Research Article

Efficient Task Offloading for 802.11p-Based Cloud-Aware Mobile Fog Computing System in Vehicular Networks

Qiong Wu^{1,2,3}, **Hongmei Ge**³, **Qiang Fan**⁴, **Wei Yin**¹, **Bo Chang**² and **Guilu Wu**³

¹Jiangsu Key Laboratory of Traffic and Transportation Security, Huaiyin Institute of Technology, Huaian 223003, China

²Jiangsu Laboratory of Lake Environment Remote Sensing Technologies, Huaiyin Institute of Technology, Huaian 223003, China

³School of Internet of Things Engineering, Jiangnan University, Wuxi 214122, China

⁴Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA

Correspondence should be addressed to Qiong Wu; qiongwu@jiangnan.edu.cn

Received 26 May 2020; Revised 11 July 2020; Accepted 25 August 2020; Published 9 September 2020

Academic Editor: Fuhong Lin

Copyright © 2020 Qiong Wu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Various emerging vehicular applications such as autonomous driving and safety early warning are used to improve the traffic safety and ensure passenger comfort. The completion of these applications necessitates significant computational resources to perform enormous latency-sensitive/nonlatency-sensitive and computation-intensive tasks. It is hard for vehicles to satisfy the computation requirements of these applications due to the limit computational capability of the on-board computer. To solve the problem, many works have proposed some efficient task offloading schemes in computing paradigms such as mobile fog computing (MFC) for the vehicular network. In the MFC, vehicles adopt the IEEE 802.11p protocol to transmit tasks. According to the IEEE 802.11p, tasks can be divided into high priority and low priority according to the delay requirements. However, no existing task offloading work takes into account the different priorities of tasks transmitted by different access categories (ACs) of IEEE 802.11p. In this paper, we propose an efficient task offloading strategy to maximize the long-term expected system reward in terms of reducing the executing time of tasks. Specifically, we jointly consider the impact of priorities of tasks transmitted by different ACs, mobility of vehicles, and the arrival/departure of computing tasks, and then transform the offloading problem into a semi-Markov decision process (SMDP) model. Afterwards, we adopt the relative value iterative algorithm to solve the SMDP model to find the optimal task offloading strategy. Finally, we evaluate the performance of the proposed scheme by extensive experiments. Numerical results indicate that the proposed offloading strategy performs well compared to the greedy algorithm.

1. Introduction

Intelligent connected vehicles can improve the traffic safety and ensure passenger comfort by supporting various applications such as autonomous driving, safety early warning, natural language processing, advertisements, and entertainments in the vehicular environment [1–4]. These applications consist of enormous latency-sensitive/nonlatency-sensitive and computation intensive tasks [5]. However, the computational capability of the vehicles is limited, which makes it difficult to support latency-sensitive application. Vehicular fog computing (VFC) has been emerged as an efficient approach to tackle this issue in the vehicular network, where the computational resources are pushed at

the network edge to satisfy the requirements of latency-sensitive tasks [6, 7]. Nevertheless, the computational resources of the VFC system are not sufficient because the number of tasks generated by the vehicular applications is so huge. Therefore, it is critical to propose a new computing paradigm for the vehicular network.

Mobile fog computing (MFC) system has been proposed as an efficient computing paradigm for the vehicular network, which extends the computational capability of the VFC system by jointly working with the remote cloud. In the MFC system, we consider each vehicle as a computational resource unit (RU) with the same computational capability [8], and the vehicles adopt the IEEE 802.11p protocol to communicate with each other [9]. The 802.11p employs the

enhanced distributed channel access (EDCA) mechanism to provide different quality-of-service (QoS) supports. Specifically, the EDCA mechanism defines different access categories (ACs) with different priorities to transmit different data traffic [10]. Since the vehicles in the MFC system generate enormous latency-sensitive/nonlatency-sensitive and computation intensive tasks due to various vehicular applications and these kinds of tasks are with different delay requirements, we consider that the latency-sensitive and computation-intensive tasks are of high priority and transmitted by higher priority AC to obtain higher-level QoS, while the nonlatency-sensitive tasks are low priority and granted with lower priority AC. When a vehicle generates a high priority task, i.e., service requester, it can offload the task to other vehicles or remote cloud, i.e., accepting by the vehicular fog or transmit it to the remote cloud. Since the QoS of the low priority task is not stringent, service requester can only offload low priority tasks to other vehicles in the vehicular fog, or the tasks are rejected by the system. Once the tasks are accepted by the vehicular fog, the system needs to determine how many RUs to be assigned to obtain maximal long-term expected reward. Note that the main goal for the MFC system is to reduce the executing time of tasks [11]. The MFC system in this paper includes the features which are proposed in previous literatures: (1) vehicles arrive/depart the system; and (2) computing tasks arrive/depart the system, resulting in that the number of available resources in the system is variable. In addition, the MFC system has its unique feature, i.e., considering different priorities of tasks transmitted by different ACs of the 802.11p EDCA mechanism, which makes it challenge to find an optimal task offloading strategy to maximize the long-term expected system reward.

To the best of our knowledge, although there are extensive studies on the task offloading scheme in the MFC system for vehicular network, no existing literature considers the feature, i.e., tasks with different delay requirements are transmitted by different ACs of the 802.11p EDCA mechanism, which poses a significant challenge to construct model to find the optimal task offloading policy. Thus it is necessary to propose an optimal offloading policy while considering the high/low priority tasks transmitted by different ACs, which motivates us to do this work.

In this paper, we consider that the different priorities of tasks transmitted by different ACs and propose an optimal task offloading policy for the MFC system. The main contributions of this paper are summarized as follows.

- (1) We propose an offloading strategy to obtain maximal long-term expected reward in the MFC system for vehicular network while jointly considering the impact of computation requirements of tasks, vehicle mobility, and the arrival/departure of high/low priority task. Specifically, we transform the task offloading process into a semi-Markov decision process (SMDP) model where all the system state set, action set, state transition probabilities, and system reward function are defined. To solve the problem efficiently, we adopt the relative value iterative algorithm to find the optimal offloading strategy for the MFC system
- (2) To demonstrate the performance of the proposed optimal strategy, we perform extensive experiments for our proposed strategy and the greedy algorithm (GA) under the same condition and obtain numerical results. The results indicate that the performance of our proposed strategy has been significantly improved compared to the GA method

The rest of this paper is organized as follows. Section 2 discusses the related work on the task offloading strategy in the vehicular network. The MFC system is described in Section 3. We construct the SMDP model to formulate the task offloading problem in Section 4. The relative value iteration algorithm is introduced in Section 5. Section 6 provides the numerical results and the corresponding performance analysis. The conclusion of this paper is given in Section 7.

