Research Article

A Resource Allocation Scheme for Real-Time Energy-Aware Offloading in Vehicular Networks with MEC

Haibo Zhang ^{1,2}, Xiangyu Liu ^{1,2}, Xia Bian ^{1,2}, Yan Cheng ¹ and Shengting Xiang ¹

¹*School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China*

²*Engineering Research Center of Mobile Communications, Ministry of Education, Chongqing 400065, China*

Correspondence should be addressed to Xia Bian; s190131078@stu.cqupt.edu.cn

Received 11 June 2021; Revised 10 November 2021; Accepted 5 December 2021; Published 10 February 2022

Academic Editor: Omprakash Kaiwartya

Copyright © 2022 Haibo Zhang 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.

With the emergence of new vehicular applications, computation offloading based on mobile edge computing (MEC) has become a promising paradigm in resource-constrained vehicular networks. However, an unreasonable offloading strategy in offloading can cause serious energy consumption and latency. A real-time energy-aware offloading scheme for vehicle networks, based on MEC, is proposed to optimize communication and computation resource to decrease energy consumption and latency. Because the problem of computation offloading and resource allocation is the mixed-integer nonlinear problem (MINLP), this article uses a bi-level optimization method to transform the original MINLP into two subproblems. Furthermore, considering the mobility of vehicle users (V-UEs) and the availability of cloud resources, an offloading scheme based on deep reinforcement learning (DRL) is adopted to help users make the optimal offloading decisions. The simulation results show that the proposed bi-level optimization algorithm reduces the total overhead by nearly 40% to the compared algorithm.

1. Introduction

With the continuous progress of the sixth-generation (6G) communication and the Internet of vehicles [1–6], novel mobile applications in vehicular networks have promoted demand for the low-delay high-quality services, for example, interactive gaming, augmented reality/virtual reality (AR/VR), face recognition, and natural language processing [7]. The demand of computing is prominent, so that it frequently exceeds the capacity that local mobile devices can provide [8]. In general, vehicle units have limited computation resources (e.g., central process unit (CPU) frequency and memory) and battery lifetime, which will bring an unprecedented challenge for effectively executing these mobile applications [9–11]. To cope with the explosive application demands of vehicular terminals, tasks of vehicles are migrated from the vehicular units to the cloud servers [12], referred to as computation offloading. However, the cloud servers are spatially far from mobile vehicles, which may cause high transmission latency and severe degradation of the offloading efficiency.

Mobile edge computing (MEC) technology [13, 14] provided the cloud computation resource close to the mobile vehicular terminals. Compared with MCC, MEC can provide lower latency, high bandwidth, and computational flexibility in computation offloading. In [15], the authors studied the offloading problem among multiple devices for mobile edge cloud computing and proposed a game-theoretic approach for achieving efficient offloading mechanisms. To minimize the energy consumption of the MEC offloading system, the authors in [10] proposed an energy-efficient computation offloading scheme, which jointly optimized the offloading decisions and the radio resource allocation strategies. In [16], the authors proposed the energy-aware offloading scheme to investigate the trade-off between the energy consumption and the latency, optimizing the computation offloading and resource allocation under the limited energy and sensitive latency. But the weighting factor to investigate the trade-off between energy consumption and latency is a constant, which can define the value of the weighting factor subjectively. The work ignored

the service conditions of the battery and the user-specific demands. In addition, the computation offloading, especially in ultra-dense networks (UDNs), may cause more interference and result in unexpected transmission latency [17]. Therefore, it is unreasonable to offload all computation tasks to the MEC servers. As a result, it is critical to make an efficient offloading decision and study the trade-off between the energy consumption of vehicle units and the latency of the corresponding tasks. Based on the above discussion, in this article, we propose a real-time energy-aware offloading scheme to study the trade-off between the energy consumption and the task latency of the vehicle units, which will optimize the allocation of communication and computing resources. The motivations behind our work are attributed to the following observations:

- (1) Build a network scenario that deploys multiple MEC servers and multiple request vehicles, where service nodes (MEC servers and vehicles) are equipped with limited wireless and computational resource.
- (2) Because of the limited computation resource, the MEC-enabled base stations (MEC-BSs) cannot provide endless computation offloading services for all tasks from V-UEs. Therefore, the computation offloading decision should be rationally determined.
- (3) The energy consumption and latency affect the quality of service of the vehicle units. The energy consumption and latency depend mainly on allocating transmission power and wireless channels when offloading the tasks to the MEC server.
- (4) Depending on the using conditions of a battery of V-UE and user-specific demands, the user preference (i.e., the weighting factor) should be redefined to allow vehicle units to choose different optimization objectives according to the actual energy consumption.

Therefore, based on the above discussions, we propose a real-time energy-aware offloading scheme to study the trade-off between the energy consumption and latency (transmission latency and execution latency) of the vehicle units, which optimizes the allocation of communication and computing resource. The main contributions of this article are as follows:

- (i) We present an integrated framework for computation offloading and resource allocation in multi-cell scenarios. For V-UEs, we present a real-time energy-aware offloading scheme that combines with computation and communication resource allocation to minimize the weighted sum of energy consumption and latency.
- (ii) We redefine the weighting factor with the residual energy, which is no longer the weighting factor of the V-UEs' subjective energy consumption and latency. The advantage of the updated definition is that the actual energy consumption is taken into account, and it can help maintain the lifetime of vehicle users.

- (iii) For the mixed-integer non-linear problem (MINLP) of computation offloading and resource allocation, a bi-level optimization approach is adopted to transform the original MINLP problem into two subproblems, that is, a lower-level problem for seeking the allocation of power and channels, and an upper-level problem for task offloading. Furthermore, taking into account the mobility of vehicle users (V-UEs) and the availability of cloud resources, an offloading scheme based on deep reinforcement learning (DRL) is proposed to enable the users to make the optimal offloading decisions.

The rest of the article is organized as follows: Section 2 describes the system model and problem formulation. Section 3 presents the system model and problem formulations. Section 4 presents the optimal computation offloading and resource allocation scheme. The simulation results are discussed in Section 5, and Section 6 concludes the article.

2. Related Works

2.1. MEC-Enabled Vehicular Networks. The MEC-enabled vehicular network has been extensively studied recently, since it can support the communication of vehicles and edge servers for real-time information sharing in the Internet of vehicles. In [18], the authors presented a cooperative autonomous driving-oriented MEC-enabled 5G-V2X prototype system design with the knowledge of big data gathered from the environment. The authors in [19] geared the intelligent battery switch service management for electric vehicles by a MEC-driven architecture. In [20], the application of multisource data fusion to support vehicle on-road analysis was proposed. It can provide some services for vehicles of peeking around the corner, extending sensing range and reinforcing local observations. In [21], the traffic prediction algorithm based on MEC was proposed for addressing short-term traffic prediction. A positioning calibration mechanism by calculating the one-way delay existing in the transmission of vehicles based on MEC and global positioning system (GPS) was proposed in [22]. The above works demonstrated that the combination of MEC and vehicular network can be meaningful. For realizing these applications, the resource allocation and offloading scheme become topics worth studying.

2.2. Offloading Scheme in Vehicular Networks. In order to reduce the latency and the energy consumption in the network as well as ensure the reliability of V-UEs, many works have studied resource allocation and offloading scheme in the vehicular network. The authors in [23] designed a distributed best response algorithm based on the computation offloading game model to maximize the utility of each vehicle. But this work only regarded the MEC server as the object to offload tasks. In [24], the authors proposed a joint allocation of wireless resource and MEC computing resource algorithm including V2X links clustering and MEC computation resource scheduling. A hierarchical

architecture was constructed based on MEC in [25] to minimize the total network delay, which jointly considers vehicle-to-RSU computation offloading, RSU peer offloading, and content caching. In [26], the authors constructed a three-layer architecture in vehicular network and formulated an energy-efficient computational offloading problem to minimize the overall power consumption while satisfying the delay constraint. The authors in [27] constructed an SDN-assisted MEC architecture for the vehicular network to reduce the system overhead by optimizing offloading decision, transmission power control, subchannels assignment, and computing resource allocation scheme. However, the above works ignored the dynamic balance of the latency and the energy consumption.

