IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 23, 2019, accepted December 30, 2019, date of publication January 6, 2020, date of current version January 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2964018

Regional Intelligent Resource Allocation in Mobile Edge Computing Based Vehicular Network

GE WANG, (Student Member, IEEE), AND FANGMIN XU

Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China
Institute of Sensing Technology and Business, Beijing University of Posts and Telecommunications, Wuxi 214135, China

Corresponding author: Fangmin Xu (xufm@bupt.edu.cn)

This work was supported by the National Science and Technology Major Project of China under Grant 2018ZX03001022.

ABSTRACT The advancement of 5G technology has brought the prosperous development of Internet of Vehicles (IoV). IoV services are not only computational intensive but also extremely sensitive to the delay. As a promising computing paradigm, mobile edge computing (MEC) can be applied to IoV scenarios. However, due to the limited resources of a single MEC server, it is difficult to cope with the suddenly increased computation loads caused by emergencies, or the intensive resource requests from busy regions. Therefore, we propose a novel regional intelligent management vehicular system with dual MEC planes, in which MEC servers in the same region cooperate with each other to achieve resource sharing. We classify computing tasks into different types according to their delay tolerances and focus on the optimization problem of resource allocation for different type tasks. And then, we design a resource allocation algorithm based on deep reinforcement learning, which can adapt to the changeable MEC environment to process high-dimensional data. Simulation results confirm that our proposed scheme is feasible and effective.

INDEX TERMS Internet of vehicles, mobile edge computing, deep reinforcement learning, resource allocation, delay tolerance.

I. INTRODUCTION

With the progress of 5G technology, the world has accelerated its pace into the 5G era. Benefit from the advantages of 5G technology, such as low latency and high bandwidth, Internet of Vehicles (IoV) has entered the rapid development stage. Internet of vehicles is the evolution of conventional Vehicular Ad Hoc Networks (VANETs), where the Vehicles-to-X (V2X) communication technology is used to exchange information between vehicles, infrastructures, pedestrians and networks. The most outstanding features of IoV services are computation-intensive and latency-sensitive. The increasing growth of vehicles, sensors and mobile devices has caused an explosion in the number and variety of computing requests, which undoubtedly increases the pressure on data processing. Besides, many IoV services have extremely strict requirements on the delay, especially when it comes to driving safety. Hence, to ensure efficient processing and low latency, it is crucial for vehicular systems to equip with sufficient processing capacity.

Due to the limited computing capabilities in on-board units (OBU) of vehicles, relevant scholars have proposed to introduce cloud computing into vehicular systems. Because of its inherent characteristics of centralized deployment and long distance from terminals, making it unable to adapt to all scenarios, especially some delay-sensitive services in internet of vehicles. As a promising computing paradigm, mobile edge computing (MEC) addresses such challenges. It deploys some service nodes at the edge of networks to provide computing resources, which reduces the computing delay greatly and avoids the waste of bandwidth caused by offloading to the cloud server [1].

However, vehicular systems with a single MEC server can hardly handle the explosion of data processing caused by emergencies, or the mass data from a busy region. Therefore, it is imperative to deploy multiple MEC servers in the vehicular system. For improving system efficiency, resource sharing can be achieved by strengthening collaboration between

The associate editor coordinating the review of this manuscript and approving it for publication was Junhui Zhao.

multiple MEC servers. To construct such a vehicular system, we mainly face the following challenges:

- Due to the high mobility, the position of the vehicle changes rapidly. Whereas, the deployment location of the service node is relatively fixed [2]. As a result, How to leverage the fixed resources to provide computing services for moving vehicles needs to be investigated.
- In the IoV system, different types of computing tasks have different demands on latency and resources. Therefore, how to allocate appropriate computing resources for different tasks to satisfy their stringent delay constraints deserves to be well studied.

Traditional methods such as the game theory [3] and the genetic annealing algorithm [4] are puzzled by the complexity. Moreover, they also have some limitations in solving resource optimization problems in the complex and changeable MEC environment [5]. As an important branch of machine learning, deep reinforcement learning (DRL) can not only solve problems with a low complexity, but also acquire knowledge from the environment, ameliorate policy to adjust to the varying environment and make a series of resource allocation decisions intelligently and adaptively.

In this paper, we emphasize the collaboration between multiple MEC servers, proposing a MEC based regional intelligent management vehicular system. We classify IoV computing tasks into different types according to their delay tolerances. Then, we mainly focus on the optimization problem of resource allocation for tasks with deferent types in the proposed system. In order to tackle the problem, we formulate the optimization problem as a Markov decision process and adopt the deep reinforcement learning technology to process high dimensional data.

Our contributions are as summarized as follows:

- We propose a novel regional intelligent management vehicular system with multi-tiers MEC servers, which can provide computation-intensive, mobility-aware and low-latency services.
- To minimize the delay, we consider the optimization problem as a Markov decision process and design an algorithm to allocate computational resources adaptively via deep reinforcement learning.

The remaining parts of this article are organized as follows. In Section II related works are discussed. Then we give the system description and models in Section III. In Section IV, we formulate the optimization problem of computing resource allocation as a Markov decision process and solve it with DRL. Simulation settings and performance evaluations of our algorithm are shown in Section V. Finally, Section VI concludes this paper.

II. RELATED WORK

In this section, we first review some related works about mobile edge computing based vehicular systems. Then recent researches of computational offloading and resource allocation schemes in MEC are presented.

A. MOBILE EDGE COMPUTING BASED VEHICULAR SYSTEMS

In recent years, there is an increasing trend to utilize MEC in vehicular systems.

Some works designed systems without the resource sharing mechanism, and only one MEC server was studied. Li et al. in [6] proposed a multi-user MEC system, where multiple user equipments realized the computational offloading via wireless channels to a MEC server. An orthogonal frequency division multiple access based cloud radio access network with an integrated MEC server is studied in [7]. Zhou *et al.* in [8] focused on the problem of reducing the completion time of Virtual Reality applications for IoV, and allowed vehicles to offload the VR tasks to the MEC server in the edge vehicular network.

