Reinforcement Learning for Joint Optimization of Communication and Computation in Vehicular Networks

Yaping Cui, Lijuan Du, Honggang Wang, *Fellow, IEEE*, Dapeng Wu, *Senior Member, IEEE*, and Ruyan Wang

***Abstract*—Ultra reliability and low latency communications (URLLC) are considered as one of the most important use cases for computation tasks in the Internet of Vehicles (IoV) edge computing networks. To meet the task requirements in IoV networks, communication and computing resources are allocated in an effective and efficient way. Thus, this paper proposes a multi-objective reinforcement learning strategy, called intelligent communication and computation resource allocation (ICCRA), which combines communication and computing resource allocation to reduce the total system cost consisting of latency and reliability. Specifically, this strategy can be decomposed into three algorithms. The algorithm called joint computation offloading and collaboration is a general framework of the strategy, it first uses the K-nearest neighbor method to select offloading layers for computation tasks, such as cloud computing layer, mobile edge computing layer, and local computing layer. Then, when selecting the local computing layer to perform the task, the algorithm called collaborative vehicle selection is used to find the target vehicle to execute collaborative computing. The allocation of communication and computing resources is defined as two independent objectives, the algorithm called multi-objective resource allocation uses the reinforcement learning to achieve the optimal solution, at the mobile edge computing layer. Simulation results show that compared with local computing, edge computing, and random computing strategies, the proposed strategy reduces the total system cost effectively. For instance, compared with edge computing strategy, the total system cost of the proposed strategy can be reduced about 50% on average with the different number of vehicle tasks.**

Manuscript received December 6, 2020; revised May 14, 2021 and September 7, 2021; accepted October 24, 2021. Date of publication November 4, 2021; date of current version December 17, 2021. This work was supported in part by the Natural Science Foundation of China under Grants 61801065, 61771082, 61871062, 61901070, and U20A20157, in part by the Science and Technology Research Program of Chongqing Municipal Education Commission under Grants KJQN202000603 and KJQN201900611, in part by the Natural Science Foundation of Chongqing under Grant cstc2020jcyj-zdxmX0024, and in part by the University Innovation Research Group of Chongqing under Grant CXQT20017. The review of this article was coordinated by Prof. Rose Qingyang Hu. *(Corresponding author: Yaping Cui.)*

Yaping Cui is with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China, and also with the School of Aeronautics and Astronautics, University of Electronic Science and Technology of China, Chengdu 611731, China (e-mail: cuiyp@cqupt.edu.cn).

Lijuan Du, Dapeng Wu, and Ruyan Wang are with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: dulijuan0706@163.com; wudp@cqupt.edu.cn; wangry@cqupt.edu.cn).

Honggang Wang is with the Department of Electrical and Computer Engineering, University of Massachusetts Dartmouth, Boston, MA 02747 USA (e-mail: hwang1@umassd.edu).

Digital Object Identifier 10.1109/TVT.2021.3125109

***Index Terms*—Internet of Vehicles, mobile edge computing, collaborative computing, ultra-reliable and low-latency communications, multi-objective reinforcement learning.**

I. Introduction

INTERNET of Vehicles (IoV) are expected to support many advanced applications for road safety, traffic efficiency, passenger infotainment, and automated driving [1]. It means that diversified services need to be provided to meet the various requirements of different applications, especially for ultra-reliable and low-latency communications (URLLC). However, due to the explosive growth of vehicles, volatile network topologies, and limited computation capacity, the reliability and latency requirements cannot be met in IoV efficiently [2]. To overcome such challenges, one potential approach is the use of mobile edge computing (MEC) [3], [4]. In MEC networks, the computation capacity is extended at the edge of wireless network, thus the computation-intensive tasks can be migrated to the MEC server. Therefore, the transmission reliability and latency would be shortened to address the requirements of diversified services in IoV networks [5]. Moreover, compared with cloud computing, the flexible and scalable deployments of MEC server lead to a cost-effective way to guarantee the heterogeneous types of services in IoV networks [6], [7]. Moreover, machine learning has proved to be an effective tool for computer vision, natural language processing, and robotics [8], and can be used for channel prediction, security architecture, and device-to-device communications in wireless networks [9]–[11]. Thus, machine learning can be applied to vehicular networks as an important component of development of the intelligent transportation systems.

In general, reliability and latency are two important performance metrics for evaluating the URLLC vehicular networks. However, these two metrics are usually considered as the separate objectives. In [12], a power minimization problem was considered under reliability and queuing latency constraints, then a Lyapunov framework was proposed to solve this problem. This paper also indicated that queuing latency is a function of the reliability requirements. Similarly, in [13], a resource allocation strategy was proposed to get a tradeoff between the average power consumption and reliability. In [14], a deep reinforcement learning based approach was proposed to allocate the communication and computation adaptively to reduce the offloading

delay. Further, a knowledge driven deep reinforcement learning approach was proposed to minimize the execution delay. In [15], a multi-user non-cooperative computation offloading game model was proposed to adjust offloading probability of each vehicle, and a distributed best response algorithm based on the computation offloading game model was constructed to minimize the latency of task execution. In [16], the reliability is defined as a function of instantaneous rate, and a risk-sensitive approach was proposed to guarantee the reliability. Further, a risk-sensitive approach was leveraged to minimize the end-to-end task offloading delay in vehicular edge computing networks [17].

Some literatures, such as reference [18], provide the survey on resource allocation of the computing and communication in heterogeneous network to maintain the extreme high quality-of-service (QoS). Especially, there are some works on mobile edge computing for URLLC vehicular networks [19]–[24]. For instance, in [19], the URLLC requirements of vehicular communications were transferred into optimization constraints, then a separate resource block and power allocation algorithm was proposed to maximize the user sum rate under this constraint. In [20], the authors proposed a utility-delay approach to maximize network utility subject to URLLC constraints. However, these works were only verified in slowly varying channel, they may be not suitable for time varying network topologies. In [21], Lyapunov optimization and matching theory were used to associate user-MEC server so as to guarantee the reliable and latency performance. In [22], a meta-heuristic approach was developed to solve the mixed integer programming of the URLLC. In [23], short packet transmission was adopted and taken into account for resource allocation algorithm design to reduce the latency and improve the reliability. In [24], a frame design algorithm and a semi-persistent scheduling algorithm were proposed to achieve optimal frame design and resource allocation scheme, which can greatly satisfy the URLLC requirements of vehicular networks.

For vehicular edge computing networks, an effective and promising resource allocation is a unified strategy, which schedules all available resources (i.e., cloud computation resource, mobile edge computation resources, local computation resources, and communication resources) to meet the various task requirements of different applications [25]. The above literatures mainly focused on computation offloading problem between users and MEC servers in vehicular edge computing networks. Only a few literatures focused on collaborative computing among MEC servers or users in such networks. However, collaborative computing among MEC servers will enhance the computation capacity of vehicles efficiently, which will help the vehicular networks to increase reliability and reduce latency significantly.

The traditional numerical optimization methods to solve the NP-hard combinatorial optimization problem include max-min Algorithm [26], iterative Algorithm [27], etc. Nowadays, due to the dynamics of the wireless channel and the network topology of vehicular networks, researchers begin to pay more attention to reinforcement learning (RL). RL can utilize historical data to address decision making under uncertainties [28]. When a new service request arrives, RL can adaptively allocate resources to maximize the long-term benefits of the vehicular networks. There are many literatures investigating the resource allocation strategies based on RL in vehicular networks. In [29], the cluster-enabled cooperative scheduling based on RL algorithm was proposed to achieve efficient and reliable communication by maximizing the capacity. In [30], the authors investigated a more efficient reinforcement learning-based resource allocation algorithm to adaptively change time division duplexing (TDD) configuration in 5 G vehicular network. In [31], to reduce the probability of delay in processing requests, the authors proposed a novel adaptive cloud resource allocation model based on Semi-Markov Decision Process and RL algorithm in the vehicular cloud system. In [32], by utilizing parked vehicles, the resource allocation algorithm that considers both the short-term and the long-term resource allocations obtained from the proposed heuristic algorithm and from the RL algorithm, respectively.