2. Related Work

The computing paradigm VFC and MFC are widely used to the vehicular network. In this section, we first review the related works on task offloading in the VFC system, and then the works about offloading in the MFC system are discussed.

2.1. Task Offloading in the VFC System. Zhu et al. [6] considered the fog node capacity, constraints on service latency, and quality loss to propose a solution for task allocation in the VFC system. They first changed the process of task allocation into a joint optimization problem which is a biobjective optimization problem. To solve the problem, they also proposed an event-triggered dynamic task allocation framework based on the linear programming-based optimization and binary particle swarm optimization. Wu et al. [11] considered the transmission delay caused by 802.11p and proposed a task offloading strategy for the VFC system. They first transformed the task offloading problem into an SMDP model, and then they adopted an iterative algorithm to solve the SMDP to attain optimal strategy. Zhou et al. [12] first proposed an efficient incentive mechanism based on contract theoretical modelling which is tailored for the unique characteristic of each vehicle type to motivate vehicles to share their resources. Then, they formulated the task assignment problem as a two-sided matching problem which is solved by a pricing-based stable matching algorithm to minimize the network delay. Zhao et al. [13] proposed a contract-based incentive mechanism, which combines resource contribution and utilization. Then, they adopted distributed deep reinforcement learning (DRL) to reduce implementation complexity in the VFC system. Finally, they proposed a task offloading scheme based on the queuing model to avoid the task offloading conflict. Lin et al. [14] proposed a resource allocation management scheme to reduce servicing time. They first introduced a serving model, and then they built a VFC system utility model based on the serving model which is solved by a two-step method. Specifically, they first presented all the suboptimal solution based on a Lagrangian algorithm, and then they provided an optimal solution selection process. Xie et al. [15] jointly considered the effect of vehicle mobility and time-varying computation capability

FIGURE 1: The model for cloud-aware MFC system. The MFC system consisted of the vehicular fog and remote cloud.

and proposed an effective resource-aware based parallel offloading policy for the VFC system.

2.2. Task Offloading in the MFC System. Ning et al. [4] developed an energy-efficient task offloading scheme. Specifically, they first formulated the optimization problem to minimize energy consumption through jointly considering load balance and delay constraint. Then, they divided the optimization problem into two stages, i.e., flow redirection and offloading decision. Finally, they adopted Edmonds-Karp and deep reinforcement learning-based Minimizing Energy Consumption algorithms to solve the optimization problem. Zheng et al. [8] considered the variability feature of resources and proposed an optimal computation resource allocation strategy to maximize the long-term expected reward of the MFC system in terms of power and processing time. Specifically, they first transformed the optimization problem into an SMDP, and then they adopted the iteration algorithm to find the optimal scheme. Lin et al. [16] took into account the heterogeneous vehicles and roadside units, then the resource allocation problem was formulated as an SMDP model. Afterwards, the formulated problem is solved by a proposed method. Zhao et al. [17] first changed a collaborative computing offloading problem into a constrained optimization by jointly optimizing computation offloading decision and computation resource allocation. They also adopted a collaborative computation offloading and resource allocation optimization scheme to solve the optimization problem. Wu et al. [18] considered the vehicles which are processing tasks and proposed a task offloading scheme to maximize the long-term system reward leaving system. Specifically, they first formulated the offloading problem as an infinite SMDP. Then, they employed the value iteration algorithm to tackle this problem. Wang et al. [19] jointly considered the heterogeneous delay requirements for vehicular applications and the variable computation resources to propose an efficient offloading policy. They first introduced a priority queuing system to model the MFC system, and then found an

application-aware offloading policy by an SMDP. Liu et al. [20] developed an offloading strategy in the MFC system to minimize the task offloading delay which is consisted of transmission delay, computational delay, waiting delay, and handover delay. Specifically, they first established the task offloading delay model, and then they developed a pricing-based one-to-one matching algorithm and pricing-based one-to-many matching algorithm to obtain offloading strategy.

From the mentioned above, we can find there is no existing work considering the different priorities of tasks transmitted by different ACs of the 802.11p, which is the motivation of our work.

3. System Model

In this section, the MFC system is first described in detail, and then we introduce how vehicles employ the IEEE 802.11p EDCA mechanism to transmit high/low priority computing tasks.

3.1. System Description. The MFC system has jointly considered the impact of priority of tasks, vehicle mobility, and arrival/departure of the computing tasks. The scenario is shown as Figure 1; it consists of both the vehicular fog and remote cloud. Once a high priority task is generated, the system needs to make a decision to assign it to the desirable computing entity, i.e., offloading the task to the vehicular fog or transmitting it to the remote cloud. Since the tasks with low priority are not latency-sensitive, the system is more likely to execute them in the vehicular fog or drop them. In addition, if a task is received by the vehicular fog, the system further determines how many available vehicles, i.e., available RUs, are assigned to execute it. A simple example is shown in Figure 1. The task from vehicle C_1 arrives at the system, and two RUs are assigned to handle it. Afterwards, the service requester receives the computing results from RUs. We assume that the maximal number of computational resources

is M in the MFC system, and the computational service rate of each vehicle for high/low priority tasks is μ_i . The vehicles move into or depart from the MFC system according to the Poisson process with parameter λ_v and μ_v , respectively. Similarly, the arrival rate of high and low priority computing tasks follows Poisson distribution with parameter λ_1 and λ_2 .

