, *EURASIP Journal on Wireless Communications and Networking*, *International Journal of Sensor Networks*, and *Journal of Communications and Networks*, and as the Distinguished Lecturer of the IEEE VTS Society. She has served as a TPC Co-Chair for IEEE VTC2020-Fall, and a TPC symposium Co-Chair for IEEE Globecom'10 and Globecom'13. She was awarded Outstanding Achievement in Graduate Studies. She is a registered Professional Engineer in British Columbia, Canada.



Wenjie Li received the B.C. degree from North China Electric Power University, Baoding, China, in 2018, and she is currently working toward the M.E. degree-with Beijing Jiaotong University, Beijing, China. Her current research interests focus on edge computing of vehicle networking.

Gang Zhu was born in 1958. He received the M.S. and Ph.D. degrees from Xian Jiaotong University, Xian, China, in 1993 and 1996, respectively. Since 1996, he has been with Beijing Jiaotong University, Beijing, China, where he is currently a Professor. From 2000 to 2001, he was a Visiting Scholar with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His current research interests include resource management in wireless communications, short distance wireless communications, and global system for mobile communications for railways. He was the recipient of Top Ten Sciences and Technology Progress of Universities in China in 2007 and First Class Award of Science and Technology in Railway in 2009.