Based on aforementioned background, a multi-objective reinforcement learning strategy, namely, intelligent communication and computation resource allocation (ICCRA), is proposed to reduce the total cost of the system consisting of latency and reliability. Our main contributions are summarizes as follows:

- Our proposed ICCRA strategy considers joint co-layer and cross-layer computation and communication resources to guarantee the various task requirements.
- We decompose the proposed strategy into three algorithms. Specifically, Algorithm 1 is a general framework to select offloading layer for computation tasks. When selecting the local computing, Algorithm 2 is used to find the target vehicle in local computing layer for collaborative computing. Algorithm 3 is used to allocate the computation and communication resources in MEC layer.
- We use multi-objective reinforcement learning for task offloading and collaboration to achieve the tradeoff between reliability and latency while considering reliability and latency constraints.

The rest of this paper is organized as follows. In Section II, the system models are presented, including network model, task model, and computation model. In Section III, the joint communication and computation resource allocation is formulated as an optimization problem. Then, a multi-objective reinforcement learning strategy, called ICCRA strategy, is proposed to guarantee the reliability and latency of diversified services in IoV networks in Section IV. Numerical results are illustrated in Section V. Section VI concludes this paper and suggests some future work.

II. System Models

In this section, the system models are described, including network model, task model, and computation model.

A. Network Model

A IoV edge computing network, consisting of cloud computing layer, MEC layer, and local computing layer, is considered, as shown in Fig. 1. Cloud computing layer, with its powerful computation capacity and remote deployment, can be

TABLE I
KEY NOTATIONS AND DEFINITIONS

Notation	Definition
ψ_n	Computation task
K_n	The data size of task ψ_n
X_n	The output size of task ψ_n
D_n	The number of CPU cycles needed to accomplish each bit of the task ψ_n
τ_n	The latency constraint of task ψ_n
$\mathbf{A}$	The offloading decision vector
$\mathbf{B}$	The bandwidth allocation vector
$\mathbf{F}$	The computing resource allocation vector in MEC layer
W	The bandwidth constraint
R	Total computing resources in MEC layer
T_n^L, T_n^M, T_n^C	The total latency for local computing layer, MEC layer and cloud computing layer
z^l, z^m, z^c	The total computation capacity of vehicle, MEC server and cloud server
C_n^{del}	The latency cost of vehicle n
C_n^{re}	The reliability cost of vehicle n
I_d, I_r	The weight of latency cost and reliability cost
Q_d, Q_r	The Q matrix of latency cost O_1 and reliability cost O_2

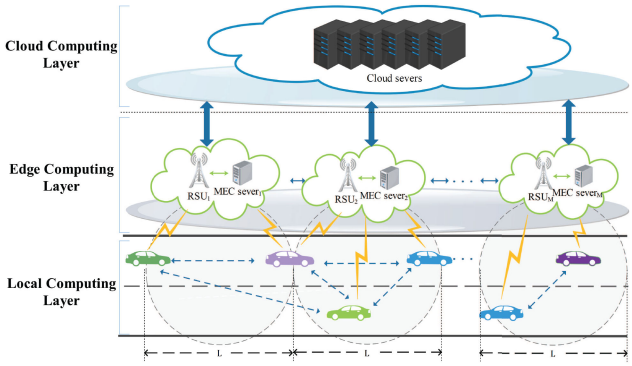


Fig. 1. Network model.

used to process the large-scale, long-term data. For MEC layer, which has moderate computation capacity and deploys close to networks, can be used to assist the vehicles. While in local computing layer, which is comprised of the vehicle equipment (VE), can be used to process the small-scale, latency-critical data with the help of MEC server.

We assume that each vehicle has a computation task, which can be executed in any layer. Moreover, collaborative computing is considered in both MEC layer and local computing layer to improve the task offloading performance. Here, we define the all road side units (RSUs) and vehicles as the set of $\mathcal{M} = \{1, 2, \ldots, M\}$ and $\mathcal{N} = \{1, 2, \ldots, N\}$, respectively. The MEC server is deployed with each RSU to provide computation capacity. For simplicity, we assume that the vehicles are covered by the nearest RSU through vehicle-to-infrastructure (V2I) link, and vehicle-to-vehicle (V2V) link is used to execute collaborative computing between vehicles.

B. Task Model

The key notations used in this paper are listed in Table I. We assume that there is a computation task of vehicle n, denoted as $\psi_n = (K_n, X_n, D_n, \tau_n)$, where K_n is the input data size of task ψ_n, X_n is the output data size of task ψ_n after calculation, D_n is the total number of CPU cycles required to accomplish each bit of the task ψ_n and partial offloading is considered in co-layer [22], [25], τ_n is defined as the maximum tolerable latency of task ψ_n. And the values of the above four parameters are different for each task. Furthermore, vehicles can offload the whole of their computation tasks to any layer in IoV edge computing network. In order to represent the offloading decision, we introduce a offloading decision vector $\mathbf{A} = [\alpha_1, \alpha_2, \ldots, \alpha_N]$, where $\alpha_n = \{\alpha_n^l, \alpha_n^m, \alpha_n^c\}$, $\alpha_n^l, \alpha_n^m, \alpha_n^c \in (0, 1)$, and $\alpha_n^l + \alpha_n^m + \alpha_n^c = 1$. Here, α_n is the offloading decision vector of vehicle n, which contains three binary indicator variables: α_n^l, α_n^m and α_n^c. $\alpha_n^l = 1$ indicates that vehicle n can offload its task ψ_n to the local computing layer, otherwise $\alpha_n^l = 0$. Similarly, if the MEC layer or the cloud computing layer is selected as the collaborative computing layer to accomplish task ψ_n, correspondingly $\alpha_n^m = 1$ or $\alpha_n^c = 1$. For α_n^l, α_n^m and α_n^c, only one of them can be set to 1. That is, the computation task ψ_n of vehicle n can only be offloaded to one layer for computing.

Similarly, we define the bandwidth allocation vector as $\mathbf{B} = [\beta_1, \beta_2, \ldots, \beta_N]$, where $0 < \beta_n \leq 1$, and $\sum_{n=1}^{N} \beta_n \leq 1$. For each vehicle, the allocated transmission rate will affect the reliability of IoV networks, thus the appropriate bandwidth allocation will improve the network reliability with the total bandwidth constraint.

C. Computation Model

For the computation model, the local computing model, MEC model, and cloud computing model are described, respectively.

1) Local Computing Model: When the computation task ψ_n is performed in local computing layer, we define the total latency for local computing layer as T_n^L. Here, we define the total computation capacity of vehicle CPUs as z^l, which is the number of CPU cycles per second.

If vehicle n has available computing resources to execute its computation task ψ_n locally, we denote z_n^l as the computation capacity allocated by vehicle n to compute the task ψ_n. Thus,

the local computation latency of executing task ψ_n on vehicle n is

$$T_n^{comp} = \frac{D_n K_n}{z_n^l}, \tag{1}$$

Vehicle n would offload the computation task ψ_n to the assistant vehicle i that has available computing resources if there is no enough computing resources in vehicle n. The collaborative execution latency of executing task ψ_n on vehicle i includes three parts: the uplink transmission latency, the collaborative computation latency of executing task ψ_n on assistant vehicle i, and the downlink transmission latency.