3.2. Data Transmission of 802.11p EDCA Mechanism. The 802.11p EDCA defines four ACs with different priorities to transmit different types of tasks. Each AC queue employs its own parameters, i.e., arbitration interframe space number (AIFS), minimum contention windows, and maximum contention windows [21]. Denote $CW_{i,\min}$ and $CW_{i,\max}$ as the minimum contention windows and maximum contention windows of AC_i ($i = 1, 2$), respectively. Thus, the maximal times that the contention window of AC_i can be doubled, i.e., R_i , is expressed by Equation (1). In this paper, the high priority tasks are transmitted by AC_1 queue and low priority tasks are transferred by AC_2 queue in the broadcast mode. Similar to most related work [22–25], we assume that the channel is ideal. The transmitting procedure of tasks in the MFC system is described as follows. Specifically, when an AC queue in a vehicle has a task to transmit and the channel is idle for the time duration of arbitration interframe space (AIFS), a backoff process is initiated. Specifically, a random value is selected between zero and the minimum contention windows as the value of the backoff counter. Then, if the channel is kept idle for one slot, the backoff timer is decreased by one. Otherwise, the backoff timer is frozen until the channel keeps idle for the duration of AIFS. If the value of the backoff timer is decreased to zero, the packet is transmitted. Once the two ACs in a vehicle are transmitting simultaneously, the internal collision happens. In this case, the high priority task will be transmitted, while the low priority task is retransmitted with a new backoff procedure with doubled contention windows. If the number of retransmissions is more than the retransmission limit L_i , the task will be dropped.

$$R_i = \log_2 \frac{CW_{i,\max} + 1}{CW_{i,\min} + 1}. \quad (1)$$

4. SMDP Model

In this section, the task offloading process is transformed into an SMDP model. To clarify the problem, we have defined the system state set, action set, state transition probabilities, and system reward function of the system.

4.1. System State Set. The system state x includes the number of vehicles, i.e., K , the number of high/low priority computing tasks which are executed by a different number of RUs and the event. The event is denoted as e , where $e \in \kappa = \{A_1, A_2, D_{1,1}, \dots, D_{1,N}, \dots, D_{2,N}, F_{+1}, F_{-1}\}$. Here, A_1 means that the service requester generates a high priority computing task, i.e., arrival of high priority computing task; A_2 means that the service requester generates a low-priority computing task, i.e., arrival of low priority computing task; $D_{i,j}$ means that the task with priority i processed by j RUs departs from the MFC, i.e., completion of a task with priority i processed by j RUs;

F_{+1} means a vehicle moving into the MFC system, i.e., arrival of a vehicle and F_{-1} means an available vehicle leaving the system, i.e., departure of a vehicle. Thus, the system state set can be expressed as

$$X = \{x \mid x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, e)\}, \quad (2)$$

where $s_{i,j}$ ($i = 1, 2; j = 1, 2, \dots, N$) means the number of the tasks with priority i processed by j RUs and N means that the maximal number of RUs for processing a task. It is no doubt that the number of busy vehicles must be smaller than K , i.e., $\sum_{i=1}^2 \sum_{j=1}^N j s_{i,j} \leq K$.

4.2. Action Set. The action $a(x)$ is related with the system current state x and reflects the decision of the MFC system under the current event. The action belongs to the set $\zeta = \{-1, 0, 1, \dots, N\}$, where $a(x) = -1$ indicates that the system takes no action when the event is one of the completion of a task, vehicle moving into or departing from the system; $a(x) = 0$ indicates that the system transmits the high priority computing task to the remote cloud or the system rejects the low priority task due to lack of computational resources; $a(x) = j$ indicates that j RUs are assigned to execute a task. Thus, the action set can be expressed as

$$A_c = \begin{cases} \{-1\}, & e \in \{D_{1,1}, \dots, D_{i,j}, \dots, D_{2,N}, F_{+1}, F_{-1}\}, \\ \{0, 1, 2, \dots, N\}, & e \in \{A_i\}, i = 1, 2. \end{cases} \quad (3)$$

4.3. State Transition Probabilities. In the SMDP model, the next state is related to the current state and action, and the state transition probabilities represent the relationship between the current state and the next state. Thus, the state transition probabilities are defined as the ratio between the arrival rate of the next event and the sum of the arrival rate of all events. Given the current state $x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, e)$ and action a , denote $P(k \mid x, a)$ as the transition probability from the current state x to the next state k after taking action $a(x)$, and $\beta(x, a)$ as the sum of arrival rate for all events. To model state transition probabilities, the arrival rate of the next event should be discussed first, which is different according to the current event and action. The detailed procedure is as follows.

As a service requester generates a high priority computing task, the MFC system transmits it to the remote cloud. In this case, the arrival rate of event A_i ($i = 1, 2$) is $K\lambda_i$. The number of busy RUs does not change when the task is executed in the remote cloud. The arrival rate of event $D_{i,j}$, i.e., completion of a task with priority i and processed by j RUs, is $s_{i,j} \cdot j\mu_i$. The arrival rate of event F_{+1} and F_{-1} are λ_v and μ_v , respectively. If a high/low priority task is generated and received by the vehicular fog, j RUs are assigned to execute it. In this case, when the next event is one of the arrival of computing task, vehicle moving into or departing from the system, the arrival rate of the next event is $K\lambda_i$, λ_v , and μ_v , respectively. Let E be the next event. When the event, i.e., the completion of a task with priority i processed by j RUs,

occurs, the arrival rate of the next event is analyzed as follows:

$$(1) e = A_i, a(x) = j, E = D_{i,j}$$

This case indicates that the service requester generates a computing task with priority i , and it is executed by j RUs. Thus, the number of tasks with priority i handled by j RUs increases, and the arrival rate of event $D_{i,j}$ is $(s_{i,j} + 1) \cdot j \mu_t$.

$$(2) e = A_i, a(x) = j, E = D_{i,m}, m \neq j$$

This case represents that a task with priority i processed by m RUs is accomplished. Thus, the arrival rate of event $D_{i,m}$ is $s_{i,m} \cdot m \mu_t$.

$$(3) e = A_i, a(x) = j, E = D_{n,w}, n \neq i$$

This case means that the task with priority n processed by w RUs is accomplished. The arrival rate of event $D_{n,w}$ is $s_{n,w} \cdot w \mu_t$.

In conclusion, given the current state $x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,1}, \dots, s_{2,N}, A_i)$ and action a , the transition probabilities can be expressed as Equations (4) and (5), which are shown on the next page.