Other systems include the resource sharing mechanism, but are limited to the sharing of the communication resources and the computing resources are relatively independent. A software-defined network inside the mobile edge computing architecture was designed for offloading vehicular communication traffic in [9]. Dab *et al.* in [10] envisioned a multi-user WiFi-based MEC architecture. A control/user plane split (CUPS)-based multiband cooperative scheme was presented in [11]. Instead of offloading application services to the MEC server, Zhou *et al.* in [12] modeled a vehicular MEC architecture where vehicular communication packets were routed through the MEC network.

In systems where there is only one MEC server or MEC resources cannot be shared, computing tasks can easily become congested. Hence, in order to improve system efficiency, we try to propose a dual-MEC-layers IoV system, in which management and computing functions are properly configured and computing resources are shared between multiple MEC servers.

B. COMPUTATIONAL OFFLOADING AND RESOURCE ALLOCATION SCHEMES IN MOBILE EDGE COMPUTING

Many scholars are devoted to researches of computational offloading strategies in different MEC based systems. To avoid interruption in the offloading process, Wang *et al.* in [13] provided a dynamic offloading scheduling scheme for MEC-enabled vehicular networks. Zhang and Cao in [14] provided a stochastic programming algorithm to minimize the energy consumption caused by offloading. Wang *et al.* in [15] proposed a distributed algorithm to obtain the optimal routing to offload the task of V2V.

Meanwhile, some papers also put forward different resource allocation algorithms in MEC. Wang *et al.* in [16] defined a resource allocation and power control method to optimize spectrum efficiency and system capacity in a D2D enabled MEC system of IWCN. A MEC based mission-critical wireless sensor network architecture and a kind of centralized computing resource management strategy were studied in [17]. Qiu *et al.* in [18] explained how to manage and orchestrate resources jointly in software-defined

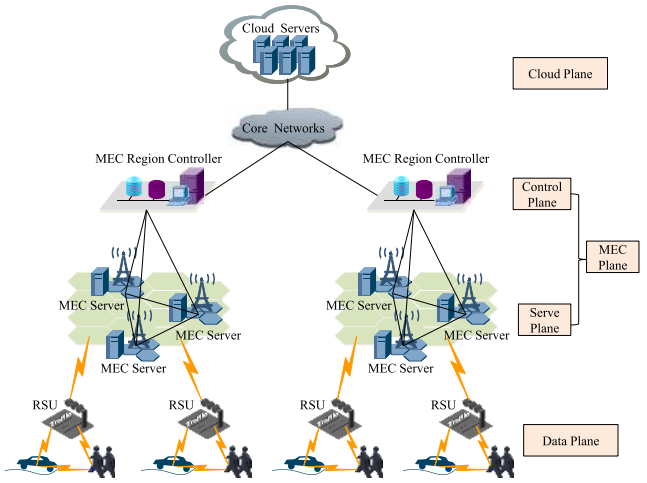


FIGURE 1. The framework of regional intelligent management IoV System.

satellite-terrestrial networks. Ye and Li in [19] developed a decentralized resource allocation mechanism for V2V communications based on deep reinforcement learning.

Some work deals with the joint problem of computing offload and resource allocation in MEC based vehicular systems. In order to obtain the maximum system utility, Zhao *et al.* in [20] designed the distributed computation offloading and resource allocation (DCORA) algorithm. Tran and Pompili in [21] jointly optimized the task offloading decision, uplink transmission power and computing resource allocation with a mixed integer nonlinear program (MINLP).

In our article, we mainly focus on the problem of resource allocation for IoV computational tasks with deferent types in the MEC based system.

III. SYSTEM DESCRIPTIONS AND MODELS

In this section, we first propose a regional intelligent management IoV system. After that, we give a brief description of this system. Then, the network model, the communication model and the computing model are introduced in detail.

A. THE REGIONAL INTELLIGENT MANAGEMENT IOV SYSTEM

To cope with the resource limitation of a single MEC server, we introduce the concept of multi-MEC servers collaboration to achieve resource sharing. And then we expand from one-MEC-layer to double-MEC-layers in the vehicular network, proposing a regional intelligent management IoV system as shown in Figure 1. In our system, the vehicular network is separated into several regions, each region is organized by a MEC region controller. Preventing a single MEC server from overloading, tasks can be calculated at other MEC servers in the same region. In the bottom-up view, our system contains three planes: the data transport plane, the MEC plane and the cloud plane.

1) DATA TRANSPORT PLANE

This plane is formed by Road Side Units (RSUs), vehicles and pedestrians within the RSU communication range. Both

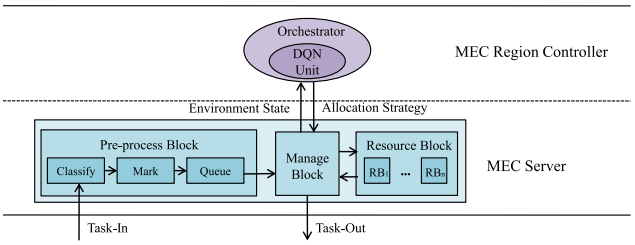


FIGURE 2. The detailed framework of MEC plane.

TABLE 1. The comparison of different computing-oriented IoV architectures.

Items	Cloud	Single	Region
Server Location	Remote	Edge	Edge
Distribution	Centralized	Decentralized	Hybrid
Mobility Support	Partially	Yes	Yes
Core Network Load	Heavy	Light	Light
MEC Server	None	Single	Multiple
Resources	Abundant	Limited	Adequate
Latency	High	Medium	Low

vehicles and pedestrians are not only the submitter of computing tasks but also the receiver of the computation results. RSUs are the network access points for data transmission and location dynamic awareness.

2) MEC PLANE

This plane has two sub-planes: the MEC region control sub-plane and the MEC serve sub-plane. In the lower sub-plane, each base station is equipped with a MEC server which provides pre-processing, management and computing services for offloaded computing tasks. In the upper sub-plane, the orchestrator of the MEC region controller is responsible for making resource allocation decisions and coordinating communications between MEC servers in the region.

