Task offloading algorithm of vehicle edge computing environment based on Dueling-DQN

Yi OUYANG 1*

1 School of Information Engineering, Jiangxi University of Science and Technology, Ganzhou, Jiangxi 341000, China
*Corresponding author’s e-mail: 6720180565@mail.jxust.edu.cn

Abstract: With the wide application of the Internet of vehicles, the rapid development of intelligent vehicles provides drivers and passengers with a good driving and riding experience. However, how to process a large amount of data messages in real time on resource-limited vehicle terminals is still a huge challenge, and it will cause great energy consumption to terminal devices. In this paper, a semi-online task distribution and offloading algorithm based on Dueling-DQN is proposed for time-varying complex vehicle environments. Firstly, the vehicle offloading system is built by the reinforcement learning, because the original optimization problem is a joint optimization problem with high complexity, and divided into the vehicle scheduling sub-problem and the vehicle computation resources optimization sub-problem. We predicts different vehicle offloading behaviour, and calculates total rewards value after a series of vehicle offloading action, so as to update the vehicle offloading decisions. The simulation results show that this algorithm can improve the efficiency and energy consumption of computation tasks to a certain extent.

1. Introduction
With the rapid development of the ubiquitous intelligent transportation system and the rise of the vehicular networks [1,2], the European Telecommunications Standards Association and the 5G Automobile Alliance have proposed artificial intelligence vehicle applications, by interconnecting its objects and devices, namely vehicle-to-everything (V2X)[3] and virtual methods expand the current vehicular networks, which attracts the widespread attention. However, car applications need to process its own surrounding environment information (location, speed, images, etc.), and the amount of data is very large and complex. The existing vehicle cloud computing has quite powerful computation capabilities, storage space and network resources, which is conducive to vehicle information efficient services between sharing and related businesses. Because cloud servers are usually deployed in data center far away from users, the time for offloading tasks delivered to the cloud and the return time of results are seriously increased. Coupled with the rapid growth of the number of vehicles in the vehicular networks and the large number of vehicle devices accessing the cloud computing center, as a result, the centralized big data processing model on cloud computing cannot provide low-latency and highly reliable services. For such "low latency, high bandwidth, high reliability" application scenarios, introducing Mobile Edge Computing (MEC) [4] technology into the vehicular networks is a good choice to solve this problem. MEC can better reduce the bandwidth pressure on the core network. By deploying computation resources, network control functions and cached data near the data center on urban roads, it greatly reduces the data backhaul link and the processing time on the backbone network and the mobility Energy consumption of equipment.
Usually in a wireless fading environment, the time-varying channel state greatly affects the optimal offloading decision of the car system to a certain extent. Especially in many vehicles scenario, the biggest obstacle is how to optimize the computation mode and resource allocation. In recent years, the above two issues have attracted the attention of many researchers. The authors in Reference [5] study the problem of intelligent task scheduling and resource allocation in the MEC vehicular networks scene, propose a two-sided matching scheme to solve the task scheduling problem, and use a deep learning method to deal with the network resource allocation problem. Authors in Reference [6] regards each vehicle-to-vehicle (V2V) link as an agent, and proposes a distributed resource allocation method based on deep reinforcement learning, and interacts with unknown environments to learn and improve resource sharing strategies. Authors in Reference [7] considers multiple concurrent tasks with different priorities, and solves the limitation of uneven vehicle load through genetic algorithm based task offloading strategy, and improves the success rate of car safety tasks. Authors in Reference [8] study the task offloading delay problem in the vehicle edge computing system, and proposes an adaptive learning task offloading algorithm based on the multi-armed bandit (MAB) to encourage vehicles to learn the offloading delay performance between adjacent vehicles while minimize task offloading delay. Authors in Reference [9] discusses the computation power and high-efficiency performance required by computationally intensive vehicle applications, and proposes a joint offloading and resource allocation method to maximize the system network utility and balance load distribution. Authors in Reference [10] first establishes a stacked auto-encoding traffic flow prediction model, then applies a novel deep learning method, and takes into account the temporal and spatial distribution correlation, and finally proposes a greedy layered unsupervised learning scheme for pre-training deep neural networks, while greatly shortening the training time.