Similarly, as the current event is one of the completion of a task with priority i processed by j RUs, vehicle moving into or departing from the system. The state transition probabilities can be calculated by Equations (6)–(8), respectively.

$$(1) x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, A_1)$$

$$P(k | x, a) = \begin{cases} \frac{K \lambda_i}{\beta(x, a)}, & a = 0, k = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, A_i), i = 1, 2, \\ \frac{s_{i,j} \cdot j \mu_t}{\beta(x, a)}, & a = 0, k = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, D_{i,j}), \\ i = 1, 2; j = 1, \dots, N, \\ \frac{\lambda_v}{\beta(x, a)}, & a = 0, k = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & a = 0, k = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{-1}), \\ \frac{K \lambda_i}{\beta(x, a)}, & a = j, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, A_i), i = 1, 2, \\ \frac{(s_{1,j} + 1) \cdot j \mu_t}{\beta(x, a)}, & a = j, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, D_{1,j}), \\ j = 1, 2, \dots, N, \\ \frac{s_{1,m} \cdot m \mu_t}{\beta(x, a)}, & a = j, m \neq j, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, D_{1,m}), \\ \frac{s_{n,w} \cdot w \mu_t}{\beta(x, a)}, & a = j, n \neq 1, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, D_{n,w}), \\ \frac{\lambda_v}{\beta(x, a)}, & a = j, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & a = j, k = (K, s_{1,1}, \dots, s_{1,j} + 1, \dots, s_{2,N}, F_{-1}). \end{cases} \quad (4)$$

$$(2) x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, A_2), a = j$$

$$P(k | x, a) = \begin{cases} \frac{K \lambda_i}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, A_i), i = 1, 2, \\ \frac{(s_{2,j} + 1) \cdot j \mu_t}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, D_{2,j}), \\ j = 1, 2, \dots, N, \\ \frac{s_{2,m} \cdot m \mu_t}{\beta(x, a)}, & m \neq j, k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, D_{2,m}), \\ \frac{s_{n,w} \cdot w \mu_t}{\beta(x, a)}, & n \neq 2, k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, D_{n,w}), \\ \frac{\lambda_v}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{2,j} + 1, \dots, s_{2,N}, F_{-1}). \end{cases} \quad (5)$$

$$(3) x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, D_{i,j}), i = 1, 2; j = 1, 2, \dots, N; a = -1$$

$$P(k | x, a) = \begin{cases} \frac{K \lambda_i}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, A_i), \\ \frac{(s_{i,j} - 1) \cdot j \mu_t}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, D_{i,j}), \\ \frac{s_{i,m} \cdot m \mu_t}{\beta(x, a)}, & m \neq j, \\ k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, D_{i,m}), \\ \frac{s_{n,w} \cdot w \mu_t}{\beta(x, a)}, & n \neq i, \\ k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, D_{n,w}), \\ \frac{\lambda_v}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & k = (K, s_{1,1}, \dots, s_{i,j} - 1, \dots, s_{2,N}, F_{-1}). \end{cases} \quad (6)$$

$$(4) x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{+1}), a = -1$$

$$P(k | x, a) = \begin{cases} \frac{(K+1) \lambda_i}{\beta(x, a)}, & k = (K+1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, A_i), \\ \frac{s_{i,j} \cdot j \mu_t}{\beta(x, a)}, & k = (K+1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, D_{i,j}), \\ \frac{\lambda_v}{\beta(x, a)}, & k = (K+1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & k = (K+1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{-1}). \end{cases} \quad (7)$$

$$(5) x = (K, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{-1}), a = -1$$

$$P(k | x, a) = \begin{cases} \frac{(K-1)\lambda_i}{\beta(x, a)}, & k = (K-1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, A_i), \\ \frac{s_{i,j} \bullet j\mu_t}{\beta(x, a)}, & k = (K-1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, D_{i,j}), \\ \frac{\lambda_v}{\beta(x, a)}, & k = (K-1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{+1}), \\ \frac{\mu_v}{\beta(x, a)}, & k = (K-1, s_{1,1}, \dots, s_{1,N}, \dots, s_{2,N}, F_{-1}). \end{cases} \quad (8)$$

Since $\beta(x, a)$ is the sum of arrival rate for all events, the expression can be denoted by Equation (9), i.e.,

$$\beta(x, a) = \begin{cases} K \sum_{i=1}^2 \lambda_i + \lambda_v + \mu_v + \sum_{i=1}^2 \sum_{j=1}^N s_{i,j} \bullet j\mu_t + j\mu_t, & e = A_i, a = j, \\ K \sum_{i=1}^2 \lambda_i + \lambda_v + \mu_v + \sum_{i=1}^2 \sum_{j=1}^N s_{i,j} \bullet j\mu_t - j\mu_t, & e = D_{i,j}, a = -1, \\ (K+1) \sum_{i=1}^2 \lambda_i + \lambda_v + \mu_v + \sum_{i=1}^2 \sum_{j=1}^N s_{i,j} \bullet j\mu_t, & e = F_{+1}, a = -1, \\ (K-1) \sum_{i=1}^2 \lambda_i + \lambda_v + \mu_v + \sum_{i=1}^2 \sum_{j=1}^N s_{i,j} \bullet j\mu_t, & e = F_{-1}, a = -1. \end{cases} \quad (9)$$

4.4. System Reward Function. Given the action, the system is rewarded when the MFC system changes from the current state to the next state. Denote $r(x, a)$ as the system reward function, which is expressed by

$$r(x, a) = h(x, a) - g(x, a), \quad (10)$$

where $h(x, a)$ means the revenue of the system by taking action a under state x and $g(x, a)$ represents the system cost during the period between two states. Next, we first discuss the revenue, and then the system cost is explained.

4.4.1. System Revenue. The main goal for the MFC system is to reduce the executing time of tasks. Let D_1 be the transmission delay from the vehicular fog to the remote cloud, T be the processing time of a task when the requester executes the task locally. $Ts_i (i = 1, 2)$ be the transmission time from the requester to the vehicular fog and $D_t(j)$ be the executing time of a task by j RUs. Similar to previous literatures [26], we do not take into account the feedback time of the analytical results. Note that the remote cloud is equipped with power capability and thus the processing time of remote cloud is ignored. If a high priority computing task arrives at the system and the computation resources are not sufficient, the system transmits the task to the vehicular fog and then transfers it to a remote cloud. In this case, the revenue of the system is denoted as $\eta[T - Ts_1 - D_1]$, where η is the price of per unit time. When a low priority task reaches the system where available RUs are insufficient, the task will be rejected and the system is terminated with punishment parameter ϕ .