Considering the collaborative computing between vehicles, the uplink transmission latency can be expressed as

$$T_{n,i}^{UL} = \frac{K_n}{r_{n,i}^{UL}}, \tag{2}$$

where $r_{n,i}^{UL}$ is the uplink transmission rate from vehicle n to vehicle i, can be expressed as

$$r_{n,i}^{UL} = w \log_2 \left(1 + \frac{P^{VE} h_{n,i}^{UL}}{\varpi_0}\right), \tag{3}$$

where w is the bandwidth for V2V communications, P^{VE} is the transmission power of vehicles, $h_{n,i}^{UL}$ is the uplink channel gain for V2V communications, ϖ_0 is the Gaussian noise.

Similarly, we denote z_i^l as the computation capacity allocated by assistant vehicle i to compute the task ψ_n. Thus, the collaborative computation latency of executing task ψ_n on assistant vehicle i is

$$T_i^{comp} = \frac{D_n K_n}{z_i^l}, \tag{4}$$

The computation results need to be transmitted back from vehicle i to vehicle n, similarly, the downlink transmission latency can be expressed as

$$T_{n,i}^{DL} = \frac{X_n}{r_{n,i}^{DL}}, \tag{5}$$

where $r_{n,i}^{DL}$ is the downlink transmission rate from vehicle i to vehicle n, can be expressed as

$$r_{n,i}^{DL} = w \log_2 \left(1 + \frac{P^{VE} h_{n,i}^{DL}}{\varpi_0}\right), \tag{6}$$

where $h_{n,i}^{DL}$ is the downlink channel gain for V2V communications.

Thus, the total latency for local computing layer can be written as

$$T_n^L = \begin{cases} T_n^{comp}, & \text{local execution} \\ T_{n,i}^{UL} + T_i^{comp} + T_{n,i}^{DL}, & \text{collaborative execution.} \end{cases} \tag{7}$$

2) Mobile Edge Computing Model: The offloading procedure can be divided into four steps, if the computation task ψ_n is performed in MEC layer. The vehicle n first uploads the input data of task to the nearest RSU, and the RSU will forward the data to the close MEC server m. Then, the available computing resources in MEC server m will be allocated to execute the computation task, and MEC server k will be chose to collaborative computing if there are not enough computing resources in MEC server m. Here, we define data size ratio for MEC server k as $\chi_n \in [0, 1]$ [33]. Specifically, $\chi_n = 0$ means that MEC server m will complete the computation task with the latency constant. Third, MEC server k feeds back the computation results to MEC server m. Finally, MEC server m returns the execution results to vehicle n. For MEC server m, the total computation capacity can be defined as z^m, and the allocated computation capacity for computation task ψ_n is z_n^m.

According to the above steps, the upload transmission latency of vehicle n can be defined as

$$T_{n,m}^{UL} = \frac{K_n}{r_n^{UL}}, \tag{8}$$

where r_n^{UL} is the upload transmission rate of vehicle n, can be expressed as

$$r_n^{UL} = w_n^{UL} \log_2 \left(1 + \frac{P^{VE} h_{n,m}^{UL}}{\varpi_0}\right), \tag{9}$$

where w_n^{UL} is the uplink bandwidth for V2I communications between RSU and vehicle n, P^{VE} is the transmission power of vehicle n, $h_{n,m}^{UL}$ is the uplink channel gain for V2I communications, ϖ_0 is the Gaussian noise.

The processing latency to perform computation task can be expressed as

$$r_{n,m,k}^{comp} = \frac{D_n \cdot (1 - \chi_n) \cdot K_n}{z_n^m} + \frac{D_n \cdot \chi_n \cdot K_n}{z_n^k}, \tag{10}$$

The download transmission latency of task computation results can be expressed as

$$T_{m,n}^{DL} = \frac{X_n}{r_n^{DL}}, \tag{11}$$

where r_n^{DL} is the download transmission rate, can be expressed as

$$r_n^{DL} = w_n^{DL} \log_2 \left(1 + \frac{P^{RSU} h_{n,m}^{DL}}{\varpi_0}\right), \tag{12}$$

where w_n^{DL} is the downlink bandwidth for V2I communications between RSU and vehicle n, P^{RSU} is the transmission power of RSU, $h_{n,m}^{DL}$ is the downlink channel gain for V2I communications, ϖ_0 is the Gaussian noise.

Thus, the total latency for MEC offloading can be written as

$$T_n^M = T_{n,m}^{\mathrm{UL}} + T_{n,m,k}^{\mathrm{comp}} + T_{m,n}^{\mathrm{DL}}. \tag{13}$$

where the data transmission latency between MEC servers is usually omitted due to the small value relative to the task processing latency [3].

3) Cloud Computing Model: The offloading procedure can be divided into three steps, if the computation task ψ_n is performed in cloud computing layer. First, the vehicle n uploads the input data of task to the nearest RSU, then the RSU forward the data to the remote cloud server. Second, the available computing resources in cloud server will be allocated to process the

computation task. The execution result will be returned to the close RSU, then the vehicle n, finally. For cloud server, the total computation capacity can be defined as z^c, and the allocated computation capacity for computation task ψ_n is z_n^c.

According to the above steps, the upload transmission latency of vehicle n can be defined as

$$T_{n,c}^{UL} = T_{n,m}^{UL} = \frac{K_n}{r_n^{UL}}, \tag{14}$$

Here, we consider the transmission latency of the vehicle is the same as the one in mobile edge computing model, since the task is uploaded to the nearest RSU in these two models.

The processing latency for cloud server to perform the task can be expressed as

$$T_{n,c}^{comp} = \frac{D_n K_n}{f_n^c}, \tag{15}$$

The transmission latency between RSU and cloud server is defined as T_c, usually it is set to a fix value.

The download transmission latency for cloud server can be expressed as

$$T_{c,n}^{DL} = T_{m,n}^{DL} = \frac{X_n}{r_n^{DL}}. \tag{16}$$

which is also the same as the one in mobile edge computing model.

Thus, the total latency for cloud computing offloading can be written as

$$T_n^C = T_{n,c}^{UL} + T_c + T_{n,c}^{comp} + T_{c,n}^{DL}. \tag{17}$$

III. Problem Formulation

The joint task offloading and resource allocation will be formulated as an optimization problem in this section. And the goal is to obtain the minimum total system cost composed of latency and transmission rate for all vehicles in the networks.

For vehicle n, the latency cost can be defined as

$$C_n^{del} = \alpha_n^l T_n^L + \alpha_n^m T_n^M + \alpha_n^c T_n^C. \tag{18}$$

And the transmission rate cost can be defined as

$$C_n^{re} = r_n^{UL} + r_n^{DL}. \tag{19}$$

The relationship between transmission rate and bandwidth can be expressed as

$$\begin{aligned} r_n &= w_n \log_2\left(1 + \frac{Ph}{\varpi_0}\right) \\ &= \beta_n W \log_2\left(1 + \frac{Ph}{\varpi_0}\right) \\ &= \beta_n r, \end{aligned} \tag{20}$$

which is a general expression of transmission rate.

Substituting (9) (12) into (19), the transmission cost can be rewritten as

$$C_n^{re} = \beta_n (r^{UL} + r^{DL}), \tag{21}$$

where r^{UL} and r^{DL} are the achievable uplink and downlink transmission rate without resource allocation, respectively.

