Joint Task Offloading and Service Migration in RIS assisted Vehicular Edge Computing Network Based on Deep Reinforcement Learning

Xiangrui Ning, Ming Zeng, Zesong Fei
School of Information and Electronics, Beijing Institude of Technology
Beijing, China
mzengzm@bit.edu.cn

***Abstract*—To address the increasingly complex computing tasks of intelligent vehicles, we consider a framework for Reconfigurable intelligent surface (RIS) assisted vehicular edge computing (VEC) networks. We aim to maximize the weighted sum throughput of all vehicular user equipments (VUEs) while limiting the latency of all VUEs in each time slot to a certain range by jointly optimizing computational edge servers for all VUEs, the deployment location of the RIS and its reflecting beamforming matrix. We propose a deep reinforcement learning (DRL) based algorithm to solve the problem. Evaluation results show the effectiveness of the proposed algorithm and verify that RIS deployment is a valid solution to enhance the communication and computation in VEC network.**

***Index Terms*—reconfigurable intelligent surfaces, vehicular edge computing, service migration, optimization, Parametrized Deep Q-Network (PDQN).**

I. Introduction

With the development of the 5th Generation (5G) communication network and mobile edge computing (MEC) paradigm, intelligent vehicles have the ability to use sensing and computing capabilities to utilize information about the surrounding environment, such as making overall arrangements for communication and computing resources through cooperation between vehicular user equipments (VUEs) and edge servers (ESs), so as to achieve a significant improvement in system energy efficiency and communication quality [1]. Vehicles can use edge resources through task offloading, and ESs achieve active balancing of the computing requirements of offloading tasks through service migration, thereby greatly improving the performance of intelligent driving systems. [2] optimized migration strategies between base stations (BSs) by jointly managing computing-and-radio resources to maximize total offload rates, quantify MEC throughput, and minimize migration costs. [3] jointly studied the service migration and mobility in vehicular edge computing (VEC) with the road traffic assignment. The author proposed a deep reinforcement learning based algorithm to maximize the comprehensive utility of communication, computing, and traffic assignment.

Reconfigurable Intelligent Surface (RIS) can be used to implement intelligent reconfigurable wireless channels and radio propagation environments for 5G wireless communication. RIS is an intelligent surface that includes a large number of reflective units, each of which is able to control the incident signal via passive beamforming. Therefore, RIS can improve the performance of MEC systems by reconfiguring the wireless environment, thereby promoting information transmission and increasing offloading rate. [4] introduced a communication system that performs machine learning tasks on a MEC server using RIS to maximize throughput. By jointly optimizing the transmit power of the users, the beamforming vectors of the BSs, and the phase shift matrix of the RIS, a framework iterative optimization solution based on alternating optimization was proposed. [5] considered a RIS assisted MEC system that improves the transmission rate through RISs. The optimization problem was proposed to optimize the upload bandwidth allocation for each user's task data.

Despite the growing body of literature on RIS-assisted MEC and vehicular communication networks, the research on RISassisted VEC remains comparatively scarce. This is largely due to the time variation of channels caused by the mobility of VUEs, which poses significant challenges to the joint optimization of task offloading, service migration, and RIS reflection beamforming matrices. Motivated by these discussions, we consider a framework for RIS assisted VEC networks to maximize the weighted sum throughput of all VUEs while limiting the latency of all VUEs in each time slot to a certain range by jointly optimizing computational ESs for all VUEs, the deployment location of the RIS and its reflecting beamforming matrix. We evaluate the convergence of the proposed Parametrized Deep Q-Network (PDQN) based algorithm and validate the effectiveness of it by comparing its performance with two other schemes under different cases.

The rest of this paper is organized as follows. Section II introduces the system model and problem formulation in VEC network. In Section III, an efficient algorithm is proposed to solve the formulated problem in Section II. Section IV presents numerical results to evaluate the performance of the proposed designs. Finally, we conclude the paper in Section V.

II. System Model and Problem Formulation

In this section, we formulate the joint optimization problem of task offloading and service migration. As shown in Fig. 1, we consider a RIS aided VEC system which is comprised

This work is supported by National Natural Science Funds of China (Grant No.62001028).

of a set BSs integrating ESs $\mathcal{E} = \{1, 2, \ldots, E\}$, a RIS and a set of VUEs with single antenna $\mathcal{K} = \{1, 2, \ldots, K\}$. The number of the BSs' antennas and the RIS's reflecting elements are represented by N_t and N_r, respectively. The ESs are able to balance the load between them through backhaul link connections. RIS is deployed to enhance the communication in the VEC network. We assume that each VUE runs various applications and generate computational tasks for each time slot $t \in \mathcal{T}$ as a time unit, where $\mathcal{T} = \{1, 2, \ldots, T\}$. In time slot t, the tasks of each VUE can be transmitted to the local BS through task offloading and computed by any but only a BS with integrated ES of the whole VEC networks through service migration and backhaul link between BSs.