If a high/low priority task is executed by the vehicular fog, the revenue is expressed as $\eta[T - Ts_i - D_t(j)]$. If the event is one of the completions of a task, vehicle moving into or leaving the system, the system takes no action and the revenue is zero. A busy vehicle moves out of the MFC system, resulting in the failure of executing tasks, and the system is punished with parameter ξ . Thus, the revenue of the system under different events and actions can be formulated by Equation (11), i.e.,

$$h(x, a) = \begin{cases} \eta[T - Ts_1 - D_1], & e = A_1, a = 0, \\ -\phi, & e = A_2, a = 0, \\ \eta[T - Ts_i - D_t(j)], & e = A_i, a = j, \\ 0, & e \in \{D_{1,1}, \dots, D_{i,j}, \dots, D_{2,N}, F_{+1}\}, a = -1, \\ 0, & e = F_{-1}, a = -1, \sum_{i=1}^2 \sum_{j=1}^N js_{i,j} < K, \\ -\xi, & e = F_{-1}, a = -1, \sum_{i=1}^2 \sum_{j=1}^N js_{i,j} = K, \end{cases} \quad (11)$$

Since the computational service rate of each vehicle for high/low priority tasks is μ_j , the executing time of a task by j RUs can be expressed as

$$D_t(j) = \frac{1}{j\mu_t}, j = 1, 2, \dots, N. \quad (12)$$

The tasks with different priorities are transmitted by different ACs, i.e., AC_1 and AC_2 , with different transmission delay. Similar to the work [27], we consider the whole tasks transmitting process as a z -domain linear model and adopt the probability generating function (PGF) method to obtain the transmission delay for ACs. $P_{td}^i(z)$ is denoted as the PGF of the transmission delay, which can be expressed by Equation (13), i.e.,

$$\begin{cases} P_{td}^1(z) = TR(z)G_{1,0}(z), \\ P_{td}^2(z) = (1 - \omega_1)TR(z) \sum_{n=0}^{L_2} \left[\omega_1^n \prod_{m=0}^n G_{2,m}(z) \right] + \omega_1^{L_2+1} \prod_{m=0}^{L_2} G_{2,m}(z), \end{cases} \quad (13)$$

where $TR(z)$ is the PGF of average transmission time, $G_{i,m}(z)$ is the PGF of backoff time of AC_i when the number of retransmission is m , and ω_i is the transmission probability of AC_i .

$W_{i,m}$ is denoted as the maximal contention window (CW) size of $AC_i (i = 1, 2)$ when the number of retransmission is m , which can be calculated by Equation (14). Let $H_i(z)$ be the PGF of average time that the backoff timer reduces by one. Thus, the PGF of $G_{i,m}(z)$ can be calculated as Equation (15).

$$W_{i,m} = \begin{cases} 2^m (CW_{i,\min} + 1), & m \leq R_i, \\ 2^{R_i} (CW_{i,\min} + 1), & R_i < m \leq L_i, \end{cases} \quad (14)$$

$$\begin{cases} G_{1,0}(z) = \frac{1}{W_{1,0}} \sum_{m=0}^{W_{1,0}-1} [H_1(z)]^m, \\ G_{2,m}(z) = \begin{cases} \frac{1}{W_{2,m}} \sum_{m=0}^{W_{2,m}-1} [H_2(z)]^m, & m \in [0, R_2], \\ \frac{1}{W_{2,R_2}} \sum_{m=0}^{W_{2,R_2}-1} [H_2(z)]^m, & m \in (R_2, L_2]. \end{cases} \end{cases} \quad (15)$$

Let T_{slot} be the duration of a slot. The PGF of $H_i(z)$ is expressed by Equation (16), i.e.,

$$H_i(z) = (1 - p_{bi})z^{T_{\text{slot}}} + p_{bi}z^{T_{\text{tr}} + \text{AIFS}_i}, \quad i = 1, 2, \quad (16)$$

where AIFS_i is the time duration of AIFS and p_{bi} is the backoff freezing probability that the requester vehicle senses other vehicles in the MFC system occupying the channel or other access categories attempting to transmit tasks. Since AC_1 needs to sense channel for A more slots than AC_0 , the p_{bi} can be calculated by

$$\begin{cases} p_{b1} = 1 - (1 - \omega_1)^{K-1} (1 - \omega_2)^K, \\ p_{b2} = 1 - [(1 - \omega_1)^K (1 - \omega_2)^{K-1}]^{A+1}, \end{cases} \quad (17)$$

where A is calculated according to AIFSN[2] and AIFSN[1] as follows.

$$A = \text{AIFSN}[2] - \text{AIFSN}[1]. \quad (18)$$

Let SIFS be the time duration of short inter-frame spacing (SIFS). The AIFS_i can be expressed by Equation (19) according to the 802.11p.

$$\text{AIFS}_i = \text{SIFS} + \text{AIFSN}[i] \times T_{\text{slot}}. \quad (19)$$

Assuming that all high/low priority tasks have the same size, i.e., $E(P)$, the average transmission time is given by

$$T_{\text{tr}} = \frac{\text{PHY}_h}{R_b} + \frac{\text{MAC}_h + E(P)}{R_d} + \sigma, \quad (20)$$

where PHY_h and MAC_h are the header length of the physical and MAC layer, and σ is the propagation time. R_b and R_d are the basic rate and data rate, respectively. Thus, the PGF of average transmission time can be expressed by Equation (21), i.e.,

$$\text{TR}(z) = z^{T_{\text{tr}}}. \quad (21)$$

According to the Markov chain in [27], the transmission probability of AC_i can be expressed by Equation (22), i.e.,

$$\begin{cases} \omega_1 = \left[\frac{W_{1,0} + 1}{2(1 - p_{b1})} + \frac{1 - \rho_1}{p_{a1}} \right]^{-1}, \\ \omega_2 = \frac{1 - \omega_1^{L_2+1}}{1 - \omega_1} \left[\frac{1 - \omega_1^{L_2+1}}{1 - \omega_1} + \frac{W_{2,0} - 1}{2(1 - p_{b2})} + \frac{W_{2,0}\omega_1[1 - (2\omega_1)^{R_2}]}{(1 - p_{b2})(1 - 2\omega_1)} \right. \\ \left. + \frac{2^{R_2-1}W_{2,0}\omega_1^{R_2+1}(1 - \omega_1^{L_2-R_2})}{(1 - p_{b2})(1 - \omega_1)} + \frac{1 - \rho_2}{p_{a2}} \right]^{-1}, \end{cases} \quad (22)$$

where ρ_i is the utilization of AC_i , and p_{ai} is the task arrival probability of AC. Since the arrival rate of all computing tasks follows Poisson distribution, the arrival probability p_{ai} can be calculated by Equation (23), i.e.,

$$p_{ai} = \sum_{m=1}^{\infty} \frac{(\lambda_i T_{\text{slot}})^m}{m!} = 1 - e^{-\lambda_i T_{\text{slot}}}, \quad i = 1, 2. \quad (23)$$