In the process of computational offloading, the large amount of data and more complex data content make traditional machine learning techniques have some shortcomings in dealing with such problems. For example, for a large number of data feature extraction, a large number of iterations are required to reach the local optimum [11]. Especially in a dynamic environment with fast fading, computational offloading strategies cannot be formulated in real time. Many existing deep learning methods need to optimize all system parameters at the same time. The algorithm has high complexity and long iteration steps, which is not suitable for the delay-sensitive vehicular networks environment. Therefore, for the MEC vehicular networks computational offloading system, this paper proposes a computation task distribution offloading algorithm based on the dueling deep Q network (Dueling-DQN) [12]. The original optimization problem is decomposed into the vehicle task scheduling sub-problem and the vehicle computation resource optimization sub-problem, and by learning the previous offloading experience under different wireless fading conditions, a distributed semi-online computation model is proposed. In the continuous state space, by predicting different vehicle behaviours only need to be selected from a few candidate actions at a time, and its vehicle offloading action generation strategy is automatically improved, which reduces energy consumption to a certain extent and the processing time of resource allocation sub-problems, while maximizing the total vehicle computational offloading efficiency.

2. System Model
As shown in Figure. 1, the system structure we consider includes one central Base Station (BS) equipped with abundant computing resources, $b$ vehicles equipped with MEC exchange information with BS by 5G/LTE, and $r$ Road Side Units (RSU) equipped with large MEC servers are evenly distributed on urban roads. In the case of ensuring high-quality communication, the vehicle communicates with the RSU through Dedicated Short range Communication Technology (DSRC). Each vehicle is equipped with a dedicated antenna, we assume that the wireless power transmission and computational offloading between V2X are performed in the same sub-band, and can avoid the above two Interfere with each other by Orthogonal Frequency Division Multiplexing (OFDM).
2.1 Communication Model
With the increasing number of task offloading, the traditional communication network model inevitably leads to the problem of high latency. In response to this problem, we have cited cellular V2X technology [13], in which the vehicle is equipped with a pool of radio resources that they can automatically select V2V communications. However, limited spectrum resources cannot efficiently and completely handle the offloading problem, unless the vehicle occupying the channel that releases the channel resource, otherwise vehicles cannot allocate the channel. In order to solve this problem, the vehicle resource pool can overlap with the cellular vehicle-to-infrastructure (V2I) interface for making better use of spectrum resources. Using OFDM can convert frequency selective wireless channels into multiple parallel channels on different subcarriers. We assume that the channel fading is roughly the same in the same sub-band and is independent of each other among different sub-band. Therefore, the signal to interference plus noise ratio (SINR) received by each V2V link during the task offloading process can be expressed as:

$$\text{SINR}_i^k = \frac{p^k h_i}{\sigma^2 + \sum_{i \neq k} p^i h_i}$$  \hspace{1cm} (1)$$

Where $p^k$ represents the energy transmission power between the vehicle and the edge device by wireless network BS. $\sigma^2$ is the noise power, $h_i$ is the channel power gain from the $k$-th vehicle to the BS/RSU, and $\hat{h}_i$ is the interference power gain between users of each vehicle. Therefore, the vehicle offloading capacity is equivalent to the processing rate of the task, as shown in Equation (2):

$$C_k = B \log_2 (1 + \text{SINR}_i^k)$$ \hspace{1cm} (2)$$

Where $B$ is the remaining bandwidth of the current system, assuming $S_i^k$ is the amount of data transmitted by the vehicle, then the task transmission time can be calculated by:

$$\gamma_i^k = \frac{S_i^k}{C_k}$$ \hspace{1cm} (3)$$

2.2 Computation Model

2.2.1 Local Computation Mode
In the local computing mode $m_0$, the vehicle can simultaneously supplement energy to the vehicle MEC equipment and compute its offloading tasks. It is assumed that task offloading only considers its computation delay, and the output data size of task offloading is much smaller than the input data size. Then, the transmission delay between the two and the backhaul transmission delay can be ignored. We define the computational capability of the vehicle MEC equipment as $F_\epsilon \beta$, $\epsilon_i^k$ denotes the computation resources required for the $i$-th task of $k$-th vehicle, and the local computation time is $T_i^{\epsilon,k} = F_\epsilon \beta / \epsilon_i^k$. The power consumption of vehicle MEC equipment is $F_\epsilon^3 \beta$. Where $\beta_i$ represents the computation energy
efficiency coefficient of the vehicle MEC device chip, therefore, the local computation energy consumption is limited to:

\[ E_L = \beta_L F_L T_{L}^{m,k} \leq E_{\text{max}} \]  

(4)

2.2.2 Edge Computation Mode

Due to the limitation of time division multiplexing, the vehicle MEC equipment offloads tasks to the BS/RSU after collecting energy, which is referred to as mode \( m \) for short. In fact, computation power and transmission power of BS/RSU are more than two orders of magnitude higher than that of vehicle MEC equipment [14]. Compared with offloading tasks to BS/RSU, the computation offloading results are downloaded to the vehicle MEC device in a shorter time. \( \varepsilon_{\text{eff}}^{l} \) denotes the remaining computation resources of the edge server, thus, the edge server computation time is:

\[ T_{E}^{m,k} = F_{L} / \varepsilon_{\text{eff}}^{l} \]  

(5)

In order to maximize the offloading efficiency, we also ignore the energy consumption of receiving the computation result from the BS/RSU, and only consider the energy consumption of data transmission. In this case, the vehicle MEC equipment is required to exhaust its collected energy with best performance. Therefore, the total energy consumption in the two modes can be expressed as:

\[ E_{\text{Total}} = \sum_{l=1}^{L} T_{L} (m_{k} P_{l} + (1 - m_{k}) \beta_{L} F_{L}^{l}) \leq E_{\text{max}} \]  

(6)

Where \( P_{l} \) represents edge device transmission power and \( m_{k} \in \{0,1\} \).

3. Semi-online task distribution offloading algorithm based on Dueling-DQN(SODO)

3.1 Problem Formulation

At present, many researchers use traditional intelligent optimization algorithms to solve edge computing problems, but the time-varying vehicle environment under fast fading channels is not suitable. In the actual vehicle edge computing scenario, the diversification of vehicle tasks lead to different computation resources requirement. The vehicle state space will also be increased exponentially with the increasing number of vehicles. Different edge devices will also have insufficient computation resources or excessive load that affect the success rate of computation tasks. In this article, in order to solve these complex problems and offload tasks between different vehicles and edge devices with higher performance, a semi-online task distribution offloading algorithm based on Dueling-DQN is proposed, which can autonomously be used without human intervention, update its offloading decision, optimize system computation resources, reduce energy consumption and network delay. Therefore, the optimization goal of this paper is to maximize the number of computation resources under the constraints of total server resources, and minimize the energy consumption during task offloading, as shown in Equation (7).

\[
\text{Goal:} \max \sum_{l=1}^{L} \frac{T_{L}^{m,k}}{E_{\text{Total}}} \\
\text{s.t.:} \begin{cases} 
C_1: \sum_{i} \varepsilon_{i}^{l} \leq \varepsilon_{\text{max}} \\
C_2: \sum_{i} C_{i}^{l} \leq C_{\text{max}} \\
C_3: \sum_{i} B_{i}^{l} \leq B_{\text{max}} 
\end{cases}
\]  

(7)

Where \( C_1 \), \( C_2 \) and \( C_3 \) respectively denote that the required computation resources, vehicle offloading capacity, and vehicle terminal transmission bandwidth cannot exceed the upper limit.