Under the limitations of maximum tolerable latency τ_n, total computing resources in MEC layer R, and total bandwidth W, the problem can be formulated as follows

$$\begin{aligned} &\min_{\mathbf{A},\mathbf{B},\mathbf{F}} \sum_{n=1}^{N} I_d C^{del} + I_r \frac{1}{C^{re}} \\ &= \min_{\mathbf{A},\mathbf{B},\mathbf{F}} \sum_{n=1}^{N} I_d(\alpha_n^l T_n^L + \alpha_n^m T_n^M + \alpha_n^c T_n^C) + I_r \frac{1}{\beta_n (r^{UL} + r^{DL})} \\ &s.t.\ C1: \alpha_n^l, \alpha_n^m, \alpha_n^c \in (0,1), \alpha_n^l + \alpha_n^m + \alpha_n^c = 1, \\ &\quad C2: \alpha_n^l T_n^L + \alpha_n^m T_n^M + \alpha_n^c T_n^C \le \tau_n, \forall n \in N, \\ &\quad C3: 0 \le f_n^m \le \alpha_n^m R, \forall n \in N, \\ &\quad C4: \sum_{n=1}^{N} \alpha_n^m f_n^m \le R, \forall n \in N, \\ &\quad C5: 0 < \beta_n \le 1, \sum_{n=1}^{N} \beta_n = 1, \\ &\quad C6: \sum_{i=1}^{N} (w_i^{UL} + w_i^{DL}) \le W, \\ &\quad C7: r_n^{min} \le \beta_n r \le r_n^{UL}. \end{aligned} \tag{22}$$

In the above formula, I_d and I_r are weights of latency cost and reliability cost. And transmission rate is considered as a metric of reliability, as discussed in Section I. For different types of computation tasks, the demands of reliability and latency are different, thus, I_d and I_r can be set to different values with $I_d + I_r = 1$. Here, we consider reliability cost is the inverse of transmission rate, since generally the higher reliability means the lower reliability cost. More details, $\mathbf{A}$ and $\mathbf{B}$ are the offloading decision vector and the bandwidth allocation vector, respectively, defined in Section II-B. And $\mathbf{F} = [f_1, f_2, \ldots, f_N]$ is the computing resource allocation vector in MEC layer.

The constraint C1 indicates that the computation task ψ_n would be performed in one of the layers. C2 means that the latency cost should be less than the maximum tolerable latency. C3 means that the allocated computing resources for each vehicle cannot exceed the total computing resources in MEC layer, and C4 means that the allocated computing resources for all vehicles cannot exceed the total computing resources in MEC layer. C5 indicates that the allocated bandwidth cannot exceed the total available bandwidth, and C6 means the joint uplink and downlink bandwidth constraints [34]. C7 means that the task transmission rate needs to guarantee the QoS requirement, and cannot exceed the achievable transmission rate. Besides, the computing resource constraint in cloud computing layer is not considered due to the powerful computation capacity.

Problem (22) would be resolved by obtain the optimal offloading decision vector $\mathbf{A}$, the bandwidth allocation vector $\mathbf{B}$, and the computing resource allocation vector $\mathbf{F}$. More specifically, offloading decision vector $\mathbf{A}$ will decide which layer is chosen to

execute the computation task. The bandwidth allocation vector $\mathbf{B}$ will decide how much bandwidth is allocated to computation task, which is relative to the transmission reliability. The computing resource allocation vector $\mathbf{F}$ will decide how much computing resources are allocated for the task. Since $\mathbf{A}$ is a binary vector, the feasible set and the objective function of the problem (22) are not convex. In addition, the size of optimal $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{F}$ will be increased rapidly with the number of computation tasks. And this issue will be exacerbated due to the volatile topologies of IoV networks. The traditional methods are transform the non-convex problem into a convex one [35]. However, these methods may be not appropriate with the data size of computation tasks, especially for multi-objective optimization problem. Therefore, we use KNN and multi-objective reinforcement learning to obtain the optimal $\mathbf{A}$, $\mathbf{B}$, $\mathbf{F}$, which results in the optimal solution of problem (22).

IV. Intelligent Communication and Computation Resource Allocation Strategy

In this section, a multi-objective reinforcement learning strategy, namely, intelligent communication and computation resource allocation strategy is proposed to obtain the optimal $\mathbf{A}$, $\mathbf{B}$, $\mathbf{F}$.

The proposed strategy can be decomposed into three algorithms. Algorithm 1, called joint computation offloading and collaboration, is a general framework. In Algorithm 1, cross-layer offloading or co-layer collaboration for computation tasks is first chosen by using KNN method. Then, Algorithm 2 and Algorithm 3 are used to allocate the computation and communication resources in the chosen layer. More specifically, Algorithm 2, namely, collaborative vehicle selection, is used to find the target vehicle to execute collaborative computing in local computing layer. The algorithm is considered in terms of distance, motion direction, and vehicle computation capacity. Meanwhile, Algorithm 3, namely, multi-objective resource allocation, is used to allocate the computation and communication resources for task offloading or collaboration to guarantee the reliability and latency constraints when MEC layer is chosen to execute the computation tasks. And multi-objective reinforcement learning is used in the algorithm to compromise between reliability and latency. Then these three algorithms will be presented in details.

Algorithm 1: Joint Computation Offloading and Collaboration.

Phase 1: Initialization
a) Computation task set $\{\psi_1, \psi_2, \ldots, \psi_N\}$.
b) Maximum tolerable latency τ_{ψ_i} of computation task ψ_i, task input size K_{ψ_i}, task output size X_{ψ_i}.
c) Current position of the task $P = 1$, task set for MEC layer ς.
d) General latency for local computing layer L, MEC layer M, and cloud computing layer C, τ_1, τ_2, τ_3, and the allowed maximum task size K_1, K_2, K_3.
Phase 2: Select the computing layer
Calculate the Euclidean distance E between $(\tau_{\psi_i}, K_{\psi_i})$ and (τ_1, K_1), $E = \sqrt{(\tau_{\psi_i} - \tau_1)^2 + (K_{\psi_i} - K_1)^2}$, $P = 1$.
for $j = 2, 3$ **do**
  Calculate the Euclidean distance d_j between $(\tau_{\psi_i}, K_{\psi_i})$ and (τ_j, K_j).
  if $d_j < E$ **then**
    $E = d_j, P = j$.
  end if
end for
Phase 3: Task offloading decision
for each computation task ψ_i **do**
  if $P = 1$ and $T_i^L < \tau_{\psi_i}$ **then**
    Calculate task in local computing layer, execute Algorithm 2.
  else
    Put task ψ_i into task set ς.
  end if
  if $P = 2$ **then**
    Put task ψ_i into task set ς.
  end if
  if $P = 3$ **then**
    Offload task into cloud computing layer.
  end if
end for
if $\varsigma \neq \phi$ **then**
  Execute Algorithm 3.
end if

A. Computing Layer Selection

Algorithm 1 is a general framework to joint computation and communication resource allocation in local computing layer, MEC layer, and cloud computing layer. First, the appropriate computing layer for task execution is selected. Then, Algorithm 2 or Algorithm 3 is used to perform task in local computing layer or MEC layer.

More specifically, KNN method is used to select the appropriate computing layer, because it is a simple but efficient classifier for satisfactory performance [36]. Then, latency τ and task size K are normalized. As defined in Algorithm 1, E is the Euclidean distance [37] between the task vehicle and the selected layer, and can be expressed as

$$E = \sqrt{(\tau_{\psi_j} - \tau_i)^2 + (K_{\psi_j} - K_i)^2}. \tag{23}$$

In the initialization phase, a task set for performing in MEC layer is defined as ς. Further, in task offloading decision phase, $P = 1$ means the task is performed in local computing layer, $P = 2$ means the task is performed in MEC layer, and $P = 3$ means the task is performed in cloud computing layer.