Different from the above computation offloading and resource allocation strategies for vehicular MEC networks, we designed a real-time energy-aware offloading scheme that combines with computation and communication resource allocation considering the task deadline constraint and the balance of the energy consumption and latency. On this basis, we proposed an offloading and resource allocation scheme based on deep reinforcement learning (DRL) to minimize the total overhead.

3. System Model and Problem Formulations

3.1. System Model. M Road Side Units (RSUs) are located along the straight road, each of which is connected to a MEC server. Each RSU covers one region, and vehicles can only access an RSU in the region where they are located. The vehicle can communicate with neighboring vehicles or RSUs to send or receive information. As shown in Figure 1, L request vehicles with a Poisson distribution are deployed, expressed as $V_i (i \in \{1, 2, \dots, L\})$. Each RSU serves $U_j (j \in \{1, 2, \dots, M\})$ V-UEs. To multiplex spectrum, we assume that multiple RSUs share a single spectral resource, so there exists interference between RSUs. The bandwidth W is divided into N channels. V-UEs connect to the RSUs in the orthogonal frequency-division multiple access (OFDMA) mode, where each V-UE channel in the same RSU is orthogonal to others. In this network, V-UE i in RSU j has a computation-intensive task $A_{i,j} = \{d_{i,j}, c_{i,j}, D_{i,j}^{\max}\}$ to be completed, in which $d_{i,j}$ is the size of computation input data, $c_{i,j}$ denotes the total number of CPU cycle required to accomplish the computation task, and $D_{i,j}^{\max}$ denotes the maximal latency that the V-UE can tolerate. For each V-UE, its task can be executed either locally on itself or remotely in the MEC server via computation offloading. Let $s_{i,j}$ be the offloading decision of V-UE i in RSU j . If the V-UE i in RSU j offloads its task to the MEC server, $s_{i,j} = 1$, and $s_{i,j} = 0$ otherwise. In order to increase the readability of the article, the definitions of some variables are listed in Table 1.

3.1.1. Local Computing. Define $f_{i,\text{loc}}$ as the computation power (i.e., CPU cycles per second) of V-UEs. When the task $A_{i,j}$ is executed locally, the computation execution time $t_{i,j}^L$ is expressed as

FIGURE 1: System model.

$$t_{i,j}^L = \frac{c_{i,j}}{f_{i,\text{loc}}}. \quad (1)$$

The corresponding energy consumption of the V-UE for local execution is given by

$$e_{i,j}^L = k(f_{i,\text{loc}})^2 c_{i,j}, \quad (2)$$

where $k = 10^{-26}$ is a coefficient that depends on the chip architecture [28, 29].

Note that $f_{i,\text{loc}}$ affects the computation time and energy consumptions simultaneously. The CPU cycle frequency can be scheduled via dynamic voltage and frequency scaling (DVS) technology [30].

3.1.2. MEC Edge Computing. When the input data are transmitted to the MEC server through RSU, the transmission expenditure between the MEC server and the RSU is neglected [31, 32]. If the V-UE accesses the RSU over channel n , the achievable uplink transmission rate can be represented as

$$R_{i,j,n} = w \log_2 \left(1 + \frac{p_{i,j,n} h_{i,j,n}}{\sigma^2 + I_{i,j,n}} \right), \quad (3)$$

where w is the channel bandwidth, $w = W/N$. $p_{i,j,n}$ and $h_{i,j,n}$ stand for the uplink transmission power and channel gain between the V-UE i and the RSU j via channel n , respectively. σ^2 denotes the noise power. $I_{i,j,n}$ is the interference of V-UE i in RSU j suffering from other V-UEs in adjacent RSUs on the same channel n , and it can be expressed as

$$I_{i,j,n} = \sum_{k=1}^{U_l} \sum_{l=1, l \neq j}^M a_{k,l,n} p_{k,l,n} h_{k,l,n}^j, \quad (4)$$

where l denotes the l th except the j th RSU, $h_{k,l,n}^j$ is the channel gain from V-UE k in RSU j over channel n , and U_l is the number of V-UEs in RSU l . Hence, the total uplink transmission rate for V-UE i in RSU j can be obtained as

$$R_{i,j} = \sum_{n=1}^N a_{i,j,n} R_{i,j,n}, \quad (5)$$

TABLE 1: Abbreviated parameters.

Variable	Concept
M	The number of RSUs
L	The number of request vehicles
N	The number of channels
V_i	The request vehicles
U_j	The RSU serves
$d_{i,j}$	The size of computation input data
$c_{i,j}$	The total number of CPU cycle to accomplish the computation task
$D_{i,j}^{\max}$	The tolerable latency for the task
$s_{i,j}$	The offloading decision of V-UE i in RSU j
$f_{i,\text{loc}}$	The computation ability of V-UE
f^C	The computation ability of MEC server
$t_{i,j}^L$	The time of local computation
$t_{i,j}^C$	The time of transmission
$e_{i,j}^L$	The energy consumption of local computation
$e_{i,j}^C$	The energy transmission of local computation
k	The coefficient that depends on the chip architecture
$R_{i,j,n}$	The transmission rate
$P_{i,j,n}$	The transmission power
$h_{i,j,n}$	The channel gain
$I_{i,j,n}$	The interference of V-UE i in RSU j
$a_{i,j,n}$	The factor of channel selection
$w_{i,j}$	The weighting factor
$r_{i,j}^E$	The residual energy rate
$O_{i,j}$	The weighted sum of energy consumption and latency

where $a_{i,j,n} \in \{0, 1\}$. $a_{i,j,n} = 1$ indicates that the channel n is assigned to V-UE i in RSU j to offload its task, and $a_{i,j,n} = 0$ otherwise. Let f^C express the CPU cycle frequency of the MEC server, which is fixed during computation task execution [30, 33]. Then, the total edge computing time of the task includes the transmission time and computation time on the MEC server, which can be written as

$$t_{i,j}^C = \frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C}. \quad (6)$$

The corresponding energy consumption of V-UE is expressed as

$$e_{i,j}^C = \sum_{n=1}^N a_{i,j,n} P_{i,j,n} \frac{d_{i,j}}{R_{i,j}}. \quad (7)$$

Here, we assume that the time and energy consumption of computation result from the MEC server to the V-UE are neglected in this case because the size of output data is much smaller than the size of input data and the download data rate is very high in general, which is similar to the study [32].

During the execution of a task, both latency and energy consumption will affect the V-UEs (i.e., battery energy limitation of V-UEs). Hence, we introduce the weighting factor $w_{i,j}$ ($w_{i,j} \in [0, 1]$) to study the trade-off between latency and energy consumption, which can define to meet the user-specific demands [31]. Energy and latency can be saved by adjusting the weighting factor. However, we introduce the residual energy rate $r_{i,j}^E$ of the battery into our model's weighting factor. It is defined as

$$w'_{i,j} = w_{i,j} r_{i,j}^E, \quad (8)$$

where $r_{i,j}^E = E^{\text{total}} - [(1 - s_{i,j}) * e_{i,j}^L + s_{i,j} * e_{i,j}^C] / E^{\text{total}}$. Molecule $E^{\text{total}} - [(1 - s_{i,j}) * e_{i,j}^L + s_{i,j} * e_{i,j}^C]$ is the residual energy of V-UE i in RSU j . E^{total} is the battery capacity in Joules [34]. $r_{i,j}^E$ is a variable that affects the real-time service conditions of the battery.