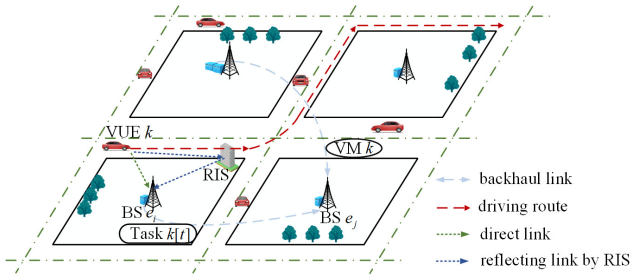

Fig. 1. The RIS aided VEC system

Each VUE is served by a dedicated virtual machine (VM), and the ES that hosts the corresponding VM of the VUE can provide computing services to it. VM is defined as a software clone of the VUE's service environment which contains the VUE's profile and the application that runs the VUE's offloading tasks [2]. It is assumed that each VM provides services only for the corresponding individual VUE, i.e., the VUE and VM are in one-to-one correspondence.

A. Service Migration

To reduce the service latency, VMs should be hosted on the ES integrated with the local BS of the VUEs or nearby. The VUE sends service requests to the local ES through the relevant BS. If the local ES hosts the corresponding VM, it will accept the requests, otherwise it will forward the requests to the ES hosting the corresponding VM via backhaul link. Therefore, to meet the service latency of the VUE, VMs should follow the mobility of the corresponding VUEs and migrate dynamically between ESs.

Let $x_{i,k}[t] \in \{0, 1\}$ denote whether VUE $k \in \mathcal{K}$ is associated with BS and ES $e_i \in \mathcal{E}$ and $y_{j,k}[t] \in \{0, 1\}$ denote whether the VM of k is hosted on ES $e_j \in \mathcal{E}$ in time slot $t \in \mathcal{T}$, where $\mathcal{T} = \{1, 2, \ldots, T\}$. In a time slot, each VUE can only be associated with one BS and its VM can only be hosted on one ES, thus satisfying

$$\begin{aligned} \sum\nolimits_{e_i \in \mathcal{E}} x_{i,k}[t] = 1, \ \forall k \in \mathcal{K}, \\ \sum\nolimits_{e_j \in \mathcal{E}} y_{j,k}[t] = 1, \ \forall k \in \mathcal{K}. \end{aligned} \tag{1}$$

The migration delay is caused by the migration of VMs from the ES integrated with BS $e_{j'} \in \mathcal{E}$ in the previous time slot $t-1$ to the ES integrated with BS $e_j \in \mathcal{E}$ in the current time slot t. The migration delay increases with the service VM data size and migration distance [6], which is defined as

$$D^M_{j',j,k}[t] = \begin{cases} 0, & j' = j, \\ \vartheta^s_k / \eta^{mig} + \sigma^{mig} d_{j',j}, & j' \neq j, \end{cases} \tag{2}$$

where ϑ^s_k is the size of the migrated service of VUE k, $d_{j',j}$ is the length of the service migration path between $e_{j'}$ and e_j, η^{mig} is the network bandwidth along the migration path, σ^{mig} is a positive coefficient.

B. Task Offloading

In time slot $t \in \mathcal{T}$, let $\mathbf{g}_i[t] \in \mathbb{C}^{N_t \times N_r}$, $\mathbf{h}^R_k[t] \in \mathbb{C}^{Nr \times 1}$ and $\mathbf{h}^D_{k,i}[t] \in \mathbb{C}^{Nt \times 1}$ denote the channel gain from RIS to BS e_i, the channel gain from VUE k to RIS and the channel gain from VUE k to BS e_i, respectively. In addition, it is assumed that the power of signal reflected more than twice by RIS can be ignored due to high path loss. We adopt a quasi static flat fading channel model, assuming that the channel state information (CSI) of all channels is completely known. We assume that the communication distance of a single link is d, the path loss model can be expressed as

$$L(d) = C_0 d^{-\alpha}, \tag{3}$$

where C_0 is the frequency-dependent pass loss, α is the path loss exponent, which usually takes a value between 2 and 6. Moreover, the phase shift $\theta_n \in [0, 2\pi)$ and the reflection amplitude $\beta_n \in [0, 1]$ applied to the incident signal by the nth reflection unit of the RIS are denoted. Moreover, the phase shift and the reflection amplitude applied to the incident signal by the nth reflection unit of the RIS are denoted by $\theta_n \in [0, 2\pi)$ and $\beta_n \in [0, 1]$. Consequently, we have $\mathbf{\Theta} = diag(\beta_1 e^{j\theta_1}, \cdots, \beta_{Nr} e^{j\theta_{Nr}})$. To simplify the calculation, $\beta_n[t] = 1$ is usually taken. We assumed that all VUEs offload the tasks on a fixed frequency band B during time T. In time slot t, we define the offloading signal and the transmission power of VUE k as $\mathbf{x}_k[t]$ and p_k, respectively. The received signal $\mathbf{y}_m[t] \in \mathbb{C}^{N_t \times 1}$ at BS $e_m \in \mathcal{E}$ is given by

$$\mathbf{y}_m[t] = \sum_{k \in \mathcal{K}} x_{m,k}[t](\mathbf{h}^D_{k,m}[t] + \mathbf{g}_m[t]\mathbf{\Theta}[t]\mathbf{h}^R_k[t])\mathbf{x}_k[t] + \mathbf{cn}_m, \tag{4}$$

where $\mathbf{cn}_m \sim \mathcal{CN}(0, \sigma^2_m)$ denotes the additive white Gaussian noise (AWGN), and $\mathcal{CN}(0, \sigma^2_m)$ is the circular symmetric complex Gaussian distribution with zero mean and σ^2_m variance.