In computing layer selection phase, the appropriate computing layer is chosen by using Euclidean distance. Then, task offloading decision is performed. In this phase, Algorithm 2 and Algorithm 3 are embedded. If $P = 1$ and the latency constraint is satisfied, the task will be calculated in the local computing layer, and the Algorithm 2 will be executed. The computation

Algorithm 2: Collaborative Vehicle Selection.

Phase 1: Initialization
a) Calculate the distance between vehicle i and vehicle j, L_{ij}, the motion direction of vehicles d_i, d_j, the available computation capacity of vehicles z_i, z_j.
b) Task size of ψ_i, K_{ψ_i}, and the required computation capacity per bit D_{ψ_i}.
c) The distance threshold ε_l.
d) The assistant vehicle $\pi = 0$.
Phase 2: Task computation collaboration
if z_i is available **then**
  Calculate task ψ_i.
else
  for candidate vehicle j **do**
    if $L_{ij} < \varepsilon_l$ and $d_i = d_j$ **then**
      if $z_j \geq K_{\psi_i} E_{\psi_i}$ **then**
        $\pi = j$, vehicle i sends task to vehicle j, vehicle j calculate the task ψ_i, then feeds back to vehicle i.
        break
      else
        continue
      end if
    end if
  end for
end if
if $\pi = 0$ **then**
  offload the task to MEC layer.
end if

Algorithm 3: Multi-Objective Resource Allocation.

Phase 1: Initialization
a) Q matrix $Q(s, a)$, learning parameter γ, reward matrix R.
b) Total system cost in local computing layer tc_{local}.
c) Multi-objective function set $O = \{o_1, o_2, \ldots, o_m\}$.
d) Multi-objective weight set $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_m\}$.
e) Multi-objective Q matrix $\mathbf{Q} = \{Q_1, Q_2, \ldots, Q_m\}$.
Phase 2: Resource allocation
for every episode **do**
  Select a random state s_t.
  for every step **do**
    Select an action a from the possible actions of state s_t.
    Execute selected action a, observe reward r_i of each objective o_i, $r_i = \frac{1}{tc^i(s,a)}$, update Q_i matrix.
    Update $\mathbf{Q}$ corresponds to Λ, $\mathbf{Q} = \sum_{i=1}^{n} \lambda_i Q_i$.
    Calculate $Q(s, a) \leftarrow R(s, a) + \gamma \cdot \max Q(s,' a')$.
    Update state $s \leftarrow s'$.
    until $\mathbf{Q}$ matrix coverage.
  end for
end for

tasks may not be executed in local computing layer due to the latency constraint. In such case, the tasks will be offloading to MEC layer or cloud computing layer. After that, Algorithm 3 will be executed if the task set for MEC layer is not empty.

B. Inter-Vehicle Collaboration

Algorithm 2 is supposed to find the target vehicle to assist the task computing in local computing layer. In such case, distance, motion direction, and vehicle computation capacity are considered to search the appropriate assistant vehicle. If not, the task will be offloaded to MEC or cloud layer.

In the initialization phase, vehicle i generates the computation task, and vehicle j is the candidate vehicle to assist the collaborative computing. In the task computation collaboration phase, the assistant vehicle is determined by using system information, such as distance, motion, computation capacity, etc.. And collaborative computing will be executed if the available assistant vehicle is found. If not, the computation task will be offloaded to other layers.

C. Multi-Objective Resource Allocation

Algorithm 3 will be executed if the task offloaded to MEC layer. Specifically, multi-objective resource allocation using reinforcement learning is developed to guarantee the reliability and latency. Latency cost and reliability cost are supposed to be objective functions, thus objective function of problem (22) can be reformulated as

$$\min_{\mathbf{A},\mathbf{B},\mathbf{F}} \sum_{n=1}^{N} I_d O_1 + I_r O_2, \tag{24}$$

where O_1 and O_2 are the latency cost and reliability cost, respectively.

Then, multi-objective reinforcement learning is used to allocate the computation and communication resources in MEC layer to minimize the total system cost in (24). $I = [I_d, I_r]$ is defined as the weight vector. If $I_d > I_r$, it indicates that the latency requirement is much higher than the reliability requirement, and vice versa. For the special cases, $I_d = 1$ represents that only latency cost is considered in the total system cost, similarly, $I_r = 1$ represents that only reliability cost is considered in the total system cost.

To obtain the optimal policy, the state, action, and reward will be presented as follows.

1) State: For vehicle n, the state vector at the time $t \in \{0, 1, \ldots, T-1\}$ can be expressed as $s = (tc, ac)$, where tc is the system cost and ac is the available resources. For latency cost O_1, ac is the available computation capacity can be expressed as $ac = R - \sum_{n=0}^{N} z_n^m$, and for reliability cost O_2, ac is available bandwidth resources.

2) Action: Actions are defined as the bandwidth allocation vector $\mathbf{B} = [\beta_1, \beta_2, \ldots, \beta_N]$ and the computing resource allocation vector $\mathbf{F} = [f_1, f_2, \ldots, f_N]$. For latency cost O_1, different computing resource allocation vector corresponds to different computing resource allocation, which leads to different latency cost. Similarly, for reliability cost O_2, different bandwidth allocation vector corresponds to different bandwidth allocation, which leads to different reliability cost.

3) Reward: The objective of Problem (22) is to obtain the minimum total system cost. While for reinforcement learning, the goal is transform to get the maximum reward, that is, $r = \frac{1}{tc(s,a)}$, where $tc(s,a)$ is the total system cost.

For reliability cost O_2, $tc(s,a) = \frac{1}{\sum_{n=1}^{N} r_n^{UL} + r_n^{DL}}$. Meanwhile, for latency cost O_1, two cases (i.e., $ac = 0$ and $ac \neq 0$) are considered to calculate $tc(s,a)$.

If $ac = 0$, it indicates that there are no computing resources in MEC server m, and the MEC server k is chosen to task computing. In this case, latency cost O_1 can be expressed as

$$\begin{aligned} tc(s,a) &= \sum_{n=1}^{N} T_{n,m}^{UL} + T_{n,k}^{comp} + T_{k,n}^{DL} \\ &= \sum_{n=1}^{N} T_{n,m}^{UL} + \frac{D_n K_n}{z_n^m} + T_{k,n}^{DL}. \end{aligned} \tag{25}$$

If $ac \neq 0$, it indicates that there are available computing resources in MEC server m, therefore, task will be executed in MEC server m, or collaborative computing will be performed between MEC servers m and k. In this case, latency cost O_1 can be expressed as

$$\begin{aligned} tc(s,a) &= \sum_{n=1}^{N} T_{n,m}^{UL} + T_{n,m,k}^{comp} + T_{m,n}^{DL} \\ &= \sum_{n=1}^{N} T_{n,m}^{UL} + \frac{D_n(1-\chi_n)K_n}{z_n^m} + \frac{D_n \chi_n K_n}{z_n^k} \\ &\quad + T_{m,n}^{DL}, \end{aligned} \tag{26}$$

where χ_n is the data size ratio for computation collaboration MEC server, can be expressed as

$$\chi_n = 1 - \frac{ac}{K_n \cdot D_n}. \tag{27}$$

In the multi-objective reinforcement learning procedure, Q-learning is used and Q values are recorded for problem optimization. Specifically, each state-action pair corresponds to a Q value. Calculate and store the Q matrix in the agent in every step, and the value is treated as the long-term reward, can be formulated as

$$Q(s,a) = R(s,a) + \gamma \cdot \max Q(s,' a'), \tag{28}$$

where s and a represent the current state and the current action, respectively, and s' and a' represent the next state and the next action, respectively. Learning parameter is denoted as γ, with $0 \leq \gamma \leq 1$. If γ tends to zero, the agent mainly considers the direct rewards, if γ tends to one, the agent mainly concerns the future rewards. And the value will be iterated for every step.