According to (1) and (2), the overhead of the task locally computed on V-UE i in RSU j , namely, the weighted sum of energy consumption and latency $O_{i,j}^L$, can be obtained as

$$O_{i,j}^L = w'_{i,j} * t_{i,j}^L + (1 - w'_{i,j}) * e_{i,j}^L. \quad (9)$$

Let $w_{i,j}^t = w'_{i,j}$, and $w_{i,j}^e = (1 - w'_{i,j})$. Equation (9) can be simplified to

$$O_{i,j}^L = w_{i,j}^t * t_{i,j}^L + w_{i,j}^e * e_{i,j}^L. \quad (10)$$

Similarly, the overhead of the task computed in the MEC sever can be expressed as

$$O_{i,j}^C = w_{i,j}^t * t_{i,j}^C + w_{i,j}^e * e_{i,j}^C. \quad (11)$$

Therefore, the overhead of the V-UE i in RSU j can be obtained by

$$O_{i,j} = s_{i,j} * O_{i,j}^C + (1 - s_{i,j}) * O_{i,j}^L. \quad (12)$$

3.2. Problem Formulation. As mentioned above, we model the problem in a multicell scenario. In the multicell network, interference management and computation offloading are considered [16]. We formulate the offloading and resource allocation for the MEC system as an optimization problem.

The objective is to minimize the overhead of V-UEs, which is formulated as follows:

$$\begin{aligned}
P1: \quad & \min_{s, p, a, f} \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[w_{i,j}^t \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) + w_{i,j}^e \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} \right] \right\} + \\
& \sum_{i=1}^{U_j} \sum_{j=1}^M \left\{ (1 - s_{i,j}) \left[w_{i,j}^t \frac{c_{i,j}}{f_{i,\text{local}}} + w_{i,j}^e k(f_{i,\text{local}})^2 c_{i,j} \right] \right\}, \\
s.t. \quad & C1: s_{i,j} \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) + (1 - s_{i,j}) \frac{c_{i,j}}{f_{i,\text{local}}} \leq D_{i,j}^{\max}, \\
& C2: s_{i,j} \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} + (1 - s_{i,j}) w_{i,j}^e k(f_{i,\text{local}})^2 c_{i,j} \leq E_{i,j}^{\max}, \forall i, j, \\
& C3: f_{\text{local}}^{\min} \leq f_{i,\text{local}} \leq f_{\text{local}}^{\max}, \forall i, j, \\
& C4: 0 \leq \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \leq p^{\max}, \forall i, j, \\
& C5: \sum_{k=1}^{U_l} \sum_{l=1, l \neq j}^M a_{k,l,n} p_{k,l,n} h_{k,l,n} \leq I, \forall j, n, \\
& C6: \sum_{n=1}^N a_{i,j,n} \leq 1, \forall i, j, \\
& C7: a_{i,j,n} \in \{0, 1\}, \forall i, j, n, \\
& C8: s_{i,j} \in \{0, 1\}, \forall i, j.
\end{aligned} \tag{13}$$

Constraint C1 is the maximum tolerance latency. C2 is the energy consumption that should not exceed the residual energy of the V-UE. C3 restricts the local CPU cycle frequency into a finite set of values. C4 guarantees the maximum transmission power on V-UE i in RSU j . C5 addresses that the interference on RSU j caused by V-UEs with offloading tasks in other RSUs on each channel must meet the predefined threshold I . C6 indicates that each V-UE can only be allocated at most one channel. C7 states the binary variables used to present channel allocation. $a_{i,j,n} = 1$ indicates that V-UE i uses subchannel n for task offloading, and $a_{i,j,n} = 0$ otherwise. C8 denotes offloading decisions as to the binary variables. In other words, C8 specifies that each V-UE completes its task either by local execution or by edge execution.

4. Optimal Computation Offloading and Resource Allocation Scheme

4.1. Optimal Local Computing via CPU Cycle Frequency Scheduling. The latency and energy consumption of local

computing is determined by local computing capability $f_{i,\text{loc}}$. To minimize the local overhead, we can schedule the CPU cycle frequency of V-UE. Considering the constraints of C1, C2, and C3 in problem P1, P1.1 can be formed as follows:

$$\begin{aligned}
P1.1: \quad & \min_f w_{i,j}^t \frac{c_{i,j}}{f_{i,\text{loc}}} + w_{i,j}^e k(f_{i,\text{loc}})^2 c_{i,j}, \\
s.t. \quad & C1: \frac{c_{i,j}}{f_{i,\text{loc}}} \leq D_{i,j}^{\max}, \\
& C2: k(f_{i,\text{loc}})^2 c_{i,j} \leq E_{i,j}^{\max}, \\
& C3: f_l^{\min} \leq f_{i,\text{loc}} \leq f_l^{\max}.
\end{aligned} \tag{14}$$

Let $O_{i,j}^L(f_{i,\text{loc}}) = w_{i,j}^t c_{i,j} / f_{i,j} + w_{i,j}^e k(f_{i,\text{loc}})^2 c_{i,j}$; we can see that the value of $O_{i,j}^L(f_{i,\text{loc}})$ depends only on $f_{i,\text{loc}}$. We take the derivative of $O_{i,j}^L(f_{i,\text{loc}})$ with respect to $f_{i,\text{loc}}$ and set it to be zero. Then, we can obtain

$$\frac{4(1-s_{i,j})k^2c_{i,j}^2(f_{i,\text{loc}})^5}{E^{\text{total}}} + \frac{2s_{i,j}e_{i,j}^Ckc_{i,j}(f_{i,\text{loc}})^3}{E^{\text{total}}} - \frac{(1-s_{i,j})kc_{i,j}^2(f_{i,\text{loc}})^2}{E^{\text{total}}} + \frac{s_{i,j}c_{i,j}e_{i,j}^C}{E^{\text{total}}} - c_{i,j} = 0. \quad (15)$$

According to [25], it can be seen that $f_{i,\text{loc}}$ is a large order of magnitude value; therefore, $(f_{i,\text{loc}})^2$ and $(f_{i,\text{loc}})^3$ are negligible. Then, the above equation can be obtained as follows:

$$f^* = \sqrt[5]{\frac{E^{\text{total}} - s_{i,j}e_{i,j}^C}{4(1-s_{i,j})c_{i,j}k}}. \quad (16)$$

For $f_{i,\text{loc}} > f^*$, $O_{i,j}^L(f_{i,\text{loc}})$ monotonously increases with the increase of $f_{i,\text{loc}}$. Otherwise, it monotonously decreases with the increase of $f_{i,\text{loc}}$. According to C1 and C2, we have

$$\begin{aligned} f_{i,\text{loc}} &\geq \frac{c_{i,j}}{D_{i,j}^{\text{max}}} = f_l, \\ f_{i,\text{loc}} &\leq \sqrt{\frac{E_{i,j}^{\text{max}}}{kc_{i,j}}} = f_h. \end{aligned} \quad (17)$$

Combining (16) with the constraint C3, we define

$$\begin{aligned} f_l' &= \max\{f_l^{\min}, f_l\}, \\ f_h' &= \min\{f_l^{\max}, f_h\}. \end{aligned} \quad (18)$$

To ensure the feasible region of $f_{i,\text{loc}}$ to be nonempty, $f_l' \leq f_h'$ should hold. Hence, the optimal $O_{i,j}^L(f_{i,\text{loc}})$ can be calculated as

$$O_{i,j}^{L*}(f_{i,\text{loc}}) = \begin{cases} O_{i,j}^L(f_l'), & f^* \leq f_l', \\ O_{i,j}^L(f^*), & f_l' < f^* \leq f_h', \\ O_{i,j}^L(f_h'), & f^* > f_h'. \end{cases} \quad (19)$$

4.2. Optimal Edge Computing through Resource Allocation and Computation Offloading. In a multicell scenario, we combine power allocation, channel allocation, and computation offloading to minimize the V-UEs' weighted sum of latency and energy consumption. Because there are two binary variables, a large number of other variables and interference terms exist. Therefore, the problem is nonconvex and MINLP, the bi-level optimization approach is taken to solve it. The original MINLP problem is decoupled into a lower-level problem for seeking the optimal power allocation and optimal channel allocation and an upper-level issue for task offloading [35]. Simultaneously, considering the mobility of vehicle users (V-UEs) and the availability of cloud resources, an offloading scheme based on DRL is proposed to enable the users to make the optimal offloading decisions [36].