3.2 Algorithm Overview

Before solving the above problem, this section mainly briefly describes the SODO algorithm in the MEC vehicular networks scenario. The main framework of the intelligent offloading system is shown
in Figure. 2, which mainly includes vehicle offloading action generation and vehicle offloading behaviour prediction. In the $t$-th time frame, vehicle observes the unknown environment and records the real environment state $S_t$ by the observation function $D_t^{(b)} = O(S_t, b)$, including system resource allocation rate, channel allocation rate, and task offloading completion degree. DQN takes the current observation $D_t^{(b)}$ as input, and outputs a vehicle offloading action $A_t$ based on the current offloading strategy $\pi : S_t \rightarrow A_t$. Then quantify it as a binary offloading action, and select an optimal offloading action based on the final $Goal(S_t, A_t)$. The intelligent offloading system network adopts the best offloading action $A_t^*$, obtains a reward $R^*(S_t, A_t^*)$, and adds the newly obtained state action pair $(S_t, A_t)$ to the replay memory.

Subsequently, a small batch of training samples are extracted from the replay memory unit for sampling, and the strategy $\pi$ is updated by predicting the vehicle behaviour. According to the latest observation $D_t^{(b)}$ and strategy $\pi_{t+1}$, which will be used in the next time frame to generate the next offloading decision $A_{t+1}^*$. After the repeated iterations, the offloading decision $\pi_t$ of DQN is gradually improved. The state transition and rewards are stochastic, which are usually modelled as Markov Decision Process (MDP) [15,16], in which the state transition probability and reward only depend on the current environment state, and the action is determined by the vehicle. Therefore, it can be represented by the state transition probability matrix $P = \{p(S_{t+1}, R^*|S_t, A_t^*)\}$. The generation of vehicle offloading action and the prediction of offloading behaviour will be described in more detail later.

3.3 Vehicle Offloading Action Generation

In the $t$-th time frame, each V2V link receives an environment observation $D_t^{(b)}$, where $t = 1, 2, \cdots$. When $t = 1$, the randomly initialized environment state is determined by the initial transmission power of all vehicle links and channel states, and the parameter $\theta$ of DQN follows a zero-mean normal distribution [17]. Each V2V link has a dedicated DQN and takes the current observation value $D_t^{(b)}$ as input, and outputs a computation offloading action $A_t^*$, the parameter Equation can be expressed as:

$$A_t^* = \{A_{t,j}^* \in [0, 1], j = 1, 2, \cdots, M\} \quad (8)$$

Where $A_{t,j}^*$ represents the offloading action of the $j$-th vehicle under the current offloading decision $\pi_t$. Our offloading model is binary offloading, in order to better generate binary offloading decisions, firstly, the offloading actions are quantified into $M$ candidate actions, and the number 0 is filling in $n$ digits from the high order to obtain a 0-1 sequence of length $n$, the real-time decision information will be sent to the edge device to decide whether to offload. Therefore, we can obtain the optimal vehicle offloading action in the $t$-th time frame by Equation (9):
The above Equation (9) can speed up the process in parallel, and then outputs the offloading actions related to the best computation resources and energy consumption accordingly.

3.4 Vehicle Behaviour Prediction

When a series of environmental conditions are taken into consideration, the problem will become more complicated. Therefore, under the premise of not requiring any labelled data, by predicting the difference of vehicle behaviour [18], an improved MEC vehicle networking scenario algorithm is designed. Due to the different behaviours of vehicle and with the increasing number of vehicles, the vehicle offloading space will be increased rapidly. Usually we consider the vehicle with better behaviour and abundant computation resources. For each task offloading, when the computation resources are overloaded, the system will receive a punitive reward \( R'(S_t, A'_t) \), which is the negative value of the absolute value for the current system reward, as shown in Equation (10):

\[
R'(S_t, A'_t) = -\left| R(S_t, A'_t) \right|
\]  

Therefore, we use the total reward value \( R_{\text{total}} \) of the system as the prediction of the vehicle offloading behaviour to better measure the pros and cons of future vehicle behaviour and the choice between local mode and edge mode, which is beneficial to update offloading decision.

\[
R_{\text{total}} = \sum_{t=1}^{T} (R'(S_t, A'_t) + R'(S_t, A'_t))
\]

At the \( t \)-th time frame, the replay memory unit is initialized, and a new training sample data \( (S_t, A'_t) \) is added to the replay memory unit. Due to its limited capacity, when it is fully loaded, the newly generated training data will replace the previous data. We use the experience replay technique to train multiple V2X training sample data, and uniformly sample a batch of the training data \( (S_t, A'_t) \) from the replay memory unit randomly. In the structure of the neural network, the stochastic gradient descent algorithm (Adam) [19] is used to reduce the average cross-entropy loss. Its main advantage is that after bias correction, each iterative learning rate has a certain range, which makes the parameters relatively stable.