For the multi-objective reinforcement learning, the Q matrix of latency cost O_1 and reliability cost O_2 are first calculated separately. Then, Q matrix for the total system cost is calculated with the weight vector of latency cost and reliability cost, can be expressed as

$$Q = I_d Q_d + I_r Q_r. \tag{29}$$

TABLE II
PARAMETER SETTINGS

Parameters	Values
Number of vehicle	$1 \sim 4$
Number of RSU/MEC	2
Local computation capacity z^l	1 GHz
MEC computation capacity z^m	5 GHz
Cloud computation capacity z^c	20 GHz
Computation task size	$200 \sim 1000$ kbits
Bandwidth	10 MHz
Distance between vehicle and RSU	$0 \sim 1000$ m
Vehicle transmission power	0.2 W
RSU transmission power	0.2 W
Standard deviation Gaussian noise	10^{-13} W

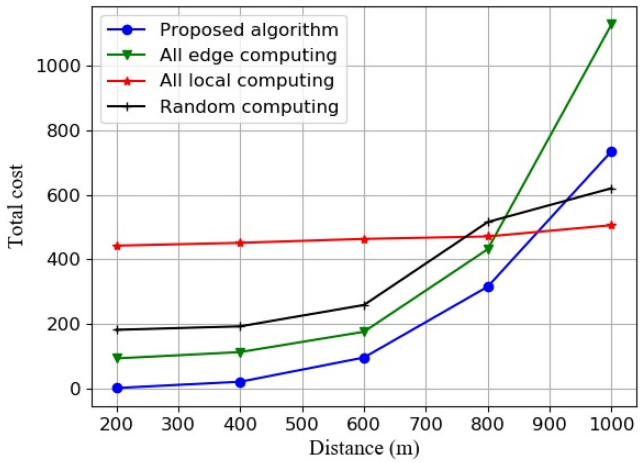


Fig. 2. Total system cost with different distance.

where Q_d and Q_r are the Q matrix of latency cost O_1 and reliability cost O_2, respectively. The optimal $\mathbf{F}$ and $\mathbf{B}$ can be obtained when the Q value coverage after the value iterations.

V. NUMERICAL RESULTS

In this section, we will evaluate the performance of the proposed strategy by simulation. Further, we set the bandwidth $W = 10$ MHz, and the MEC is deployed in each RSU. This paper uses the WINNER II D2a channel model, which is measured and modeled in the highway road scenario. Meanwhile, mobility is considered through this channel model. The system parameters are listed in Table II.

The baseline strategies to compare the proposed strategy are as follows.

1) All Edge Computing: in this strategy, computation tasks are offloaded to the MEC layer.

2) All Local Computing: in this strategy, computation tasks are performed in the local computing layer.

3) Random Computing: in this strategy, computation tasks are executed in MEC layer or local computing layer randomly.

The total system cost comparisons with different distance are shown in Fig. 2. The number of vehicle task is set to two, and the task size is generated randomly. The system cost of the proposed strategy is always lower than that of all the baseline strategies

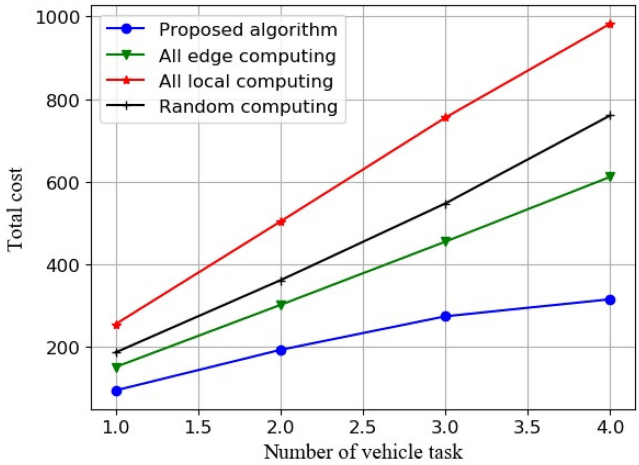


Fig. 3. Total system cost with the different number of vehicle task.

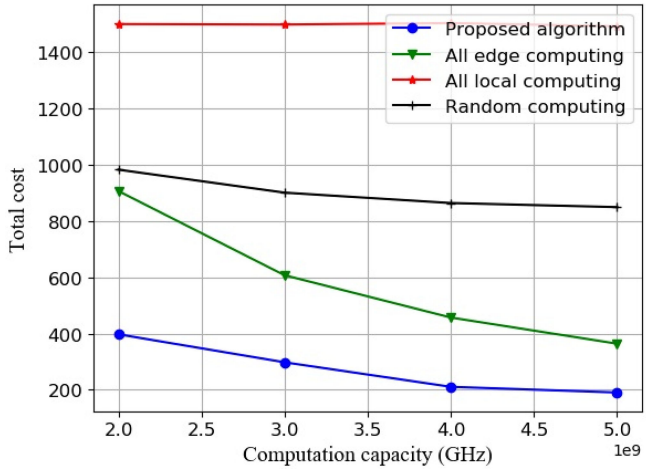


Fig. 5. Total system cost with different computation capacity.

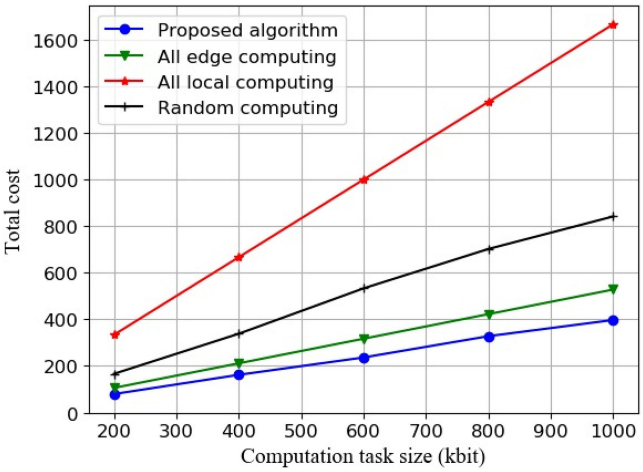


Fig. 4. Total system cost with the different number of computation task size.

when the distance of RUS and vehicle is smaller than 880 m. And the total system cost is beyond the all local computing and random computing when the distance is 930 m. It indicates that the proposed strategy is a better one in an appropriate region. And all local computing may be a choice when the distance is larger than a given value, in this case, it is 880 m.

Fig. 3 shows the total system cost comparisons with the different number of vehicle task. The distance between RSU and vehicle is set to 700 m, and the task size is generated randomly. The performance of all the strategies increases with the number of vehicle task. The slope of each curve is different, and the gap between each other becomes larger and larger with the number of vehicle tasks. For example, when the number of vehicle tasks is 3, the total system cost of the proposed strategy is 250, and the total system cost of all edge computing, random computing, and all local computing is 470, 550, and 760, respectively. Thus, for the proposed strategy, 46.8.%, 54.5%, and 67.1% system cost can be saved compared to each baseline strategy, and the average 56% system cost is saved.

The total system cost comparisons with different computation task size are shown in Fig. 4. The number of vehicle task is set to three, and the distance is set to 700 m. Obviously, the performance of strategies increases with the number of computation task size. And the total system cost is always the smaller one than other strategies. For example, when the task size is 600, the total system cost of the proposed strategy is 210, while the total system of all edge computing, random computing, and all local computing is 310, 520, and 1000, respectively. It means that, for the proposed strategy, 32.2%, 59.6%, and 79% system cost can be saved compared to each baseline strategy. Further, for all local computing, the curve increases sharply with computation task size, it indicates that, MEC and cloud computing are necessary for IoV networks.