Initialize the utilization ρ_i , then ω_i and p_{bi} can be calculated according to Equations (17) and (22) through iterative method. Substituting ω_i and p_{bi} into Equation (13), the PGF of transmission time can be obtained. Thus, the transmission time can be obtained by

$$T_{s_i} = \left. \frac{dP_{td}^i}{dz} \right|_{z=1}, \quad i = 1, 2. \quad (24)$$

4.4.2. System Cost. Given the current state x and action a , the system cost means the cost caused by executing tasks during the period of two states and is expressed by Equation (25).

$$g(x, a) = c(x, a)\tau(x, a), \quad (25)$$

where $c(x, a)$ is the number of busy vehicles under the current event after taking action, which can be calculated by Equation (26) and $\tau(x, a)$ is the corresponding expected time.

$$c(x, a) = \sum_{i=1}^2 \sum_{j=1}^N j s_{i,j}. \quad (26)$$

Denote α as the continuous-time discounted factor. In this paper, we adopt the discounted reward model found in [28], i.e.,

$$g(x, a) = c(x, a)E_x^a \left\{ \int_0^{\tau} e^{-\alpha t} dt \right\} = \frac{c(x, a)}{\alpha + \beta(x, a)}. \quad (27)$$

Thus, the system revenue function can be rewritten as Equation (28).

$$r(x, a) = h(x, a) - \frac{c(x, a)}{\alpha + \beta(x, a)} \quad (28)$$

- 1: For each state $x \in X$, initialize the state-value $\tilde{v}_*(x) = 0$. Set a small constant $\epsilon > 0$, and initialize the number of iterative $m = 0$.
- 2: For each state x , calculate $\tilde{v}_*^{m+1}(x)$ by Bellman optimal equation:

$$\tilde{v}_*^{m+1}(x) = \max_{a \in A_c} [\tilde{r}(x, a) + \tilde{\gamma} \sum_{k \in X} \tilde{P}(k | x, a) \tilde{v}_*(k).]$$
- 3: If $\|\tilde{v}_*^{m+1}(x) - \tilde{v}_*^m(x)\| < \epsilon(1 - \tilde{\gamma})/2\tilde{\gamma}$ for all $x \in X$, go to step 4; otherwise the number of iterative increases, i.e., $m = m + 1$ and go to step 2.
- 4: For all state $x \in X$, calculate the optimal strategy $\pi_*(x)$ by

$$\pi_*(x) \in \underset{a \in A_c}{\operatorname{argmax}} [\tilde{r}(x, a) + \tilde{\gamma} \sum_{k \in X} \tilde{P}(k | x, a) \tilde{v}_*(k)]$$

ALGORITHM 1

5. Relative Value Iteration Algorithm

In this section, we adopt the relative value iteration algorithm to find the optimal task offloading strategy to maximize the long-term expected reward in terms of reducing the executing time of tasks. Specifically, the relative value iteration algorithm is used to solve the Bellman optimal equation [29]. The Bellman equation is expressed by Equation (29), where $\gamma \in [0, 1]$ is the discount factor that determines the impact of future reward on current state.

Since the continuous-time SMDP is hard to solve and the discrete MDP can be solved directly through iterating the Bellman optimal equation to obtain optimal strategy, we transform the continuous-time SMDP into the discrete MDP by uniformizing the system revenue, the discount factor, and state transition probabilities according to Equations (30)–(32).

$$v_*(x) = \max_{a \in A_c} \left[r(x, a) + \gamma \sum_{k \in X} P(k | x, a) v_*(k) \right], \quad (29)$$

$$\tilde{r}(x, a) = r(x, a) \frac{\alpha + \beta(x, a)}{\alpha + Q}, \quad (30)$$

$$\tilde{\gamma} = \frac{Q}{\alpha + Q}, \quad (31)$$

$$\tilde{P}(k | x, a) = \begin{cases} 1 - \frac{[1 - P(k | x, a)]\beta(x, a)}{Q}, & k = x, \\ \frac{P(k | x, a)\beta(x, a)}{Q}, & k \neq x, \end{cases} \quad (32)$$

where $Q = M \sum_{i=1}^2 \lambda_i + \lambda_v + \mu_v + M \cdot N \mu_t$. Substituting Equations (30)–(32) into Equation (29), the Bellman optimal equation can be written as

$$\tilde{v}_*(x) = \max_{a \in A_c} \left[\tilde{r}(x, a) + \tilde{\gamma} \sum_{k \in X} \tilde{P}(k | x, a) \tilde{v}_*(k) \right]. \quad (33)$$

The detailed description of the relative value iterative algorithm to solve Equation (33) is presented in Algorithm 1.

TABLE 1: Experiment parameters.