3) CLOUD PLANE

In traditional cloud-computing-based systems, the cloud plane plays the role of controlling and processing information. However, the cloud servers are far from the end-users in our system. In this paper, we treat cloud resources as backup and de-emphasize the cloud functions.

Those functions are realized by the collaboration of different parts within the MEC plane. As we can see from Figure 2, tasks are classified, marked and waiting in the pre-process block. The manage block sends resource states and executes commands from the MEC region controller. The resource block pools and manages computing resources of the MEC server, the smallest computational resource unit is RB. In addition, there is a deep reinforcement learning unit in the orchestrator, which determines allocation strategies automatically and intelligently. That is the reason why our system can realize intelligent self-management.

As shown in Table 1, the advantages of our regional intelligent management IoV system (Region) compared with the

TABLE 2. Task classification.

Type	Name	Delay tolerance
1	Urgent task	0-20ms
2	Real-time task	20-150ms
3	General task	150-500ms
4	Best-effort task	$>$ 500ms

traditional cloud based IoV system (Cloud) and the single MEC based IoV system (Single) are as follow:

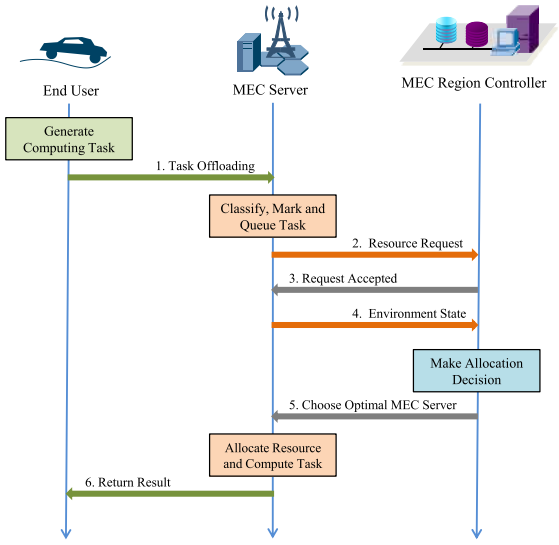
- Our system combines the advantages of decentralized and centralized systems, the data processing can be managed independently in each region.
- Better than a single MEC server, multi-MEC servers collaboration can reduce the possibility of task congestion and provide more appropriate resources for tasks.

B. SYSTEM DESCRIPTION

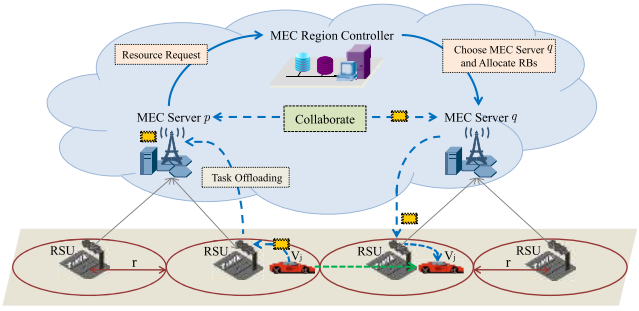
In this article, we mainly focus on the issue of how to allocate resources for IoV computing tasks to meet the high-level requirement of delay. The maximum delay acceptable to the task is the delay tolerance. We classify IoV computing tasks into different types according to their delay tolerances [22], [23], $type = \{1, 2, 3, 4\}$. The task type and the classification criterion are shown in Table 2.

- Type 1 is the urgent task with an extremely high-level requirement of the delay. Such as real-time traffic alerts, automated driver assistance, remote driving, and formation driving.
- Type 2 is the real-time task which mainly includes queries under human intervention, high definition real-time maps and adaptive cruising of vehicles.
- Type 3 is the general task with a wide-range delay tolerance, generated by entertainment, communication, and health IoV services.
- Type 4 is the best-effort task with a low-level requirement of the delay. Such as environmental monitoring, vehicle maintenance, and automotive software updates.

The details of task processing refer to Figure 3. We stipulate that when a task is generated, it is first offloaded to the nearest RSU and then to the MEC server which the RSU belongs to. And tasks are classified, marked and waiting in the MEC server deployed in the base station. Meanwhile, the MEC server uploads resource requests to the MEC Region controller. After accepting the request, the MEC region controller obtains the environment state from MEC servers and makes the resource allocation decision as soon as possible. The decision involves choosing the most appropriate MEC server and selecting some RBs in it. The specific decision-making mechanism is described in Section IV. Finally, the selected MEC server provides computing resources for calculating the task and passes the results back to the corresponding vehicle.



(a) The flow diagram.



(b) The schematic diagram.

FIGURE 3. The details of task processing.

C. NETWORK MODEL

We divide the entire vehicular network into R regions, and each region is organized by a MEC region controller. The set of MEC region controllers is denoted by $\mathbb{C} = \{1, \ldots, c, \ldots, C\}$. Each one MEC server is placed in one base station to provide computing services to the vehicles, and we set $\mathbb{M} = \{1, \ldots, m, \ldots, M\}$ to represent the number of base stations and MEC servers. We assume that a MEC server contains n computational resource units (RBs), $\mathbb{N} = \{n_1, n_2, \ldots, n_M\}$ is the set of the number of RBs in each MEC server in one region. The difference in the number of RBs reflects the difference in the computing ability of the MEC server.

D. COMMUNICATION MODEL

In our system, each vehicle is equipped with an on-board unit (OBU) containing signal sending and receiving devices. Besides, the RSU supports simultaneous communication with multiple vehicles. And the base station supports simultaneous communication with multiple RSUs. C-V2X (Cellular network based Vehicle to Everything) is a kind of cellular-based

wireless communication technology, which is adopted by the RSU and the OBU for communications. The RSU obtains computing resources of MEC servers through wired connections. In addition, communications between MEC servers and communications between MEC servers and the MEC region controller are also wired.