The total system cost comparisons with different computation capacity are shown in Fig. 5. The number of vehicle task is set to three, the distance is set to 700 m, and the task size is generated randomly. The curve of all local computing is not changed with the computation capacity of MEC due to no task is offloaded to other layers. The total system cost of other strategies is decrease with computation capacity. These trends are caused because more and more task will be offloaded to MEC layer due to the increased computation capacity. It is note that, the gap between the proposed strategy and all edge computing becomes smaller and smaller with computation capacity. Thus, an efficient resource allocation design is important especially for IoV networks.

The complementary cumulative distribution function (CCDF) of the proposed strategy with different task offloading ratio is plotted in Fig. 6, which describes the cumulative probability that a variable is greater than a certain value. The x-axis represents the ratio allocated for task offloading as a percentage of the total rate, and the y-axis represents the probability that the ratio is greater than the value of x-axis. From Fig. 6, we can see that, when the distance between the vehicle and the RSU is 700 m, the 10% task offloading ratio can be guaranteed with almost 100% probability in the proposed strategy.

Acquiring accurate wireless channel state information is a vital challenge in the fast-varying vehicular environments, especially for V2V and V2I links. Fortunately, RL can provide a robust and principled way to treat environment dynamics and

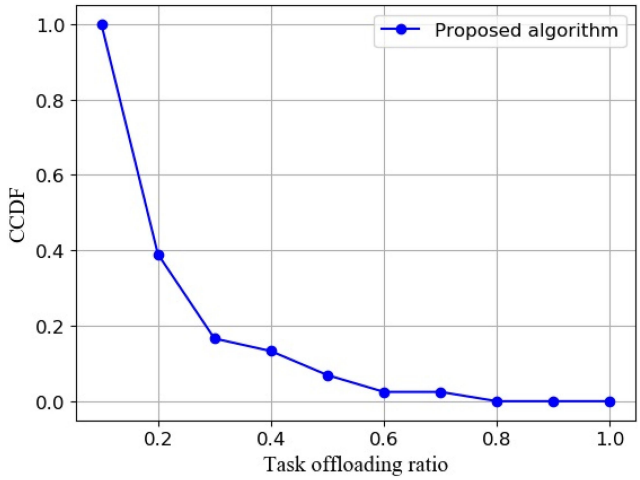


Fig. 6. CCDF of the proposed strategy.

perform the sequential decision making under uncertainty [38], thus representing a promising method to handle the challenging dynamics of V2V and V2I links. Therefore, we can implement RL to capture the environment characteristics to build the multi-path fingerprint database, which stores the long-term multipath channel characteristics associated with vehicles location [39]. Several fingerprinting-based positioning schemes using machine learning approaches have been proposed [40], [41] in recent years. In fingerprinting-based localization methods, there is a fingerprint database, which records fingerprints at different locations. By using RL, each vehicle can obtain the geometry of the environment such as the roadside infrastructures and vehicles to train the V2I and V2V links. Once the database is built, we can learn the long-term multipath information from the database to lower reduce the cost.

VI. Conclusion

For IoV networks, ultra reliability and low latency are key issues and should be addressed efficiently, especially for computation-intensive tasks. Thus, in this work, a multi-objective reinforcement learning strategy is proposed to meet the various task requirements in IoV networks. Specifically, three algorithms are proposed in this strategy. Numerical results show the effectiveness of the proposed strategy. In the vehicular environment, the impact of network topology and MEC capacity limitation should be considered carefully. Thus, the efficient machine learning-enabled resource allocation algorithm for multi-objective optimization will be investigated in depth in the future work.

References

[1] S. Elayoubi, P. Brown, M. Deghel, and A. Galindo-Serrano, "Radio resource allocation and retransmission schemes for URLLC over 5G networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 4, pp. 896–904, Apr. 2019.

[2] Y. Ni, J. He, L. Cai, and Y. Bo, "Delay analysis and message delivery strategy in hybrid V2I/V2V networks," in *Proc. IEEE Glob. Commun. Conf.*, Washington, DC, USA, 2016, pp. 1–6.

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

[4] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, and S. Chen, "Vehicular fog computing: A viewpoint of vehicles as the infrastructures," *IEEE Trans.Veh. Technol.*, vol. 65, no. 6, pp. 3860–3873, Jun. 2016.

[5] R. Xie, Q. Tang, Q. Wang, X. Liu, F. Yu, and T. Huang, "Collaborative vehicular edge computing networks: Architecture design and research challenges," *IEEE Access*, vol. 7, pp. 178942–178952, Dec. 2019.

[6] M. Jia, J. Cao, and W. Liang, "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks," *IEEE Trans. Cloud Comput.*, vol. 5, no. 4, pp. 725–737, Oct.–Dec. 2017.

[7] Y. Wang, X. Tao, X. Zhang, P. Zhang, and Y. Hou, "Cooperative task offloading in three-tier mobile computing networks: An ADMM framework," *IEEE Trans.Veh. Technol.*, vol. 68, no. 3, pp. 2763–2776, Mar. 2019.

[8] H. Ye, L. Liang, G. Li, J. Kim, L. Lu, and M. Wu, "Machine learning for vehicular networks: Recent advances and application examples," *IEEE Veh. Technol. Mag.*, vol. 13, no. 2, pp. 94–101, Jun. 2018.

[9] C. Luo, J. Ji, Q. Wang, X. Chen, and P. Li, "Channel state information prediction for 5G wireless communications: A deep learning approach," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 1, pp. 227–236, Jan.–Mar. 2020.

[10] R. Wang, H. Yang, H. Wang, and D. Wu, "Social overlapping community-aware neighbor discovery for D2D communications," *IEEE Wireless Commun.*, vol. 23, no. 4, pp. 28–34, Aug. 2016.

[11] R. Wang, H. Liu, H. Wang, Q. Yang, and D. Wu, "Distributed security architecture based on blockchain for connected health: Architecture, challenges, and approaches," *IEEE Wireless Commun.*, vol. 26, no. 6, pp. 30–36, Dec. 2019.

[12] M. Ashraf, C. Liu, M. Bennis, W. Saad, and C. Hong, "Dynamic resource allocation for optimized latency and reliability in vehicular networks," *IEEE Access*, vol. 6, pp. 63843–63858, Nov. 2018.

[13] L. Feng, W. Li, Y. Lin, L. Zhu, S. Guo, and Z. Zhen, "Joint computation offloading and URLLC resource allocation for collaborative MEC assisted Cellular-V2X network," *IEEE Access*, vol. 8, pp. 24914–24926, Feb. 2020.

[14] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Top. Comput.*, vol. 9, no. 3, pp. 1529–1541, Jul.–Sep. 2021.

[15] Y. Wang *et al.*, "A game-based computation offloading method in vehicular multi-access edge computing networks," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4987–4996, Jun. 2020.

[16] T. Vu, M. Bennis, M. Debbah, M. Latva-aho, and C. Hong, "Ultra-reliable communication in 5G mmWave networks: A risk-sensitive approach," *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 708–711, Apr. 2018.

[17] S. Batewela, C. Liu, M. Bennis, H. Suraweera, and C. Hong, "Risk-sensitive task fetching and offloading for vehicular edge computing," *IEEE Commun. Lett.*, vol. 24, no. 3, pp. 617–621, Mar. 2020.

[18] Y. Xu, G. Gui, H. Gacanin, and F. Adachi, "A survey on resource allocation for 5G heterogeneous networks: Current research, future trends, and challenges," *IEEE Commun. Surv. Tut.*, vol. 23, no. 2, pp. 668–695, Apr.–Jun. 2021.