In this paper, as the reinforcement learning algorithm gradually learns the optimal vehicle offloading behaviour, the performance of the entire system is improved, and on the basis of satisfying the minimum delay, \( R_{\text{total}} \) is maximized under the current optimal offloading decision. Therefore, the above summary of the related MEC vehicular networks scenario pseudocode algorithm is described as follows:

---

Algorithm 1: Semi-Online Task Distribution Offloading With Dueling-DQN

Input: Current environment state observation \( D_t^{(0)} \)

Output: Optimal vehicle offloading action \( A'_t \)

1: Start environment simulator and generate vehicles and V2X links
2: Initialize experience replay memory unit and \( Q(S_t, A'_t) \)
3: for each episode do
4: for each V2V agent do
5: Observe environment state \( D_t^{(0)} \)
6: Generate vehicle offloading action \( A'_t \) according to offloading decision \( \pi \)
7: end for
8: Quantify vehicle offloading action into a binary 0-1 sequence
9: Receive reward \( R_{\text{total}} \) according to Vehicle Behaviour Prediction
10: Choose vehicle offloading action \( A'_t \) according to \( \arg \max_{A'_t} \text{Goal}(h_t, A'_t) \)
for each V2V agent do
  Observe the next environment state $D_{t+1}^{(b)}$
  Store new action pair $(S_t, A_t)$ and update experience replay memory unit
  update network parameters using the Adam algorithm
end for

4. SIMULATION RESULTS
In this section, simulation results are presented to validate the proposed SODO for vehicular networks scenario, and we custom built our simulator according to the Manhattan city case evaluation method defined in 3GPP TR 36.885 [20]. Among them, vehicles are randomly distributed on the road according to the spatial Poisson process, and each vehicle can communicate with neighbour vehicles. Without loss of generality, it is assumed that the channel state gain remains unchanged in the same time frame and is independent of each other in different time frames.

Our DQN is a four-layer fully connected neural network with two hidden layers. Here, we use Relu as the activation function in the hidden layer, i.e., $f_g(x) = \max(0,x)$, such that the relaxed offloading action $A^\pi_t$ satisfies $A^\pi_t \in (0,1)$. In the initial stage, the learning rate is set to 0.001 and decreases exponentially. The training data was mainly performed by Adam. The detail parameters can be found in Table 1.

| Parameter                  | Value                          |
|----------------------------|--------------------------------|
| Task data size             | 64kb-100kb                     |
| Vehicle transmit power     | 1.3W                           |
| Vehicle computation power  | 4W-8W                          |
| Total bandwidth            | 60 Mbps                        |
| Vehicle speed              | 36 km/h                        |
| Noise power                | -144 dBm                       |
| Vehicle numbers            | 10-100                         |
| Per task running time      | 25ms                           |
| Vehicle computation capacity | $4 \times 10^6$ round / s   |
| MEC computation capacity   | $8 \times 10^7$ round / s     |

Based on the Q learning network framework, the traditional DQN method uses the value function to evaluate the selection of actions. In general, this method tends to choose the action that maximizes the reward value for the next step. However, its action selection strategy is greedy, which will inevitably lead to overestimation of the Q value. In order to better solve this problem, Hasselt [21] proposed a deep double Q network (DDQN), which uses different value functions to evaluate the selection of actions, and then uses a neural network to evaluate the selection strategy and approximate the true value. The target Q value function can be expressed as:

$$ Y_t^{DDQN} = R_{t+1} + \gamma Q(S_{t+1}, \arg\max_a Q(S_{t+1}, \alpha; \theta), \theta^-) $$