$$\begin{aligned} \text{P1.2: } \min_{s,p,a} F(s, p, a) &= \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[w_{i,j}^t \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) + w_{i,j}^e \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} \right] \right\} + \\ &\sum_{i=1}^{U_l} \sum_{j=1}^M \{(1-s_{i,j})O_{i,j}^{L*}\}, \\ \text{s.t. C1: } &s_{i,j} \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) + (1-s_{i,j}) \frac{c_{i,j}}{f_{i,\text{local}}} \leq D_{i,j}^{\text{max}}, \forall i, j, \\ \text{C2: } &s_{i,j} \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} + (1-s_{i,j}) w_{i,j}^e k (f_{i,\text{local}})^2 c_{i,j} \leq E_{i,j}^{\text{max}}, \forall i, j, \\ \text{C3: } &f_{\text{local}}^{\min} \leq f_{i,\text{local}} \leq f_{\text{local}}^{\max}, \forall i, j, \\ \text{C4: } &0 \leq \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \leq p^{\text{max}}, \forall i, j, \\ \text{C5: } &\sum_{k=1}^{U_l} \sum_{l=1, l \neq j}^M a_{k,l,n} p_{k,l,n} h_{k,l,n} \leq I, \forall j, n, \\ \text{C6: } &\sum_{n=1}^N a_{i,j,n} \leq 1, \forall i, j, \\ \text{C7: } &a_{i,j,n} \in \{0, 1\}, \forall i, j, n, \\ \text{C8: } &s_{i,j} \in \{0, 1\}, \forall i, j. \end{aligned} \quad (20)$$

Because both the objective function and constraints are nonconvex, P1.2 is a nonconvex problem and belongs to mixed-integer non-linear problem.

Proof: the proof of nonconvexity of objective function is shown in Appendix A. The proof of nonconvexity of constraints refers to the proof of objective function

However, formula (21) is an MINLP and it is very hard to find its optimal solution. Next, we use the bi-level optimization approach to solve it. Note that P1.2 involves two embedded problems. One is to calculate the computing overhead of the entire system only when the offloading strategy s is known, and the other is to be inversely influenced by power p and channel allocation strategy a .

4.3. Bi-Level Optimization Approach. In the view of P1.2, we adopt a bi-level optimization approach to solve the original

problem. The bi-level optimization problem is treated as a multistage game between the upper problem and the lower problem. Firstly, given the task offloading strategy s , the optimal power allocation p and channel allocation strategy a can be solved by function $F(s, p, a)$. Then, according to the optimal power $p^*(s)$ and channel allocation strategy $a^*(s)$, which have been calculated in lower-level problem, the optimal task offloading strategy s^* is solved by function $F(s, p^*(s), a^*(s))$ in the upper-level problem.

4.3.1. Lower-Level Problem. Given the task offloading strategy s , the local overhead becomes the known quantity, and the lower-level problem can be written in the form of P1.2 as follows:

$$\begin{aligned}
 P2: \min_{s,p,a} F(p, a) &= \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[w_{i,j}^t \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) + w_{i,j}^e \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} \right] \right\} + \sum_{i=1}^{U_j} \sum_{j=1}^M \{(1 - s_{i,j}) O_{i,j}^{L*}\}, \\
 s.t. \text{ C1: } &s_{i,j} \left(\frac{d_{i,j}}{R_{i,j}} + \frac{c_{i,j}}{f^C} \right) \leq D_{i,j}^{\max}, \forall i, j, \\
 \text{C2: } &s_{i,j} \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} \leq E_{i,j}^{\max}, \forall i, j \\
 \text{C4: } &0 \leq \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \leq p^{\max}, \forall i, j \\
 \text{C5: } &\sum_{k=1}^{U_l} \sum_{l=1, l \neq j}^M a_{k,l,n} p_{k,l,n} h_{k,l,n} \leq I, \forall j, n \\
 \text{C6: } &\sum_{n=1}^N a_{i,j,n} \leq 1, \forall i, j, \\
 \text{C7: } &a_{i,j,n} \in \{0, 1\}, \forall i, j, n, \\
 \text{C8: } &s_{i,j} \in \{0, 1\}, \forall i, j.
 \end{aligned} \tag{21}$$

Constraint C1 can be written as $s_{i,j} d_{i,j}/R_{i,j} \leq \chi_i$, where $\chi_i = D_{i,j}^{\max} - s_{i,j} c_{i,j}/f^C$. $s_{i,j} d_{i,j}/R_{i,j}$ is upper-bounded by χ_i for all feasible solutions. By replacing each term

$s_{i,j} d_{i,j}/R_{i,j}$ in the objective function of P2 with its upper bound $s_{i,j} \chi_i$, we obtain the following convex approximation of P2.1:

$$\begin{aligned}
P2.1: \min_{p,a} F(p,a) &= \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[w_{i,j}^t \left(\chi_i + \frac{c_{i,j}}{f^C} \right) + w_{i,j}^e \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \chi_i \right] \right\} + \sum_{i=1}^{U_j} \sum_{j=1}^M \{ (1 - s_{i,j}) O_{i,j}^{L*} \}, \\
s.t. \text{ C1: } & s_{i,j} \frac{d_{i,j}}{R_{i,j}} \leq \chi_i, \forall i, j, \\
\text{C2: } & s_{i,j} \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \frac{d_{i,j}}{R_{i,j}} \leq E_{i,j}^{\max}, \forall i, j, \\
\text{C3: } & 0 \leq \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \leq p^{\max}, \forall i, j, \\
\text{C5: } & \sum_{k=1}^{U_l} \sum_{l=1, l \neq j}^M a_{k,l,n} p_{k,l,n} h_{k,l,n} \leq I, \forall j, n, \\
\text{C7: } & a_{i,j,n} \in \{0, 1\}, \forall i, j, n.
\end{aligned} \tag{22}$$

Obviously, P2.1 is almost a strictly convex problem, except for the discrete channel assignment value $a_{i,j,n}$. Relaxing the integer variable $a_{i,j,n}$ to be continuous between

[0, 1], the problem of P2.1 is written as the Lagrangian function:

$$\begin{aligned}
L(\lambda_1, \lambda_2, \lambda_3, p, a) &= \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[w_{i,j}^t \left(\chi_i + \frac{c_{i,j}}{f^C} \right) + w_{i,j}^e \sum_{n=1}^N a_{i,j,n} p_{i,j,n} \chi_i \right] \right\} + \sum_{i=1}^{U_j} \sum_{j=1}^M \{ (1 - s_{i,j}) O_{i,j}^{L*} \} \\
&+ \lambda_1 \left(\frac{s_{i,j} d_{i,j}}{\chi_i} - R_{i,j} \right) + \lambda_2 \left(\frac{s_{i,j} a_{i,j,n} p_{i,j,n} d_{i,j}}{E_{i,j}^{\max}} - R_{i,j} \right) + \lambda_3 (a_{i,j,n} p_{i,j,n} - p^{\max}).
\end{aligned} \tag{23}$$

For fixed s , the optimal power and channel allocation value can be obtained by partial derivation of the $p_{i,j,n}$ and $a_{i,j,n}$ in formula (23). Therefore, the deviation of $p_{i,j,n}$ in formula (23) can be calculated:

$$\frac{\partial L(\lambda_1, \lambda_2, \lambda_3, p, a)}{\partial p_{i,j,n}} \Big|_{p_{i,j,n} = p_{i,j,n}^*} = 0. \tag{24}$$

Then, we can obtain the optimal power of V-UE i on channel n .

$$\begin{aligned}
p_{i,j,n}^* &= \frac{-B + \sqrt{B^2 + 4AC}}{2A}, \\
A &= \frac{2s_{i,j}(1 - s_{i,j})a_{i,j,n}^2 w_{i,j} \chi_i}{E^{\text{total}}}, \\
B &= (1 - w_{i,j})a_{i,j,n} s_{i,j} \chi_i + \frac{s_{i,j}^2 e_{i,j}^L w_{i,j} a_{i,j,n} \chi_i}{E^{\text{total}}} + \lambda_2 \frac{a_{i,j,n} s_{i,j} d_{i,j}}{E_{i,j}^{\max}} - \frac{(1 - s_{i,j})a_{i,j,n} s_{i,j} \chi_i^2}{E^{\text{total}}} - \frac{(1 - s_{i,j})a_{i,j,n} s_{i,j} w_{i,j} c_{i,j} \chi_i}{E^{\text{total}} f^C}, \\
C &= -\frac{(\lambda_1 + \lambda_2) w_{i,j}}{\ln 2}.
\end{aligned} \tag{25}$$