[19] W. Sun, E. Strom, F. Brannstrom, K. Sou, and Y. Sui, "Radio resource management for D2D-Based V2V communication," *IEEE Trans.Veh. Technol.*, vol. 65, no. 8, pp. 6636–6650, Aug. 2016.

[20] T. Vu, C. Liu, M. Bennis, M. Debbah, M. Latva-aho, and C. Hong, "Ultra-reliable and low latency communication in mmWave-Enabled massive MIMO networks," *IEEE Commun. Lett.*, vol. 21, no. 9, pp. 2041–2044, Sep. 2017.

[21] C. Liu, M. Bennis, M. Debbah, and H. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.

[22] N. Kherraf, S. Sharafeddine, C. Assi, and A. Ghrayeb, "Latency and reliability-aware workload assignment in IoT networks with mobile edge clouds," *IEEE Trans. Netw. Ser. Manag.*, vol. 16, no. 4, pp. 1435–1449, Dec. 2019.

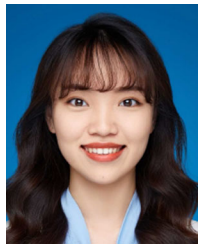
[23] W. R. Ghanem, V. Jamali, Y. Sun, and R. Schober, "Resource allocation for multi-user downlink MISO OFDMA-URLLC systems," *IEEE Trans. Commun.*, vol. 68, no. 11, pp. 7184–7200, Nov. 2020.

[24] H. Yang, K. Zhang, K. Zheng, and Y. Qian, "Joint frame design and resource allocation for ultra-reliable and low-latency vehicular networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3607–3622, May 2020.

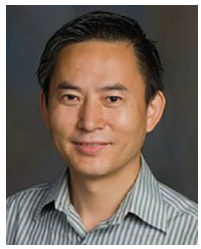
[25] C. Lin, D. Deng, and C. Yao, "Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units," *IEEE Internet Things J.*, vol. 5, no. 5, pp. 3692–3700, Oct. 2018.

[26] Y. Xu, H. Xie, and R. Q. Hu, "Max-min beamforming design for heterogeneous networks with hardware impairments," *IEEE Commun. Lett.*, vol. 25, no. 4, pp. 1328–1332, Apr. 2021.
[27] Y. Xu and G. Gui, "Optimal resource allocation for wireless powered multicarrier backscatter communication networks," *IEEE Wireless Commun. Lett.*, vol. 9, no. 8, pp. 1191–1195, Aug. 2020.
[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[29] Y. Xia, L. Wu, Z. Wang, X. Zheng, and J. Jin, "Cluster-enabled cooperative scheduling based on reinforcement learning for high-mobility vehicular networks," *IEEE Trans.Veh. Technol.*, vol. 69, no. 11, pp. 12664–12678, Nov. 2020.
[30] Y. Zhou, F. Tang, Y. Kawamoto, and N. Kato, "Reinforcement learning-based radio resource control in 5G vehicular network," *IEEE Wireless Commun. Lett.*, vol. 9, no. 5, pp. 611–614, May 2020.
[31] H. Liang, X. Zhang, J. Zhang, Q. Li, S. Zhou, and L. Zhao, "A novel adaptive resource allocation model based on SMDP and reinforcement learning algorithm in vehicular cloud system," *IEEE Trans.Veh. Technol.*, vol. 68, no. 10, pp. 10018–10029, Oct. 2019.
[32] S.-S. Lee and S. Lee, "Resource allocation for vehicular fog computing using reinforcement learning combined with heuristic information," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10450–10464, Oct. 2020.
[33] J. Ren, G. Yu, Y. Cai, Y. He, and F. Qu, "Partial offloading for latency minimization in mobile-edge computing," in *Proc. IEEE Glob. Commun. Conf.*, Singapore, Singapore, 2017, pp. 1–6.
[34] U. Saleem, Y. Liu, S. Jangsher, and Y. Li, "Performance guaranteed partial offloading for mobile edge computing," in *Proc. IEEE Glob. Commun. Conf.*, Abu Dhabi, UAE, 2018, pp. 1–6.
[35] F. Wang, J. Xu, and Z. Ding, "Multi-antenna NOMA for computation offloading in multiuser mobile edge computing systems," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 2450–2463, Mar. 2019.
[36] M. Chen, M. Dong, and B. Liang, "Resource sharing of a computing access point for multi-user mobile cloud offloading with delay constraints," *IEEE Trans. Mobile Comput.*, vol. 17, no. 12, pp. 2868–2881, Dec. 2018.
[37] P. Zhang, Y. Zhang, H. Dong, and H. Jin, "Mobility and dependence-aware QoS monitoring in mobile edge computing," *IEEE Trans. Cloud Comput.*, vol. 9, no. 3, pp. 1143–1157, Jul.–Sep. 2021.
[38] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.
[39] V. Va, J. Choi, T. Shimizu, G. Bansal, and R. W. Heath, "Inverse multipath fingerprinting for millimeter wave V2I beam alignment," *IEEE Trans.Veh. Technol.*, vol. 67, no. 5, pp. 4042–4058, May 2018.
[40] G. B. Tarekegn, R.-T. Juang, H.-P. Lin, A. B. Adege, and Y. Y. Munaye, "DFOPS: Deep-learning-based fingerprinting outdoor positioning scheme in hybrid networks," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3717–3729, Mar. 1, 2021.
[41] P. Wang, T. Koike-Akino, and P. V. Orlik, "Fingerprinting-based indoor localization with commercial MMWave WiFi: NLOS propagation," in *Proc. IEEE Glob. Commun. Conf.*, Taipei, Taiwan, 2020, pp. 1–6.

Yaping Cui received the B.E. degree in communication engineering from PLA Information Engineering University, Zhengzhou, China, in 2008, and the M.S. degree in communication and information system and the Ph.D. degree in traffic information engineering and control from Southwest Jiaotong University, Chengdu, China, in 2011 and 2017, respectively. From 2011 to 2012, he was a Baseband Algorithm Engineer with ZTE Corporation, Shenzhen, China. In 2017, he joined the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing, China, where he is currently a Lecturer. His research interests include mobile edge computing and network function virtualization for vehicular networks.



Lijuan Du is currently working toward the M.S. degree with the Chongqing University of Posts and Telecommunications, Chongqing, China. Her research interests include vehicular networks and mobile edge computing.



Honggang Wang (Fellow, IEEE) received the Ph.D. degree in computer engineering from the University of Nebraska-Lincoln, Lincoln, NE, USA, in 2009. From 2001 to 2004, he was a Member of technical staff with Bell Labs Lucent Technologies, China. He is currently an Associate Professor with the University of Massachusetts Dartmouth, Dartmouth, MA, USA. His research interests include Internet of Things, body area networks (BANs), cyber security, mobile multimedia and cloud, wireless networks and cyber-physical systems, and Big Data in mHealth. He is an Associate Editor-in-Chief (EiC) of the IEEE INTERNET OF THINGS JOURNAL, IEEE TRANSACTIONS ON BIG DATA, IEEE TRANSACTIONS ON MULTIMEDIA, the Editor of the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and the Technical Editor of the IEEE NETWORK MAGAZINE.

Dapeng Wu (Senior Member, IEEE) received the M.S. degree in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in June 2006 and the Ph.D. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2009. His research interests include ubiquitous networks, IP QoS architecture, network reliability, and performance evaluation in communication systems.

Ruyan Wang received the M.S. degree from the Chongqing University of Posts and Telecommunications (CQUPT), Chongqing, China, in 1997 and the Ph.D. degree from the University of Electronic and Science Technology of China, Chengdu, China, in 2007. Since December 2002, he has been a Professor with the Special Research Centre for Optical Internet and Wireless Information Networks, CQUPT. His research interests include network performance analysis and multimedia information processing.