(12)

With the increasing number of vehicles, the vehicle state space increases exponentially, which will inevitably bring complexity problems. Due to Dueling-DQN pays more attention to the relationship between core system states and actions, it can reduce computational complexity and solve the problem of state space dimensions. Dueling-DQN divides the Q network into two parts, the first part is only related to the state $S$ and unrelated to the adopted action $A$. It is called the value function part and denoted as $V(S, \omega, \alpha)$. The second part is related to both the state $S$ and the action $A$, this part is called the advantage function part, denoted as $A(S, A, \omega, \beta)$. In the end, value function can be re-expressed as:
\[ Q(S, A, \omega, \alpha, \beta) = V(S, \omega, \alpha) + (A(S, A, \omega, \beta) - \frac{1}{|A|} \sum_{a \in A} A(S, a', \omega, \beta)) \] (13)

Where \( \omega, \alpha \) and \( \beta \) is the neural network parameter.

In order to find out the reasons for the excellent performance of different learning strategies, we compare the SODO with DDQN, local computation and greedy algorithm respectively, and investigate the probability that the vehicle satisfies the delay constraint under different vehicle numbers.

It can be seen from Figure. 3 that as the number of vehicles increases, V2V links are also increasing. Therefore, it is difficult to ensure that each vehicle can satisfy the time delay constraint. The local mode has the fastest decline rate and the worst performance among the above algorithms. Because a single vehicle MEC device does not have enough energy nor can it handle a large amount of task data. The traditional DDQN method is not the best algorithm, because this method cannot pay attention to the relationship between the core system state and actions, which leads to an increase in the complexity of the problem. The greedy algorithm does not fully consider the dynamic changes of the network state, and the SODO can reduce the complexity, the dimension of the state space and overcome the overestimation problem. Therefore, there is a greater probability that the vehicle can satisfy the delay constraint condition.

![Figure 3](image)

Figure. 3 Probability of satisfied vehicle terminal delay constraint versus the number vehicles

Figure. 4 shows the comparison of the maximum computation rate of the vehicle task with different offloading strategies. The result of the exhaustive method can be regarded as the upper bound, and SODO can reach 98% of the upper bound performance. Compared with the greedy algorithm, which is higher than 20%, this is because the greedy algorithm, in contrast to reinforcement learning, always chooses the action with the largest reward value at the current moment, and the intensification of resource competition. Compared with the random algorithm higher than close to 50%, it is obvious that the random algorithm is the worst performance among the above four algorithms. Although the exhaustive method can reach the upper bound, it is not practical due to its high complexity. In short, the performance of SODO is similar to the exhaustive method, and is better than the greedy algorithm and the random algorithm.
Finally, we study the impact of different offloading algorithms on the energy consumption of the vehicle task as the number of vehicles increases. As shown in Figure 5, SODO maintains the lowest energy consumption during the whole task offloading process, and the random algorithm maintains the highest energy consumption. When the number of vehicles is 20, the energy consumption of SODO only accounts for 33.6%, 36.8% and 22.7% of the DQN, DDQN, and random algorithm respectively. This is because the distributed offloading model adopted in this paper improves the task processing efficiency. DQN cannot overcome the interference of overestimation. Even though DDQN overcomes the problem of overestimation, due to the interference of high spatial complexity. In short, SODO can maximize performance while greatly reducing energy consumption.

5. Conclusion
In this paper, SODO is used to solve the problem of computation resource allocation for the diversity of vehicle offloading behaviours. The algorithm is a distributed task offloading, and there is no need to know the global information in each offloading decisions, reduces the transmission overhead,
converges after a limited iteration, and obtains a better offloading strategy. It can be seen from the simulation results that SODO maintains good performance while reducing energy consumption and improving the efficiency of computational offloading. In the future, we will consider more complex and dynamic environments and characteristics of vehicle security services in accordance with actual vehicular networks scenarios.