We consider the large-scale fading, such as Rayleigh fading, as well as the small-scale fading, such as the shadow fading and the path loss [24]. We assume that there are enough orthogonal wireless channels than the number of vehicles, so we ignore the interference between vehicles here. Moreover, we model the mobility problem of vehicles as the effect of position on the transmission rate shown in Figure 3(b). The Signal to Noise Ratio (SNR) and the transmission rate of the vehicle V_j are represented as:

$$SNR_j = \frac{|\eta_j|^2 \rho_j}{\delta^2}, \tag{1}$$

$$u_j = B\log_2(1 + SNR_j), \tag{2}$$

where, ρ is the transmitting power. $\eta_j = \sqrt{s_j} \cdot \eta_j'$ is the composite channel between V_j and the RSU. η_j' is the channel coefficient following the Rayleigh distribution with a mean of 0 and a variance of 1. $s_j = (d/r)^{-\xi} 10^{(\theta/10)}$ is equivalent to a small scale fading from V_j to the RSU, including the path loss and the shadow fading. d is the distance between V_j and the RSU. r is the reference distance. ξ is the path loss factor. θ is a normally distributed random variable with an average value of δ^2. The shadow fading conforms to the independent lognormal random distribution with the standard deviation δ. u_j is the transmission rate of the vehicle V_j, which is related to the channel bandwidth B and the SNR of the channel.

E. COMPUTING MODEL

It is the fact that the flow of tasks arriving at the MEC server and the requests arriving at the MEC region controller follow the Poisson distribution [25]. Due to the transmission delays of wire transmissions are negligible, the delay $D(i)$ of task i with type t is divided into four parts:

1). Delay of offloading to the RSU, $D_{upload}(i)$;
2). Delay of waiting, $D_{queue}(i)$;
3). Delay of calculating, $D_{compu}(i)$
4). Delay of passing back to the vehicle, $D_{return}(i)$.

We assume that the amounts of data and CPU cycles are similar for the same type of tasks. S_t^u and S_t^d are the amounts of data in offloading and downloading phases respectively. u_i^u and u_i^d are the transmission rates for offloading and downloading for task i. D_t indicates the number of CPU cycles needed to calculate the type t tasks. F_{RB} shows the computing ability that a RB can provide. n is the number of RBs allocated for task i by the deep-reinforcement-learning based resource allocation algorithm. K_i indicates the load state of the MEC server for calculating task i.

Both the waiting time of tasks in the MEC server and the sojourn time of resource requests in the MEC region controller can be obtained using the queuing theory M/M/1 model. Compared to the queuing time, the decision time of the controller is negligible. λ_m and λ_r are the arrival rates of data arriving at the MEC server and the MEC region controller, which indicate the arriving number of tasks or requests per second. β_m and β_r are the processing rates of the MEC server and the MEC region controller respectively.

All parts of delay can be calculated as follows (3) to (7):

$$D_{upload}(i) = \frac{S_t^u}{u_i^u}, \tag{3}$$

$$D_{queue}(i) = \frac{\lambda_m}{\beta_m(\beta_m - \lambda_m)} + \frac{\lambda_r}{\beta_r(\beta_r - \lambda_r)}, \tag{4}$$

$$D_{compu}(i) = \frac{D_t}{n \cdot F_{RB}} \cdot K_i, \tag{5}$$

$$D_{return}(i) = \frac{S_t^d}{u_i^d}, \tag{6}$$

$$D(i) = D_{upload}(i) + D_{queue}(i) + D_{compu}(i) + D_{return}(i). \tag{7}$$

The computing resources in our MEC based vehicular network are complex and variational. In addition, the distributions of vehicles and resource requests have certain regularities. It may be viable to take advantage of machine learning to learn the regularities of vehicles and requests. Consequently, we consider solving the resource allocation problem with deep reinforcement learning.

IV. PROBLEM FORMULATION AND SOLUTION

In this section, based on the above models, we first formulate the resource allocation problem as a Markov decision process by defining the state space, the action space, and the reward function. After that, we solve this problem with an intelligent resource allocation algorithm via deep reinforcement learning.

In our system, the resource allocation strategy is decided in the MEC region controller of each region. We discuss resource allocation problem in region G which contains m MEC servers.

A. STATE SPACE

Since the resource environment is changing, it is necessary to update the available resources in time. Our design of the state space at time instant t is as follows:

$$S(t) = \{w_i \in W, L(t), B(t)\}, \tag{8}$$

where

$$W = \{1, 2, 3, 4\}, \tag{9}$$

$$L(t) = [l_1, l_2, l_3, \ldots, l_m], \tag{10}$$

$$B(t) = [b_1, b_2, b_3, \ldots, b_m], \tag{11}$$

where, w_i is the type of task i that needs computing resources at time instant t. $L(t)$ records the load states of m MEC servers in region G. And $B(t)$ records the available numbers of RBs in each MEC server.

B. ACTION SPACE

The intelligent agent must select one of the MEC servers and some of RBs in that MEC server. We stipulate that RBs allocated for task i must be concentrated in the same MEC server. Furthermore, different types of tasks have different demands for delay. In case delay-insensitive tasks consume too much computing resources, we give the upper bound of the allocated RB numbers for each type, $upper = (4, 3, 2, 1)$. Thus, the action space can be represented as:

$$A(t) = \{mec_i \in \Gamma, rb\_sel(t)\}, \tag{12}$$

where

$$\Gamma = \{1, 2, 3, \ldots, m\}, \tag{13}$$

$$rb\_sel(t) = [rb_1, rb_2, rb_3, \ldots, rb_z]. \tag{14}$$

The action space includes two parts: mec_i and $rb\_sel(t)$. The former is the choice of MEC server for calculating task i. The latter is the choice of RBs in mec_i, and there are z RBs in mec_i. $rb_k \in \{0, 1\}$, when $rb_k = 1$ means k^{th} RB has already been selected at time instant t. Otherwise, $rb_k = 0$. The number of selected RBs must be less than or equal to the number of available RBs.

C. REWARDS AND PUNISHMENTS

The optimization goal is to minimize the delay $D(i)$ while satisfying the delay tolerances of all types of tasks. The $D(i)$ can refer to formula (3) to (7). The reward function $r(t)$ can be summarized as follows:

$$r(t) = \begin{cases} 1/D(i), & D(i) \le D_{tole}(w_i) \\ D_{tole}(w_i) - D(i), & D(i) > D_{tole}(w_i). \end{cases} \tag{15}$$

Due to large reward values are preferred, we set the reward value to calculate the inverse of the delay. For tasks that exceed the delay tolerance D_{tole}, we set the reward value to negative as the punishment.