Once the optimal power allocation $p^*(s)$ is calculated, the optimal channel allocation strategy can be calculated through the following expression:

$$\frac{\partial L(\lambda_1, \lambda_2, \lambda_3, p, a)}{\partial a_{i,j,n}} \Big|_{a_{i,j,n} = a_{i,j,n}^*} = 0. \tag{26}$$

Then, the optimal channel allocation $a_{i,j,n}^*$ can be obtained as follows:

$$a_{i,j,n}^* = \frac{1}{2s_{i,j}P_{i,j,n}^*} + \frac{c_{i,j}}{2s_{i,j}P_{i,j,n}^*f^C} - \frac{(1-w_{i,j})E^{\text{total}}}{2s_{i,j}(1-s_{i,j})P_{i,j,n}^*w_{i,j}} - \frac{e_{i,j}^L}{2s_{i,j}P_{i,j,n}^*} - \frac{\lambda_2 d_{i,j} E^{\text{total}}}{2s_{i,j}(1-s_{i,j})P_{i,j,n}^*w_{i,j}\chi_i^2 E_{i,j}^{\text{max}}} - \frac{\lambda_3 E^{\text{total}}}{2s_{i,j}^2(1-s_{i,j})P_{i,j,n}^*\chi_i^2}, \quad (27)$$

where variable matrices λ_1 , λ_2 , and λ_3 are updated using their corresponding subgradients:

$$\lambda_1(m+1) = \lambda_1(m) + \mu_1 \left(\frac{s_{i,j}d_{i,j}}{\chi_i} - R_{i,j} \right), \quad (28)$$

$$\lambda_2(m+1) = \lambda_2(m) + \mu_2 \left(\frac{s_{i,j}a_{i,j,n}P_{i,j,n}d_{i,j}}{E_{i,j}^{\text{max}}} - R_{i,j} \right), \quad (29)$$

$$\lambda_3(m+1) = \lambda_3(m) + \mu_3 (a_{i,j,n}P_{i,j,n} - p^{\text{max}}), \quad (30)$$

where μ_1 , μ_2 , and μ_3 are the appropriate step sizes of the subgradient algorithm.

Hence, the pseudocode of the algorithm is shown in Algorithm 1. Algorithm 1 details the iterative solution process for resource allocation using the two-tier optimization algorithm.

4.3.2. Upper-Level Problem. Given the power and channel allocation strategies $(p^*(s), a^*(s))$ for fixed s , at this point, the optimal objection becomes

$$P3: \min_s F(p^*(s), a^*(s), s) = \sum_{i=1}^{U_l} \sum_{j=1}^M \left\{ s_{i,j} \left[\begin{aligned} &w_{i,j}^t \left(\frac{d_{i,j}}{w \log(1 + p_{i,j,n}^* h_{i,j,n} / \sigma^2 + I_{i,j,n})} + \frac{c_{i,j}}{f^C} \right) + \\ &s_{i,j} w_{i,j}^e \sum_{n=1}^N a_{i,j,n}^* P_{i,j,n}^* \frac{d_{i,j}}{w \log(1 + p_{i,j,n}^* h_{i,j,n} / \sigma^2 + I_{i,j,n})} \end{aligned} \right] \right\} + \sum_{i=1}^{U_l} \sum_{j=1}^M \{(1-s_{i,j})O_{i,j}^{L*}\}, \quad (31)$$

$$s.t. \text{ C8: } s_{i,j} \in \{0, 1\}, \forall i, j,$$

where the optimization problem is convex concerning $s_{i,j}$, and if the constraint C1 is satisfied, the optimal offloading decision can be found by Q-learning deep reinforcement learning offloading scheme.

4.4. Q-Learning Deep Reinforcement Learning Algorithm. Q-Learning deep reinforcement learning algorithm can ensure the V-UEs make optimal offloading decisions based on the dynamics of the system in terms of V-UE and cloudlet behaviors. The problem is represented as a Markov decision process (MDP), taking into account the number of task requests for all V-UEs, the maximum number of service resources that a V-UE can provide and the size of the number of remaining tasks in the edge cloud, and the distance between the V-UE and edge cloud as state space. The primary objective is to find the optimal actions on how many tasks the V-UE should process locally and how many tasks will be offloaded to each cloudlet such that the V-UE's utility obtained by task execution is minimized while minimizing the energy consumption and latency [36].

4.4.1. MDP-Based Offloading Problem Formulation. The problem of making an offloading decision for the V-UE is expressed as a finite MDP. The MDP can be represented as a tuple $M = (S, A, P, R)$, where S and A represent state and action spaces, $R(s, a)$ is the immediate reward obtained by doing action a at state s [36]. π is a policy that maps from a state s to an action a , that is, $\pi(s) = a$. The primary goal of the V-UE is to find the optimal policy π^* to minimize the V-UE's utility.

Therefore, the state space S is defined as follows:

$$S = \{s = (Q^u, Q^{um}, Q^c, D)\}, \quad (32)$$

where Q^u , Q^{um} , Q^c , and D denote the number of task requests for all V-UEs, the maximum number of service resources that a V-UE can provide, and the size of the number of remaining tasks in the edge cloud and distance state, respectively.

The action space A is defined as

$$A = \{a = (a_0, \dots, a_i, \dots, a_N) | a_i \in \{0, 1, \dots, a_{\text{max}}\}\}, \quad (33)$$

```

(1) Input :  $s_{i,j} \in I_{dd}, O_{i,j}$ 
(2) Output:  $p_{i,j,n}^*, a_{i,j,n}^*$ 
(3) Initialize weights matrix  $p_{i,j,n}, a_{i,j,n}$ 
(4) for  $o = 1: O_{i,j}$ 
(5)  $m \leftarrow 0$ 
(6)   while  $m \leq I_{dd}$  or  $|p_{i,j,n}^*(m+1) - p_{i,j,n}^*(m)| > \varepsilon$ 
(7)     for  $n = 1: N$ 
(8)       for  $i = 1: L$ 
(9)         Calculate  $p_{i,j,n}^*$  according to (25)
(10)        Obtain the optimal channel allocation  $a_{i,j,n}^*$  using (27)
(11)      end for
(12)    end for
(13)  Update the variables  $\lambda_1, \lambda_2$  and  $\lambda_3$  using (28), (29) and (30)
(14) end while
(15) end for

```

ALGORITHM 1: Optimal resource allocation algorithm.

where a_0 represents locally processed task sequence, and a_i ($i = 1, \dots, N$) is the number of tasks to be offloaded to the cloudlet i . $a_{\max}$ is maximum number of tasks that are processed locally or offloaded to the MEC within each decision period. In addition, the total number of tasks per action a must be equal to or less than the number of tasks currently remained in V-UE's queue.