Parameter	Value	Parameter	Value
AIFSN[1]	2	AIFSN[2]	3
R_b	6 Mbps	R_d	3 Mbps
MAC_h	112 bits	PHY_h	48 bits
T_{slot}	13 μs	SIFS	32 μs
$\text{CW}_{1,\min}$	3	$E[P]$	1920 bytes
$\text{CW}_{2,\min}$	7	$\text{CW}_{2,\max}$	15
L_i	2	σ	2 μs
N	2	μ_t	20
λ_1	2	λ_2	3
λ_f	9	μ_f	2
η	5	T	0.095
D_1	0.085	ϕ	10
ξ	38	α	0.1

6. Numerical Results and Analysis

In this section, we conduct extensive experiments in MATLAB 2010a and obtain numerical results to demonstrate the performance of the proposed offloading strategy though comparing it with the greedy algorithm. The GA method indicates that the system is always inclined to allocate as many RUs as possible to execute tasks. The considered scenario is shown in Figure 1. In the experiment, we first initialize the system state set, action set, state transition probabilities, and system reward according to Equations (2)–(10). Then, the relative value iterative algorithm is used to solve the Bellman optimal equation to obtain an optimal offloading strategy. Finally, we compare the proposed strategy with the GA method in terms of the long-term expected reward. We assume that the maximal number of RUs for executing a task is 2. $b1$ and $b2$ mean that the system assigns one RU and two RUs to execute a high priority task. Similarly, $B1$ and $B2$ indicate that the system allocates one RU and two RUs to process a low priority task. $b0$ represents that a high priority task is transmitted to remote cloud. $B0$ means that the system rejects a low priority task. For simplicity, the number of retransmissions limit is set to be 2. The parameters of 802.11p are adopted according to the IEEE 802.11p standard [30], and the main parameters in the experiment are shown in Table 1.

FIGURE 2: The transmission delay of high/low priority computing tasks under the different maximal number of vehicles in the system. T_{s1} and T_{s2} represent the transmission delay of high/low priority tasks, respectively.

FIGURE 3: The action probabilities of the MFC system under the different maximal number of vehicles.

Figure 2 shows the transmission delay of high/low priority computing tasks when the maximal amounts of vehicles changes. We can see that, when the maximal amounts of

vehicles increases, the transmission delay of tasks keeps increasing. With the increase of the maximal number of vehicles in the system, the packet collision probability is

FIGURE 4: The long-term expected reward of the MFC system under different maximal number of vehicles. SMDP is the proposed strategy and GA is the greedy algorithm.

TABLE 2: Offloading strategy for the high priority tasks. ($K = 9, s_{1,1} = 0, s_{2,1} = 0, 1$).

$s_{1,1}, s_{1,2}$	$s_{2,1}, s_{2,2}$									
	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
0,0	2	2	2	2	0	2	2	2	1	0
0,1	2	2	2	0		2	2	1	0	
0,2	2	2	0			2	1	0		
0,3	2	0				1	0			
0,4	0					0				

TABLE 3: Offloading strategy for the low priority tasks. ($K = 9, s_{1,1} = 0, s_{2,1} = 0, 1$).

$s_{1,1}, s_{1,2}$	$s_{2,1}, s_{2,2}$									
	0,0	0,1	0,2	0,3	0,4	1,0	1,1	1,2	1,3	1,4
0,0	2	2	2	2	1	2	2	2	1	0
0,1	2	2	2	1		2	2	1	0	
0,2	2	2	1			2	1	0		
0,3	2	1				1	0			
0,4	1					0				

increased, thus incurring the degraded transmission delay. In addition, we can find that the transmission delay of high priority tasks, i.e., T_{s_1} , is higher than that of low priority tasks, i.e., T_{s_2} . This is because that the contention window of AC_1 queue which is used to transmit high priority tasks is smaller than that of AC_2 .

Figure 3 shows the action probabilities of the MFC system when the maximal number of vehicles in the MFC system changes. It can be seen that the probabilities of b_0 and B_0 become smaller with the increasing of the maximal number of vehicles, which can be explained as follows. When the maximal number of vehicles increases, the computational resources become sufficient, resulting in that the system tends to execute tasks in the vehicular fog. Since the system is inclined to process tasks in the vehicular fog, the probabilities of b_1 , B_1 , b_2 , and B_2 become larger. With the number of vehicles further increasing, the available resource is abundant and the system assigns as many RUs as possible to process

tasks to maximize the system reward in terms of reducing executing time of tasks. Therefore, the probabilities of b_1 and B_1 decrease, while those of b_2 and B_2 continue to increase. Since the difference of transmission delay for high/low priority is small and the computational rate for different priorities of tasks is the same, the difference of system revenue for action b_2 and B_2 is not obvious. Thus, the probabilities of b_2 and B_2 are the same.

Figure 4 shows the long-term expected reward of the MFC system when the maximal number of vehicles in the MFC system changes. We can see that, when the amounts of vehicles increase, the proposed strategy has a significantly larger system reward than the greedy algorithm. This is because when the available computing resource increases, more tasks can be processed by the vehicular fog. Moreover, we can find that the proposed offloading strategy performs well in the system reward as compared to the GA method. This is because the proposed scheme considers the long-

term reward when assigning RUs to execute tasks. However, the GA method only allocates as many RUs as possible to process tasks and does not take into account the long-term system reward.

Next, we compare the proposed task offloading strategy for the high priority tasks with that for the low priority tasks, which is shown in Tables 2 and 3, respectively. Note that the blank part of the tables indicates the corresponding state does not exist. It can be seen that when the number of available resources is larger than the maximal number of RUs processing a task, i.e., N , the system allocates as many RUs as possible to execute the high/low priority tasks to maximize the system reward. Moreover, when the number of available RUs is smaller than N and larger than one, the system assigns one RU to process computing tasks. When the number of available resources is very small, the system will adopt different action to execute high/low priority tasks, i.e., transmitting the high priority task to the remote cloud to obtain maximal long-term expected reward and allocating the one RU to process the low priority task or rejecting the low priority task.

7. Conclusions

In this paper, we developed a task offloading strategy for the MFC system to maximize the system reward in term of reducing the processing time of tasks while considering the impact of computation requirements of tasks which are transmitted by different ACs of the 802.11p EDCA mechanism, mobility of vehicles, and the arrival/departure of high/low priority tasks. We first transformed the offloading problem into an SMDP model. Afterwards, the relative value iterative algorithm was used to solve the model to obtain the optimal strategy. Finally, we demonstrated the performance of the proposed scheme by comparing it to the GA method. In the future, we will consider the task queuing time and study the task offloading problem in the vehicle platoon.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61701197, in part by the 111 Project under Grant No. B12018, in part by the Foundation for Jiangsu key Laboratory of Traffic and Transportation Security under Grant No. TTS2020-02, and in part by the Jiangsu Laboratory of Lake Environment Remote Sensing Technologies Open Fund under Grant No. JSLERS-2020-001.