D. Q-LEARNING TO DEEP Q-LEARNING

Q-learning is a typical reinforcement learning algorithm, which learns regularities and makes decisions through the interaction between a agent and the environment.

In a decision cycle, the agent senses environment state s_t. According to the action selection strategy π, the agent selects action a_t which transfers the current environment to a new state s_{t+1}. At the same time, the environment gives a feedback reward value r_t to the learning subject agent. Then at the next decision cycle, the agent and the environment interact in the same way.

Q-learning selects action-state $Q(s_t, a_t)$ as the value function, and evaluates the value function with the temporal difference method:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)), \tag{16}$$

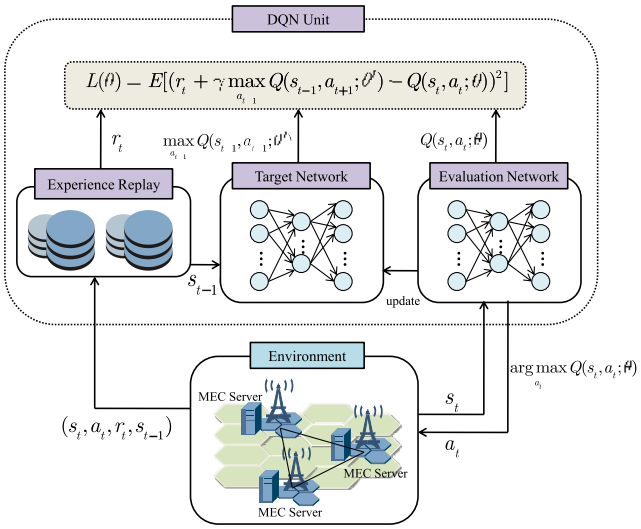


FIGURE 4. The detailed interaction of deep Q network.

where α is the learning efficiency, used to control convergence. γ is the discount factor which trades off the immediate reward and the long-term reward. In Q-learning, each $Q(s_t, a_t)$ is put into a corresponding position of the Q-table. It is possible for the agent to select the action a_t with the maximum Q-value, $a_t = \arg\max_{a_t} Q(s_t, a_t)$.

However, the MEC environment in our proposed system is complex and dynamically changing. It is not only impractical to store all values in one Q-table but also time-consuming to query a specific value in a large table frequently. Moreover, neural networks can be used to compensate for the Q-learning limitations in terms of generalization and function approximation capability [26]. Therefore, deep Q-Learning, which is the combination of reinforcement learning and deep learning, can be adopted into our multi-tires MEC based IoV system to deal with the optimization problem of resource allocation while maintaining QoS.

E. DQL BASED RESOURCE ALLOCATION ALGORITHM

Since deep Q-learning (DQL) introduces the neural network with parameters θ on the basis of Q-learning (QL), the action-state value has been changed from $Q(s_t, a_t)$ to $Q(s_t, a_t; \theta)$. In each DQL learning iteration, neural networks are trained to minimize the loss function, that is, minimize the deviation between the target Q-value and the evaluation Q-value. The loss function can be defined as:

$$L(\theta) = E[(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta') - Q(s_t, a_t; \theta))^2]. \tag{17}$$

The detailed interaction of deep Q network (DQN) as shown in Figure 4.

Compared with traditional QL, DQL has the following outstanding innovative features:

- The experience replay. The deep Q network stores experiences in the experience replay, and randomly selects batches of (s_t, a_t, r_t, s_{t+1}) to train neural networks.

Algorithm 1 The DQL Based Resource Allocation Algorithm

Input: discount rate γ, exploration rate ε, experience replay capacity P

Initialize the experience replay to capacity P.
Initialize the evaluation network with parameters θ.
Initialize the target network with parameters θ'.
for episode $k = 1 : K$ **do**
  Select a initial state s_t randomly.
  while $s_t != s_{goal}$ **do**
    Select action a_t based on ε-greedy policy and type constraints.
    Obtain the immediate reward value r_t and the next state s_{t+1}.
    Store experience (s_t, a_t, r_t, s_{t+1}) in the experience replay.
    Randomly selects batches of experiences from the experience replay.
    Calculate the target Q-value $Q_{target}(t)$:
    if $s_{t+1} != s_{goal}$
    $Q_{target}(t) = r_t$,
    else
    $Q_{target}(t) = r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta')$.
    Train the evaluation network to minimize loss function $L(\theta)$.
    Each C steps update the parameters of the fixed target network, $\theta' = \theta$.
    $s_t \leftarrow s_{t+1}$.
  end while
end for

Algorithm 2 Decision Procedure

Input: The trained evaluation network with parameters θ.

Initialize the environment state s_t.
for each decision cycle t **do**
  Decide action $a_t = \arg\max_{a_t} Q(s_t, a_t; \theta)$.
  Execute action a_t.
  Obtain reward r_t and s_{t+1}.
  $s_t \leftarrow s_{t+1}$.
end for

This mechanism disrupts temporal correlation by using historical experience for training.

- There are two identical neural networks in DQN, evaluation network and fixed target network. The evaluation network updates parameters θ in each training step to decrease the loss function. The parameters θ' of the fixed target network is updated by copying parameters from the evaluation network every stationary C steps.

Our DQL based resource allocation algorithm runs at the DQN unit of the MEC region controller. The algorithm can be divided into two stages, training and allocating. Each step of this algorithm is shown in Algorithm 1 and Algorithm 2. The former trains the neural networks, and the latter utilizes the trained neural network to make resource allocation decisions.

TABLE 3. Parameter settings in the simulation.