Immediate reward for V-UEs minimizes overhead when making optimal offloading decisions in each system state. Therefore, we define the immediate reward function $R(s, a)$ given an action a at state s as follows:

$$R(s, a) = U(s, a) - C(s, a), \quad (34)$$

where $U(s, a)$ and $C(s, a)$ are immediate utility and cost functions. For the utility, we define the utility function as

$$U(s, a) = \rho(O_{i,j}^L + O_{i,j}^C), \quad (35)$$

where ρ stands for a utility constant. The immediate cost function $C(s, a)$ is expressed as

$$C(s, a) = \eta_1 E(s, a) + \eta_2 T(s, a), \quad (36)$$

where η_1 and η_2 are a constant. $E(s, a)$ is energy consumption, and $T(s, a)$ is delay, that is,

$$\begin{aligned} E(s, a) &= a_0 e_{i,j}^L(s, a) + a_i e_{i,j}^C(s, a), \\ T(s, a) &= a_0 t_{i,j}^L(s, a) + a_i t_{i,j}^C(s, a). \end{aligned} \quad (37)$$

4.4.2. DQN-Based Offloading Decision Algorithm. To find out the solution based on MDP offloading problem, we propose an online learning scheme based on the model-free deep RL algorithm, called deep Q-network (DQN) [37]. In the DQN-based learning scheme, the V-UE acts as an agent that interacts with the mobile edge cloud environment, receives sensory data, and performs a valid action. Then, the agent selects an offloading action a_t for state s_t at time slot t in a fashion that minimizes the future discounted reward

over a long run. A deep neural network, called a deep Q-network, is used to approximate the optimal action-value function [37]:

$$Q^*(s, a) = -\max_{\pi} E \left[r_t + \sum_{k=1}^{\infty} \gamma^k r_{t+k} \mid s_t = s, a_t = a, \pi \right]. \quad (38)$$

Taking action a at the state s at the time slot t in the offloading policy π discounted through γ to minimize the sum of reward r_t . $\gamma \in (0, 1)$ is a discount factor to balance immediate reward and future reward. $Q^*(s, a)$ denotes the expected total reward in state s and action a . Note that $E[\cdot]$ denotes the expectation function.

The Q-network can be regarded as a neural network approximator, which is used to estimate the action-value function. θ is a weight value of $Q(s, a; \theta)$. At each decision period, the V-UE first takes the state vector $S = (Q^u, Q^{um}, Q^c, D)$ as the input of the Q-network and obtains the Q-values $Q(s, \cdot)$ for all possible action a as outputs. Then, the V-UE selects the action according to the ε -greedy exploration method.

Furthermore, the Q-network can be trained by iteratively adjusting the weights θ to minimize the loss function sequence, where the loss function at time slot t is

$$L_t(\theta_t) = -E \left[\left(r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta_{t-1}) - Q(s_t, a_t, \theta_t) \right)^2 \right], \quad (39)$$

where θ_t represents the network parameters in iteration t , and the previous network parameters θ_{t-1} are used to compute the target $(r_t + \gamma \max_{a'} Q(s_{t+1}, a'))$.

In other words, given a transition s_t, a_t, r_t, s_{t+1} , the weights θ updates the Q-network in a way that minimizes the squared error loss between the current predicted Q-value of $Q(s_t, a_t)$ and the target Q-value of $(r_t + \gamma \max_{a'} Q(s_{t+1}, a'))$.

In addition, in the DQN algorithm, the empirical replay technique is used as the training method to improve the Q-network instability due to the use of a non-linear approximation function. More specifically, the V-UE's experiences set $e_t = s_t, a_t, r_t, s_{t+1}$ are stored in a

```

(1) : Initialize:  $t = 1$ , weight value  $\theta$   $Q = 0$ ,  $max\_convergence\_time$ 
(2) : for  $t = 1, 2, \dots$ , do
(3) : while  $t < max\_convergence\_time + 1$  do
(4) : Observer current state  $s_t$ 
(5) : Select action  $a_t$  according to  $\epsilon - gree dy$  policy
(6) : Offload the tasks according to action  $a_t$  and observe reward  $r_t = R(s_t, a_t)$  and next states  $s_{t+1}$ 
(7) : Store experience  $s_t, a_t, r_t, s_{t+1}$  into replay memory  $\Omega$ 
(8) : Randomly select a set of transitions  $s, a, r, s'$  from replay memory  $\Omega$ 
(9) : Train the Q-network based on selected transitions using  $(r_t + \gamma \max_{a'} Q(s_{t+1}, a'; \theta_{t-1}) - Q(s_t, a_t; \theta_t))^2$  as loss function
(10) : end for
(11)  $t = t + 1$ :
(12) : end while
(13) : until the algorithm converges

```

ALGORITHM 2: DQN-based offloading decision algorithm.

TABLE 2: Simulation parameters.

Parameters	Values
Maximum transmit power $p^{\max}$	23 dBm
Bandwidth W	20 MHz
The CPU frequency of MEC servers f^C	4 GHz/cycle
The total number of CPU cycles $c_{i,j}$	[0.1,1] GHz
The minimum CPU frequency of V-UEs $f_l^{\min}$	0.2 GHz/cycle
The maximum CPU frequency of V-UEs $f_l^{\max}$	1 GHz/cycle
Noise power σ^2	-113 dBm
The total energy consumption E^{total}	100 J
The maximum latency $D_{i,j}^{\max}$	0.5 s
The maximum number of iterations $I_{d,d}$	600
Step size μ_1	2×10^{-18}
Step size μ_2	10^{-5}
Step size μ_3	10^{-10}
Power allocation precision ϵ	10^{-5}
V-UE's queue size $ Q^u $	10
Cloudlets' queue size $ Q_i^c $	4
Number of distance states H	4
Decision period T	60s
Input data size $d_{i,j}$	[300,1600]KB

replay memory $\Omega = \{e_{t-\psi}, \dots, e_t\}$, where ψ is the replay memory capacity. While training the Q-network, a random mini-batch is taken from the replay memory instead of the most recent transition e_t .

The pseudocode of Algorithm 2, a detailed DQN-based offloading decision algorithm is presented. The lines (2)-(4) are recursive, presenting the V-UE's action on making an offloading decision at the beginning of each decision period. Then, the Q-values are estimated by using Q-network. The lines (5)-(7) are to train the Q-network by using the experience replay method.

5. Simulation Results

In this section, the MATLAB simulation platform verifies that the proposed resource allocation mechanism is efficient in the MEC-based vehicle network. We follow the highway parameter settings described in the 3GPP TR 36.885 and build the system model as specified in the MEC white paper.

Among them, the RSU is deployed on the highway side with a communication radius of 250 meters. In addition, the MEC server is deployed on the RSU to provide services to the vehicles. The specific simulation parameters are shown in Table 2.

In a multicell scenario, each RSU shares the same spectrum to improve spectral efficiency. However, V-UEs are connected to the RSU by OFDMA technology, and they suffer from the interference of neighboring RSUs. For example, assuming that each RSU has ten channels, each RSU has 25 V-UEs, and there are 5 RSUs.

Figure 2 shows the bi-level optimization approach compared with the F-W algorithm [16], the exhaustive resource allocation algorithm (ERAA) [38], and the nonresource allocation (no power allocation and channel allocation) in preference analysis. Although the iterative times of the ERAA are high, the optimal allocation value is found in all the cases, so it converges to almost optimal value in each iteration. As seen from Figure 2, the latency without resource allocation is the highest, followed by the

F-W algorithm, and the bi-level optimization approach has the lowest latency. This fully shows that resource allocation can reduce the latency of the system. Furthermore, compared with the F-W algorithm, the proposed bi-level optimization approach can find better power allocation and channel allocation values so that the latency is lower.

Figure 3 shows that the total energy consumption increases with the increase in the number of users. When the number of users is less than 10, there is a slight difference in the total energy consumption among different algorithms due to the small number of users. When the number of users is more than 10, the total energy consumption using F-W algorithm increases gradually as the number of users increases. The total energy consumption without resource allocation keeps growing at high speed all the time. It can be seen that resource allocation is significant for reducing energy consumption. ERAA algorithm can converge to the optimal value through multiple iterations. The performance of the bi-level optimization approach we adopted is very close to that of the ERAA algorithm in terms of reducing energy consumption. Still, the complexity of ERAA is much higher than that of the bi-level optimization approach. Because the energy consumption is related to the transmission power, and the advantage of the bi-level optimization approach compared with the F-W algorithm is that it can find a better power allocation value, so the performance of the bi-level optimization approach is better than that of the F-W algorithm in terms of reducing the total energy consumption.