Acknowledgments

This work is partially supported by Jiangxi Provincial of China Department of Education Science and Technology Project Funding(GJJ180443)

Reference

[1] Qi Q, Ma Z. Vehicular edge computing via deep reinforcement learning[J]. arXiv preprint arXiv:1901.04290, 2018.
[2] XIAO Y. Research on computation offloading strategy for 5G low latency and high reliability in-vehicle services[D]. Xidian University, 2019.
[3] GU X H, ZHANG G A. Survey of mobile edge computing applications in vehicular networks[J]. Application Research of Computers, 2020, 37(06): 1615-1621.
[4] XIE R C, LIAN X F, JIA Q M, et al. Overview of mobile edge computing offload technology[J]. Journal of Communications, 2018, 39(11):138-155.
[5] Ning Z, Dong P, Wang X, et al. Deep reinforcement learning for vehicular edge computing: An intelligent offloading system[J]. ACM Transactions on Intelligent Systems Technology, 2019, 10(6): 1-24.
[6] Ye H, Li G Y, Juang B-H F. Deep reinforcement learning based resource allocation for V2V communications[J]. IEEE Transactions on Vehicular Technology, 2019, 68(4): 3163-3173.
[7] YU X, LIU Y X, SHI X Q, et al. Offloading strategy for mobile edge computing in Vehicle Network[J]. Computer Engineering, 2020:1-8.
[8] Sun Y, Guo X, Song J, et al. Adaptive learning-based task offloading for vehicular edge computing systems[J]. IEEE Transactions on vehicular technology, 2019, 68(4): 3061-3074.
[9] Dai Y, Xu D, Maharjan S, et al. Joint offloading and resource allocation in vehicular edge computing and networks[C]. 2018 IEEE Global Communications Conference (GLOBECOM), 2018: 1-7.
[10] Chen X, Jiao L, Li W, et al. Efficient multi-user computation offloading for mobile-edge cloud computing[J]. IEEE/ACM Transactions on Networking, 2015, 24(5): 2795-2808.
[11] GOU Y, LI J M, WEI X. Deep learning of IOT and task unloading scheduling strategy for edge computing[J]. Computer Applications and Software, 2019, 36(08): 125-129.
[12] Wang Z, Schaul T, Hessel M, et al. Dueling network architectures for deep reinforcement learning[C]. International conference on machine learning, 2016: 1995-2003.
[13] Liang L, Ye H, Li G Y. Spectrum sharing in vehicular networks based on multi-agent reinforcement learning[J]. IEEE Journal on Selected Areas in Communications, 2019, 37(10): 2282-2292.
[14] Wang F, Xu J, Wang X, et al. Joint offloading and computing optimization in wireless powered mobile-edge computing systems[J]. IEEE Transactions on Wireless Communications, 2017, 17(3): 1784-1797.
[15] Liu Y-J, Cheng S-M, Hsueh Y-L. eNB selection for machine type communications using reinforcement learning based Markov decision process[J]. IEEE Transactions on Vehicular Technology, 2017, 66(12): 11330-11338.
[16] Wei Y, Yu F R, Song M. Distributed optimal relay selection in wireless cooperative networks with finite-state Markov channels[J]. IEEE Transactions on Vehicular Technology, 2010, 59(5): 2149-2158.
[17] Yan J, Bi S, Zhang Y J A. Offloading and resource allocation with general task graph in mobile edge computing: a deep reinforcement learning approach[J]. IEEE Transactions on Wireless Communications, 2020, 19(8): 5404-5419.

[18] Tang L, He S. Multi-user computation offloading in mobile edge computing: A behavioral perspective[J]. IEEE Network, 2018, 32(1): 48-53.

[19] Ruder S. An overview of gradient descent optimization algorithms[J]. arXiv preprint arXiv:04747, 2016.

[20] Chen S, Hu J, Shi Y, et al. Vehicle-to-everything (V2X) services supported by LTE-based systems and 5G[J]. IEEE Communications Standards Magazine, 2017, 1(2): 70-76.

[21] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning[C]. Proceedings of the AAAI Conference on Artificial Intelligence, 2016.