Parameters	Values
Task types	1, 2, 3, 4
The amount of offloaded data	{0.2, 1.5, 5.0, 10.0} Mbits
The amount of downloaded data	{0.1, 0.75, 2.5, 5.0} Kbits
The number of CPU cycles	$\{1, 7.5, 25, 50\} * 10^8$
The delay tolerances	{20, 150, 500, 1000} ms
The upper bound of RB numbers	4, 3, 2, 1
The number of MEC servers	3
The number of RBs in each MEC server	8, 6, 4
The computing ability provided by a RB	5 GHz
The arrival rates of requests and tasks	100
The processing rates of servers and controllers	200, 300
The coverage radius of a RSU	500 m
The number of RSUs under a base station	2
The path-loss exponent	3
The shadowing standard deviation	8 dBm
The noise power	0.01 W
The transmission bandwidth	10 MHz
The speed of vehicles	5 m/s
The learning effective factor	0.01
The learning discount factor	0.9
The epsilon-greedy factor	0.9

V. SIMULATION RESULTS AND ANALYSIS

In this section, we utilize computer simulation to evaluate the performance of proposed DQL-based resource allocation scheme in the MEC based vehicular network. We first describe the simulation settings and then present the simulation results.

A. SIMULATION SETTINGS

The experiments are conducted in the Windows operating system (CPU Intel core i7-7800x, 16GB of memory; GPU RTX 2080Ti, including 4352 CUDA and 11GB graphics memory; Python 3.7.5).

(1). Network framework: We use the MEC based regional intelligent management framework which we mentioned in Section III, including one MEC region controller, three base stations containing three MEC severs, and two RSUs under each base station.

(2). Parameter settings: The incoming resource requests at MEC region controller and computing tasks at MEC Servers obey Poisson distribution with same arrival rate. We assume that all RSUs have same transmission power 26dBm, and all vehicles have same transmission power 23dBm. Summarily, other values of parameters in the simulation are listed in Table 3.

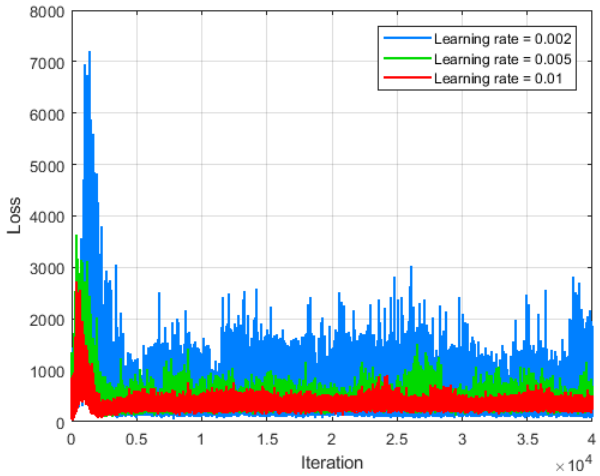


FIGURE 5. The training curves of different learning rates.

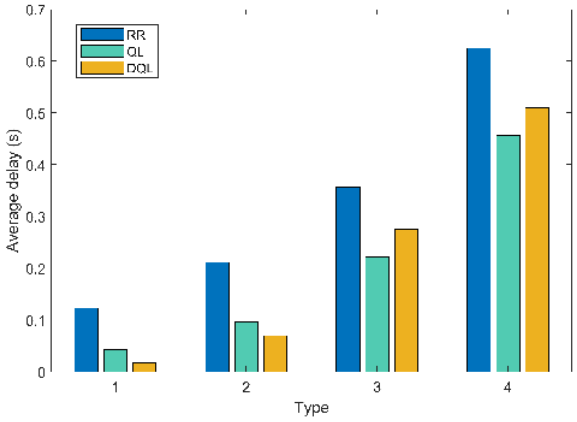
B. SIMULATION RESULTS

Figure 5 shows the convergence performance of the proposed DQL-based resource allocation algorithm when the learning rates are 0.01, 0.005, and 0.002. The training curves present a similar trend. In the beginning, the two neural networks have the same parameters, the loss value is very small. Then, as the evaluation network continues to learn and update parameters, the loss value starts to increase. Finally, due to the training takes effect, the loss value drops down. The reasons for the jitter of the curve can be summarized as the following two points. In the action selection, there is $1 - \varepsilon$ probability that the action is selected at random. In addition, the experience data used for training the network is generated by many single tasks, rather than many batch tasks.

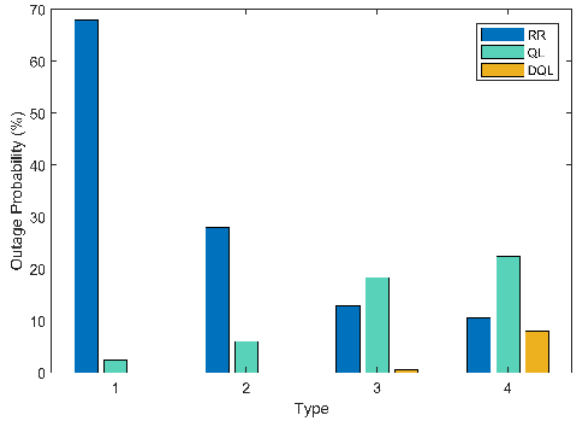
With the growth of training iterations, the loss value decreases gradually and tends to be stable, which proves that our algorithm has a certain convergence. Moreover, we found that the learning rate affects learning efficiency, it is very important to set an appropriate learning rate in the training process. A larger learning rate means a larger learning step, which may lead to miss the optimal solution. Conversely, a smaller learning rate slows down the speed of the convergence. As can be seen from Figure 5, the learning effect is the best when the learning rate is 0.01, and it converges around 3000 iterations. After training the neural networks in the DQN unit, we use them in the following simulation.

In order to demonstrate the delay performance of the proposed algorithm, we compared three resource allocation algorithms, as shown in Figure 6.

- RR: Round robin algorithm is a classic resource allocation algorithm, which is widely used in many scenarios.
- QL: A QoS enabled resource allocation algorithm based on Q-learning (reinforcement learning).
- DQL: The proposed DQL-based resource allocation algorithm, which is designed for processing IoV computing tasks with different delay sensitivities.



(a) The average delays of different types.



(b) The outage probabilities of different types.

FIGURE 6. The comparison of different resource allocation algorithms.