Figure 4 shows the relationship between the number of V-UEs and the total overhead. The total overhead is the weighted sum of the latency and energy consumption. It can be seen from the figure that when the number of users reaches 20, the ERAA algorithm becomes stable and no longer increases. In other words, the ERAA algorithm eventually converges to the optimal value. But it needs to be iterated many times, which makes the complexity very high compared with the proposed bi-level optimization approach. It can be seen from Figure 4 that the total overhead is the largest when the resources do not allocate. Compared with the F-W algorithm, the bi-level approach we used better affects the channel and power allocation. In summary, considering the problem of minimum complexity and overhead, the bi-level optimization algorithm can provide better performance.

Figure 5 shows the effect of different weighting factors on V-UEs' sum of energy consumption and latency. The definition of the weighting factor considers the real-time residual energy awareness of V-UEs. Compared with the subjective weighting, we redefine the weighting factor $w_1 = w * rE$, which can achieve a better compromise, and save more energy than $w = 0.8$ and $w = 0.5$ from Figure 6. Because energy consumption is proportional to the weighting factor and inversely proportional to latency, the weighting factor $w_1 = w * rE$ can decrease latency comparing with $w = 0.2$ from Figure 7. It shows that the proposed algorithm can obtain lower cost than other weighting factors.

FIGURE 2: The total latency with different V-UEs.

FIGURE 3: The total energy consumption with different V-UEs.

Figure 8 shows the learning curves, which present the total reward obtained by the offloading policy learned in each training episode. Note that an episode includes 5000 iterations (i.e., the number of decision periods), in each of which the V-UE selects an offloading action for each state according to the current policy learned by the proposed algorithm. As shown in Figure 8, the total reward obtained in each episode decreases steadily when the learning time, that is, the number of episodes, increases from 1 to 200. Then, all learning curves become stable and no longer decrease when the episode number is higher than 200. This result indicates that the proposed DQN-based learning algorithm converges after 200 learning episodes. It can also be seen from Figure 8 that when the discount factor $\gamma = 0.4$, the convergence speed is the fastest, and when $\gamma = 0.8$, the overhead is the smallest. Because minimal overhead is the primary consideration, $\gamma = 0.8$ is chosen.

FIGURE 4: The total overhead with different V-UEs.

FIGURE 5: The total overhead under different weighting factors.

FIGURE 6: The total latency under various weighting factors.

FIGURE 7: The total energy consumption under different weighting factors.

FIGURE 8: Learning curves.

6. Conclusions

In this work, we proposed a resource allocation scheme for offloading in real-time energy-aware vehicular networks with MEC to minimize the total overhead, which balanced latency and energy consumption. We jointly considered the offloading resource allocation and redefined the weighting factor based on the real-time residual energy aware of the V-UEs. In the multicell networks, a bi-level optimization approach was proposed to approximate the MINLP problem to a convex problem. By taking partial derivatives, we found a better resource allocation scheme, which simplified the

problem and reduced the computational complexity. The scheme in this article reduced the total overhead by nearly 40% to the compared algorithm. Also, the proposed offloading scheme based on deep reinforcement learning (DRL) enabled the users to make the optimal offloading decisions by considering uncertainties of resource availabilities of users and cloudlets.

Appendix

A. Proof of Nonconvexity of Objective Function

The authors use the proof of absurdity to prove that the objective function in formula (20) is nonconvex. The authors suppose that the objective function in formula (20) is convex. The affine transformation is composed of a linear addition and a translation. It does not change the concavity and convexity of functions. Therefore, according to affine transformation and convexity preserving operations, the objective function in formula (20) can be equated to

$$\min_{s,p,a} F(s, p, a) = \frac{s_{i,j}}{\log_2(1 + p_{i,j,n})} + \frac{s_{i,j}a_{i,j,n}p_{i,j,n}}{\log_2(1 + p_{i,j,n})}, \quad (\text{A.1})$$

and formula (A.2) is obtained through a series of affine transformations and convexity preserving operations according to formula (20). Formula (20) and formula (A.2) have the same concavity and convexity.

According to the theorem, if the function $f: \Omega \rightarrow \mathbb{R}, f \in C^2$ is defined on the open convex set $\Omega \subset \mathbb{R}^n$, f is defined on the convex function Ω , if and only if for any $x \in \Omega$, f is semidefinite on the Hessian matrix $F(x)$ at point x . Then, the authors calculate the Hessian matrix and determine whether it is semidefinite:

$$\frac{\partial^2 F}{\partial s_{i,j}^2} = 0, \quad (\text{A.2})$$

$$\frac{\partial^2 F}{\partial a_{i,j,n}^2} = 0, \quad (\text{A.3})$$

$$\frac{\partial^2 F}{\partial s_{i,j} \partial a_{i,j,n}} = \frac{\partial^2 F}{\partial a_{i,j,n} \partial s_{i,j}} = \frac{p_{i,j,n}}{\log_2(1 + p_{i,j,n})}, \quad (\text{A.4})$$

$$\frac{\partial^2 F}{\partial s_{i,j} \partial p_{i,j,n}} = \frac{\partial^2 F}{\partial p_{i,j,n} \partial s_{i,j}} = \frac{a_{i,j,n} \log_2(1 + p_{i,j,n}) - 1 + a_{i,j,n} p_{i,j,n} / p_{i,j,n} \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \quad (\text{A.5})$$

$$\frac{\partial^2 F}{\partial a_{i,j,n} \partial p_{i,j,n}} = \frac{\partial^2 F}{\partial p_{i,j,n} \partial a_{i,j,n}} = \frac{s_{i,j,n} \log_2(1 + p_{i,j,n}) - s_{i,j,n} / \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \quad (\text{A.6})$$

$$\begin{aligned} \frac{\partial^2 F}{\partial p_{i,j,n}^2} &= \frac{(s_{i,j,n} a_{i,j,n} / p_{i,j,n} \ln 2 + s_{i,j,n} / \ln 2 p_{i,j,n}^2) (\log(1 + p_{i,j,n}))^2}{(\log(1 + p_{i,j,n}))^4} \\ &\quad - \frac{(s_{i,j,n} a_{i,j,n} \log(1 + p_{i,j,n}) - s_{i,j,n} (1 + p_{i,j,n}) / p_{i,j,n} \ln 2) 2 \log(1 + p_{i,j,n}) / p_{i,j,n} \ln 2}{(\log(1 + p_{i,j,n}))^4}. \end{aligned} \quad (\text{A.7})$$

Let $F_{s,a} = p_{i,j,n} / \log_2(1 + p_{i,j,n})$,

$$\begin{aligned} F_{a,s} &= \frac{p_{i,j,n}}{\log_2(1 + p_{i,j,n})}, \\ F_{s,p} &= \frac{a_{i,j,n} \log_2(1 + p_{i,j,n}) - 1 + a_{i,j,n} p_{i,j,n} / p_{i,j,n} \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \\ F_{p,s} &= \frac{a_{i,j,n} \log_2(1 + p_{i,j,n}) - 1 + a_{i,j,n} p_{i,j,n} / p_{i,j,n} \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \\ F_{a,p} &= \frac{s_{i,j,n} \log_2(1 + p_{i,j,n}) - s_{i,j,n} / \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \\ F_{p,a} &= \frac{s_{i,j,n} \log_2(1 + p_{i,j,n}) - s_{i,j,n} / \ln 2}{(\log_2(1 + p_{i,j,n}))^2}, \\ F_{p,p} &= \frac{(s_{i,j,n} a_{i,j,n} / p_{i,j,n} \ln 2 + s_{i,j,n} / \ln 2 p_{i,j,n}^2) (\log(1 + p_{i,j,n}))^2 - (s_{i,j,n} a_{i,j,n} \log(1 + p_{i,j,n}) - s_{i,j,n} (1 + p_{i,j,n}) / p_{i,j,n} \ln 2) 2 \log(1 + p_{i,j,n}) / p_{i,j,n} \ln 2}{(\log(1 + p_{i,j,n}))^4}. \end{aligned} \quad (\text{A.8})$$