References

- [1] F. Haider, C. X. Wang, B. Ai, H. Haas, and E. Hepsaydir, "Spectral/energy efficiency tradeoff of cellular systems with mobile femtocell deployment," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3389–3400, 2016.
- [2] H. Zheng, K. Xiong, P. Fan, Z. Zhong, and K. B. Letaief, "Fog-assisted multiuser SWIPT networks: local computing or offloading," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5246–5264, 2019.
- [3] Q. Wu, S. Xia, P. Fan, Q. Fan, and Z. Li, "Velocity-adaptive V2I fair-access scheme based on IEEE 802.11 DCF for platooning vehicles," *Sensors*, vol. 18, no. 12, article 4198, 2018.
- [4] Z. Ning, P. Dong, X. Wang et al., "Deep reinforcement learning for intelligent internet of vehicles: an energy-efficient computational offloading scheme," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1060–1072, 2019.
- [5] Y. Liu, K. Xiong, Q. Ni, P. Fan, and K. B. Letaief, "UAV-assisted wireless powered cooperative mobile edge computing: joint offloading, CPU control and trajectory optimization," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2777–2790, 2020.
- [6] C. Zhu, J. Tao, G. Pastor et al., "Folo: latency and quality optimized task allocation in vehicular fog computing," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4150–4161, 2019.
- [7] H. Zheng, K. Xiong, P. Fan, L. Zhou, and Z. Zhong, "SWIPT-aware fog information processing: local computing vs. fog offloading," *Sensors*, vol. 18, no. 10, article 3291, 2018.
- [8] K. Zheng, H. Meng, P. Chatzimisios, L. Lei, and X. Shen, "An SMDP-based resource allocation in vehicular cloud computing systems," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7920–7928, 2015.
- [9] Q. Wu, S. Xia, Q. Fan, and Z. Li, "Performance analysis of IEEE 802.11p for continuous backoff freezing in IoV," *Electronics*, vol. 8, no. 12, article 1404, 2019.
- [10] Q. Wu and J. Zheng, "Performance modeling of the IEEE 802.11p EDCA mechanism for VANET," in *2014 IEEE Global Communications Conference*, pp. 57–63, Austin, TX, USA, December 2014.
- [11] Q. Wu, H. Liu, R. Wang, P. Fan, Q. Fan, and Z. Li, "Delay sensitive task offloading in the 802.11p based vehicular fog computing systems," *IEEE Internet of Things Journal*, vol. 7, no. 1, pp. 773–785, 2020.
- [12] Z. Zhou, P. Liu, J. Feng, Y. Zhang, S. Mumtaz, and J. Rodriguez, "Computation resource allocation and task assignment optimization in vehicular fog computing: a contract-matching approach," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3113–3125, 2019.
- [13] J. Zhao, M. Kong, Q. Li, and X. Sun, "Contract-based computing resource management via deep reinforcement learning in vehicular fog computing," *IEEE Access*, vol. 8, pp. 3319–3329, 2019.
- [14] F. Lin, Y. Zhou, G. Pau, and M. Collotta, "Optimization-oriented resource allocation management for vehicular fog computing," *IEEE Access*, vol. 6, pp. 69294–69303, 2018.
- [15] J. Xie, Y. Jia, Z. Chen, Z. Nan, and L. Liang, "Efficient task completion for parallel offloading in vehicular fog computing," *China Communications*, vol. 16, no. 11, pp. 42–55, 2019.
- [16] C. C. Lin, D. J. Deng, and C. C. Yao, "Resource allocation in vehicular cloud computing systems with heterogeneous

- vehicles and roadside units,” *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3692–3700, 2018.
- [17] J. Zhao, Q. Li, Y. Gong, and K. Zhang, “Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7944–7956, 2019.
 - [18] Q. Wu, H. Ge, H. Liu, Q. Fan, Z. Li, and Z. Wang, “A task offloading scheme in vehicular fog and cloud computing system,” *IEEE Access*, vol. 8, pp. 1173–1184, 2020.
 - [19] Z. Wang, Z. Zhong, and M. Ni, “Application-aware offloading policy using smdp in vehicular fog computing systems,” in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Kansas City, MO, USA, May 2018.
 - [20] P. Liu, J. Li, and Z. Sun, “Matching-based task offloading for vehicular edge computing,” *IEEE Access*, vol. 7, pp. 27628–27640, 2019.
 - [21] J. Zheng and Q. Wu, “Performance modeling and analysis of the IEEE 802.11p EDCA mechanism for VANET,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 4, pp. 2673–2687, 2016.
 - [22] Y. Kim, Y. H. Bae, D. S. Eom, and B. D. Choi, “Performance analysis of a MAC protocol consisting of EDCA on the CCH and a reservation on the SCHs for the IEEE 802.11p/1609.4 WAVE,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 5160–5175, 2016.
 - [23] Q. Wu and J. Zheng, “Performance modeling and analysis of the ADHOC MAC protocol for vehicular networks,” *Wireless Networks*, vol. 22, no. 3, pp. 799–812, 2016.
 - [24] Q. Wu, H. Zhang, Z. Li, Y. Liu, and C. Zhang, “Performance evaluation of the V2I fair access with a finite retry limit,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2018, no. 1, p. 20, 2018.
 - [25] H. Peng, D. Li, K. Abboud et al., “Performance analysis of IEEE 802.11p DCF for multiplatooning communications with autonomous vehicles,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2485–2498, 2017.
 - [26] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang, “Computation offloading and resource allocation in wireless cellular networks with mobile edge computing,” *IEEE Transactions on Wireless Communications*, vol. 16, no. 8, pp. 4924–4938, 2017.
 - [27] Y. Yao, L. Rao, and X. Liu, “Performance and reliability analysis of IEEE 802.11p safety communication in a highway environment,” *IEEE Transactions on Vehicular Technology*, vol. 62, no. 9, pp. 4198–4212, 2013.
 - [28] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, 2014.
 - [29] M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, Wiley, New York, NY, USA, 2005.
 - [30] K. A. Hafeez, A. Anpalagan, and L. Zhao, “Optimizing the control channel interval of the DSRC for vehicular safety applications,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 5, pp. 3377–3388, 2016.