Figure 6(a) shows the average delays for completing different types of tasks. As can be seen from Figure 6(a), for both urgent tasks (type 1) and real-time tasks (type 2), the DQL algorithm successfully reduces the average delay significantly. Whereas, the average delays of the DQL algorithm for both general tasks (type 3) and best-effort tasks (type 4) are slightly higher than that of the QL algorithm. The reasons can be summarized as follows:

- Firstly, our optimization goal is not to blindly pursue the shortest average delay but to pursue the minimum delay under the premise of not exceeding the delay tolerances of four types. Hence, rather than causing important delay-sensitive tasks to time-out, it would be smarter to allow tasks with looser delay limitations to finish more slowly within the delay tolerance.
- Secondly, since the number of computing resources in our system is fixed, when multiple resource requests arrive at the controller at the same time, the agent inevitably trades off between the four types. Compared with QL, the DQL algorithm can more accurately and globally learn the differences of types. The above effect in Figure 6(a) proves that our DQL algorithm can

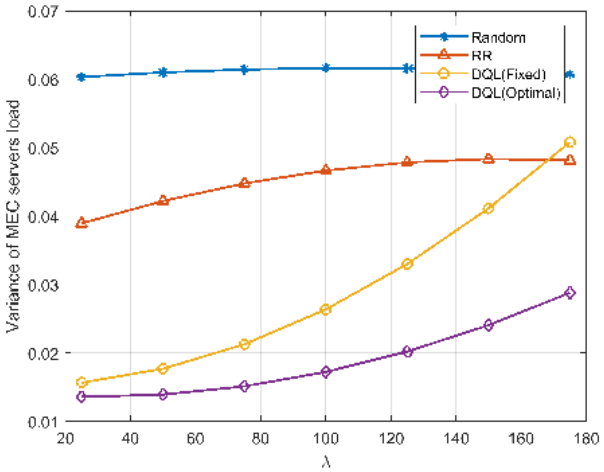


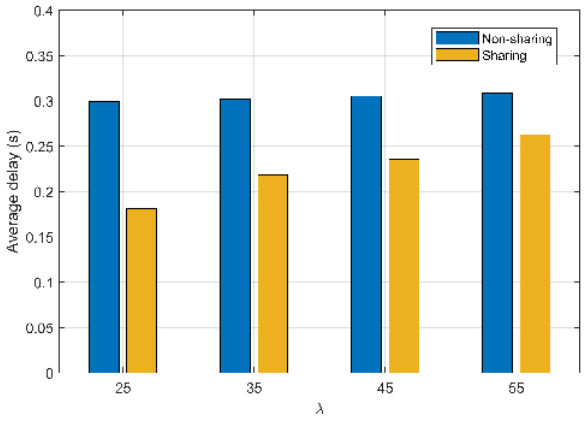
FIGURE 7. The effect of arrival rates λ on server load.

allocate limited resources more reasonably and intelligently, which is one of our contributions.

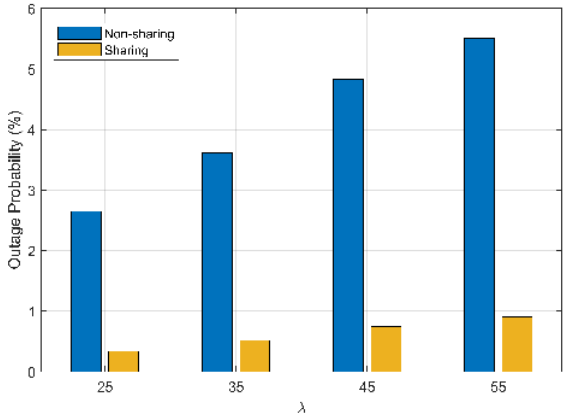
Figure 6(b) shows the outage probabilities for different types of tasks. When the delay tolerance is exceeded, the status of the task turns into the outage. Compared with other algorithms, the DQL algorithm reduces the outage probabilities of other types to a very low level at the cost of the best-effort tasks. The RR algorithm lacks intelligence in resource allocation due to its inability to perceive dynamically changing resource environments and different requirements of tasks. Therefore, the RR algorithm has the worst performance in terms of average delay and outage probability.

In our resource allocation mechanism, it involves the task migration between multiple MEC servers in a region. We use variances of MEC servers load to represent load differences as shown in Figure 7. With the increase of arriving rates, more tasks accumulate in servers, leading to the load variances grow. Random algorithm and RR algorithm fail to consider the load balancing, resulting in some servers are overloaded while other servers are idle, the variances are large. As we can see, the load on MEC servers under the DQL mechanism is more balanced. This is mainly because our method DQL(Optimal) choosing the optimal server by comprehensively taking the MEC environment and the delay requirements into consideration, instead of DQL(Fixed) blindly choosing the MEC server which the task belongs to or randomly (Random) or inflexible (RR) in selecting servers.

Moreover, we compare the proposed sharing mechanism with the non-sharing mechanism (servers without sharing) in Figure 8. For fairness, the numbers of servers and resources are exactly the same. Referring to the 8(a) and 8(b), the average delay and outage probability under both mechanisms increase with the increase of the arrival rate λ. In the non-sharing mechanism, a single server has limited resources, resulting in task congestions and long waiting delays. On the contrary, the sharing mechanism can make more flexible and reasonable use of all resources in the region, getting better



(a) The effect of arrival rates λ on average delay.



(b) The effect of arrival rates λ on outage probability.

FIGURE 8. The comparison of different sharing mechanisms.

performance in average delay and outage probability. This proves the feasibility of our DQL resource sharing mechanism in the proposed vehicular system with multiple MEC servers.

VI. CONCLUSION

In this article, we propose a novel MEC based regional intelligent management vehicular system, which can provide closer and more sufficient computing resources to reduce the delay. To solve the optimization problem of resource allocation for different types of IoV tasks, we model the Markov decision process by defining state space, action space and reward function. In addition, we design an intelligent and flexible resource allocation algorithm via deep Q-learning. Simulation results illustrate that the proposed algorithm could achieve better performance in terms of time efficiency, outage probability and load balance, which indicate the feasible and effective of the proposed system. Although deep reinforcement learning has certain advantages in solving above problems, it is highly dependent on the training data. Once the data changes, the state transition rule also changes, the network needs to be retrained. Future works are going to consider how to address this challenge.