Then, we can obtain the Hessian matrix:

$$H = \begin{pmatrix} 0 & F_{s,a} & F_{s,p} \\ F_{a,s} & 0 & F_{a,p} \\ F_{p,s} & F_{p,a} & F_{p,p} \end{pmatrix}. \quad (\text{A.9})$$

Due to $F_{s,a} = p_{i,j,n} / \log_2(1 + p_{i,j,n}) > 0$ and $F_{a,s} = p_{i,j,n} / \log_2(1 + p_{i,j,n}) > 0$, $\begin{vmatrix} 0 & F_{s,a} \\ F_{a,s} & 0 \end{vmatrix} = -(F_{s,a})^2 < 0$. Therefore, H is not a positive definite matrix. This shows that the objective function in formula (20) is nonconvex.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

X. L. and X. B. wrote the original draft; X. L. and H. Z. reviewed and edited the manuscript; X. B. and Y. C. helped with software; X. L., Y. C., and H. Z. validated the study; Y. C. and S. X. contributed to visualization; H. Z. helped with the funding acquisition; H. Z. involved in project administration; H. Z. provided the resources; S. X. supervised the study and carried out the formal analysis. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This study was funded by the National Natural Science Foundation of China (61801065), the Program for Changjiang Scholars and Innovative Research Team in University (IRT16R72), Chongqing Innovation and Entrepreneurship Project for Returned Overseas Chinese Scholars (cx2020059), and the General Project on Foundation and Cutting-Edge Research Plan of Chongqing (cstc2018jcyjAX0463).

References

- [1] S. E. Shladover, "Opportunities and challenges in cooperative road vehicle automation," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 2, pp. 216–224, 2021.
- [2] H. Zhou, W. Xu, J. Chen, and W. Wang, "Evolutionary V2X technologies toward the Internet of vehicles: challenges and opportunities," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 308–323, 2020.
- [3] H. Zhang and X. Lu, "Vehicle communication network in intelligent transportation system based on Internet of Things," *Computer Communications*, vol. 160, pp. 799–806, 2020.
- [4] Y. Qi, L. Tian, Y. Zhou, and J. Yuan, "Mobile edge computing-assisted admission control in vehicular networks: the convergence of communication and computation," *IEEE Vehicular Technology Magazine*, vol. 14, no. 1, pp. 37–44, 2019.
- [5] F. Guo, F. R. Yu, H. Zhang, X. Li, H. Ji, and V. C. M. Leung, "Enabling massive IoT toward 6G: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 11891–11915, 2021.
- [6] Z. Lv, L. Qiao, and I. You, "6G-Enabled network in box for Internet of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 8, pp. 5275–5282, 2021.
- [7] J. Zhu, J. Wang, Y. Huang, F. Fang, K. Navaie, and Z. Ding, "Resource allocation for hybrid NOMA MEC offloading," *IEEE Transactions on Wireless Communications*, vol. 19, no. 7, pp. 4964–4977, 2020.
- [8] L.-M. Ang, K. P. Seng, G. K. Ijamaru, and A. M. Zungeru, "Deployment of IoV for smart cities: applications, architecture, and challenges," *IEEE Access*, vol. 7, pp. 6473–6492, 2019.
- [9] L. Liu, S. Lu, R. Zhong et al., "Computing systems for autonomous driving: state of the art and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6469–6486, 2021.
- [10] K. Zhang, Y. Mao, S. Leng et al., "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, pp. 5896–5907, 2016.
- [11] F. Jauro, H. Chiroma, A. Y. Gital, M. Almutairi, S. i. M. Abdulhamid, and J. H. Abawajy, "Deep learning architectures in emerging cloud computing architectures: recent development, challenges and next research trend," *Applied Soft Computing*, vol. 96, p. 106582, 2020.
- [12] S. Guo, B. Xiao, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *Proceedings of the IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.
- [13] F. Cicirelli, A. Guerrieri, G. Spezzano et al., "Edge computing and social Internet of things for large-scale smart environments development," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2557–2571, 2018.
- [14] J. Pan and J. McElhannon, "Future edge cloud and edge computing for Internet of things applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 439–449, 2018.
- [15] M. Liwang, J. Wang, Z. Gao, X. Du, and M. Guizani, "Game theory based opportunistic computation offloading in cloud-enabled IoV," *IEEE Access*, vol. 7, pp. 32551–32561, 2019.
- [16] J. Zhang, X. Hu, Z. Ning et al., "Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633–2645, 2018.
- [17] R. Zheng, H. Wang, M. De Mari, M. Cui, X. Chu, and T. Q. S. Quek, "Dynamic computation offloading in ultra-dense networks based on mean field games," *IEEE Transactions on Wireless Communications*, vol. 20, no. 10, pp. 6551–6565, 2021.
- [18] H. Ma, S. Li, E. Zhang, Z. Lv, J. Hu, and X. Wei, "Cooperative autonomous driving oriented MEC-aided 5G-V2X: prototype system design, field tests and AI-based optimization tools," *IEEE Access*, vol. 8, pp. 54288–54302, 2020.
- [19] Y. Cao, X. Zhang, B. Zhou, X. Duan, D. Tian, and X. Dai, "MEC intelligence driven electro-mobility management for battery switch service," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4016–4029, 2021.
- [20] Y. Yu, X. Tang, J. Wu, B. Kim, T. Song, and Z. Han, "Multi-leader-follower game for MEC-assisted fusion-based vehicle on-road analysis," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11200–11212, 2019.
- [21] S.-R. Yang, Y.-J. Su, Y.-Y. Chang, and H.-N. Hung, "Short-term traffic prediction for edge computing-enhanced autonomous and connected cars," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3140–3153, 2019.

- [22] Y. Ling, X. Chu, Z. Lu, L. Wang, and X. Wen, "PCM: a positioning calibration mechanism for connected autonomous vehicles," *IEEE Access*, vol. 8, pp. 95046–95056, 2020.
- [23] Y. Wang, P. Lang, D. Tian et al., "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4987–4996, 2020.
- [24] Y. Hou, C. Wang, M. Zhu, X. Xu, X. Tao, and X. Wu, "Joint allocation of wireless resource and computing capability in MEC-enabled vehicular network," *China Communications*, vol. 18, no. 6, pp. 64–76, 2021.
- [25] Z. Ning, K. Zhang, X. Wang et al., "Intelligent edge computing in Internet of vehicles: a joint computation offloading and caching solution," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2212–2225, 2021.
- [26] 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.
- [27] H. Zhang, Z. Wang, and K. Liu, "V2X offloading and resource allocation in SDN-assisted MEC-based vehicular networks," *China Communications*, vol. 17, no. 5, pp. 266–283, 2020.
- [28] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, 2013.
- [29] M. Shojafar, N. Cordeschi, and E. Baccarelli, "Energy-efficient adaptive resource management for real-time vehicular cloud services," *IEEE Transactions on Cloud Computing*, vol. 7, no. 1, pp. 196–209, 2019.
- [30] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile edge computing: partial computation offloading using dynamic voltage scaling," *IEEE Transactions on Communications*, vol. 64, no. 10, pp. 4268–4282, 2016.
- [31] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432–7445, 2017.
- [32] 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.
- [33] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, 2015.
- [34] S.-T. Hong and H. Kim, "QoE-aware computation offloading to capture energy-latency-pricing tradeoff in mobile clouds," *IEEE Transactions on Mobile Computing*, vol. 18, no. 9, pp. 2174–2189, 2019.
- [35] K. Cheng, Y. Teng, W. Sun, A. Liu, and X. Wang, "Energy-efficient joint offloading and wireless resource allocation strategy in multi-MEC server systems," in *Proceedings of the 2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.
- [36] D. Van Le and C. Tham, "A deep reinforcement learning based offloading scheme in ad-hoc mobile clouds," in *Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 760–765, Honolulu, HI, USA, July 2018.
- [37] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [38] Z. Liu, Y. Liu, and F. Liu, "Fast antenna selection algorithm for full-duplex MIMO communication system," in *Proceedings of the 2015 IEEE/CIC International Conference on Communications (ICCC)*, pp. 1–5, Shenzhen, China, November 2015.