REFERENCES

[1] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Mobile edge computing and networking for green and low-latency Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 39–45, May 2018.
[2] M. Li, F. R. Yu, P. Si, and Y. Zhang, "Green machine-to-machine communications with mobile edge computing and wireless network virtualization," *IEEE Commun. Mag.*, vol. 56, no. 5, pp. 148–154, May 2018.
[3] Z. Junhui, Y. Tao, G. Yi, W. Jiao, and F. Lei, "Power control algorithm of cognitive radio based on non-cooperative game theory," *China Commun.*, vol. 10, no. 11, pp. 143–154, Nov. 2013.
[4] J. Zhao, X. Guan, and X. P. Li, "Power allocation based on genetic simulated annealing algorithm in cognitive radio networks," *Chin. J. Electron.*, vol. 22, no. 1, pp. 177–180, Jan. 2013.
[5] J. Wang, L. Zhao, J. Liu, and N. Kato, "Smart resource allocation for mobile edge computing: A deep reinforcement learning approach," *IEEE Trans. Emerg. Topics Comput.*, to be published.
[6] J. Li, H. Gao, T. Lv, and Y. Lu, "Deep reinforcement learning based computation offloading and resource allocation for MEC," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6.
[7] S. Wang, C. Pan, and C. Yin, "Joint heterogeneous tasks offloading and resource allocation in mobile edge computing systems," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process.*, Oct. 2018, pp. 1–6.
[8] J. Zhou, F. Wu, K. Zhang, Y. Mao, and S. Leng, "Joint optimization of offloading and resource allocation in vehicular networks with mobile edge computing," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process.*, Oct. 2018, pp. 1–6.
[9] C.-M. Huang, M.-S. Chiang, D.-T. Dao, W.-L. Su, S. Xu, and H. Zhou, "V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) architecture," *IEEE Access*, vol. 6, pp. 17741–17755, 2018.
[10] B. Dab, N. Aitsaadi, and R. Langar, "A novel joint offloading and resource allocation scheme for mobile edge computing," in *Proc. 16th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2019.
[11] J. Zhao, S. Ni, L. Yang, Z. Zhang, Y. Gong, and X. You, "Multiband cooperation for 5G HetNets: A promising network paradigm," *IEEE Veh. Technol. Mag.*, vol. 14, no. 4, pp. 85–93, Dec. 2019.
[12] S. Zhou, P. P. Netalkar, Y. Chang, Y. Xu, and J. Chao, "The MEC-based architecture design for low-latency and fast hand-off vehicular networking," in *Proc. IEEE 88th Veh. Technol. Conf.*, Aug. 2018, pp. 1–7.
[13] H. Wang, X. Li, H. Ji, and H. Zhang, "Dynamic offloading scheduling scheme for MEC-enabled vehicular networks," in *Proc. IEEE/CIC Int. Conf. Commun. (ICCC Workshops)*, Aug. 2018, pp. 206–210.
[14] L. Zhang and B. Cao, "Stochastic programming method for offloading in mobile edge computing based Internet of vehicle," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.
[15] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated offloading scheme to minimize latency in MEC-enabled vehicular networks," in *Proc. IEEE Globecom Workshops*, Dec. 2018, pp. 1–6.
[16] D. Wang, H. Qin, B. Song, X. Du, and M. Guizani, "Resource allocation in information-centric wireless networking with D2D-enabled MEC: A deep reinforcement learning approach," *IEEE Access*, vol. 7, pp. 114935–114944, 2019.
[17] F. Xu, H. Ye, F. Yang, and C. Zhao, "Software defined mission-critical wireless sensor network: Architecture and edge offloading strategy," *IEEE Access*, vol. 7, pp. 10383–10391, 2019.
[18] C. Qiu, H. Yao, F. R. Yu, F. Xu, and C. Zhao, "Deep Q-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5871–5883, Jun. 2019.
[19] H. Ye and G. Y. Li, "Deep reinforcement learning for resource allocation in V2V communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.
[20] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.
[21] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.
[22] Y. Chen, K. Yan, M. Song, Q. Liu, and Y. Xue, "Analyisis and application of 5G V2X bussicess based on MEC," *Designing Techn. Posts Telecommun.*, no. 11, pp. 80–85, Nov. 2018.
[23] *White Paper on C-V2X Use Cases: Methodology, Examples and Service Level Requirements*. Accessed: Jun. 19, 2019. [Online]. Available: http://www.5gaa.org
[24] X. Huang, G. Wang, S. Zhou, X. Tang, and R. Yao, "V2R communication power allocation scheme for VANETS," *J. THz Sci. Electron. Informat. Technol.*, vol. 16, no. 3, pp. 516–521, Jun. 2018.
[25] X. Wang, Z. Ning, and L. Wang, "Offloading in Internet of vehicles: A fog-enabled real-time traffic management system," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4568–4578, Oct. 2018.
[26] F. Shah-Mohammadi and A. Kwasinski, "Deep reinforcement learning approach to QoE-driven resource allocation for spectrum underlay in cognitive radio networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, May 2018, pp. 1–6.

GE WANG received the B.S. degree in software engineering from the Beijing University of Technology, China, in 2016. She is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications. Her current research interests include edge computing and the Internet of Vehicles.

FANGMIN XU received the M.S. and Ph.D. degrees in communication engineering from the Beijing University of Posts and Telecommunication (BUPT), China, in 2003 and 2008, respectively. From 2008 to 2014, he was with Samsung Electronics, where he actively contributed to 3GPP LTE/LTE-A and IEEE 802.16m. He is currently an Associate Professor with the School of Information and Communication Engineering, BUPT, China. He is the author of two books, 20 peer-reviewed international research articles, 50 standard contributions, and the inventor of 15 issued or pending patents among which four have been adopted in the specifications of 4G (3GPP LTE/LTE-A and IEEE 802.16m) standards. His research interest includes advance technologies in wireless networks, especially the Internet of Things (IoT) field.