After processing by a beamforming vector $\mathbf{w}_k[t] \in \mathbb{C}^{N_t \times 1}$, which satisfies $|\mathbf{w}_k[t]||^2 = 1$, the estimated signal for VUE k can be written as

$$\hat{\mathbf{x}}_k[t] = \sum\nolimits_{e_j \in \mathcal{E}} x_{i,k}[t] \mathbf{w}^H_i[t] \mathbf{y}_i[t], \tag{5}$$

Accordingly, when we know that $x_{i,k}[t] = 1$, the signal to interference and noise ratio (SINR) and the offloading throughput of VUE k at time slot t is expressed as

$$SINR_{i,k}[t] = \frac{p_k \left| \mathbf{w}^H_k[t] \mathbf{h}_k[t] \right|^2}{\sum\limits_{k' \in \mathcal{K}, k' \neq k} x_{i,k'}[t] p_{k'} \left| \mathbf{w}^H_k[t] \mathbf{h}_{k'}[t] \right|^2 + \sigma^2_i}, \tag{6}$$

$$R_{i,k}^{off}[t] = B\log_2(1 + SINR_{i,k}[t]), \tag{7}$$

where $\mathbf{h}_k[t] = \sum_{e_m \in \mathcal{E}} x_{m,k}[t](\mathbf{h}_{k,m}^D[t] + \mathbf{g}_m[t]\mathbf{\Theta}[t]\mathbf{h}_k^R[t]), \forall k \in \mathcal{K}$.

Given the local BS for VUE k, the coordinates of the RIS and its reflecting beamforming matrix $\mathbf{\Theta}$, the solution of $\mathbf{w}_k[t]$ for VUE k can be computed in closed-form by [7]

$$\mathbf{w}_k[t] = \frac{\left(\mathbf{I}_{N_t} + \sum_{k' \in \mathcal{K}} \frac{p_{k'} x_{i,k'}[t]}{\sigma_i^2} \mathbf{h}_{k'}[t]\mathbf{h}_{k'}^H[t]\right)^{-1} \mathbf{h}_k[t]}{\left\| \left(\mathbf{I}_{N_t} + \sum_{k' \in \mathcal{K}} \frac{p_{k'} x_{i,k'}[t]}{\sigma_i^2} \mathbf{h}_{k'}[t]\mathbf{h}_{k'}^H[t]\right)^{-1} \mathbf{h}_k[t] \right\|_2}, \tag{8}$$

Let $\vartheta_k^o[t]$ denote the size of the offloading task of VUE k in time slot t, the offloading latency is obtained as

$$D_{i,k}^T[t] = \vartheta_k^o[t] / R_{i,k}^{off}[t], \tag{9}$$

C. Task Backhaul

When the computation ES of VUE k is different from its local ES, the tasks and results need to be backhauled. The backhaul latency depends on the length of task backhaul path and the data size of tasks [6], which is expressed as

$$D_{i,j,k}^B[t] = \begin{cases} 0, & i = j, \\ \vartheta_k^o[t]/\eta^b + \sigma^b d_{i,j}, & i \neq j, \end{cases} \tag{10}$$

where $x_{i,k}[t] = 1$, $y_{j,k}[t] = 1$, $d_{i,j}$ is the length of the task backhaul path between e_i and e_j, η^b is the network bandwidth along the backhaul path, σ^b is a positive coefficient.

D. Task Computing

Each ES accommodates VUEs' offloaded tasks into their own VMs and executes them in parallel [2]. Considering the Input/Output (I/O) interference in parallel computing, we use the computational model in [8] to describe the computational throughput. When $y_{j,k}[t] = 1$, the computational throughput and the computational latency for performing the computational task of VUE k are expressed as

$$R_{j,k}^{com}[t] = f_{j,k}(1 + d_j)^{1 - \sum_{k \in \mathcal{K}} y_{j,k}[t]}, \tag{11}$$

$$D_{j,k}^C[t] = \vartheta_k^o[t] / R_{j,k}^{com}[t], \tag{12}$$

where $f_{j,k}$ denotes the expected computation rate of the VM of VUE k when running in isolation at the ES integrate with BS e_j, $d_j > 0$ is the performance degradation factor of the ES integrate with BS e_j to specify a decrease in the computation rate of a VM when reused with another VM.

E. Problem Formulation

According to the definitions above, the weighted sum throughput and total delay of VUE k in time slot t are denoted as

$$R_k[t] = \sum_{e_i \in \mathcal{E}} \sum_{e_j \in \mathcal{E}} x_{i,k}[t] y_{j,k}[t] [\varpi R_{i,k}^{off}[t] + (1 - \varpi) R_{j,k}^{com}[t]], \tag{13}$$

$$D_k[t] = \sum_{e_i \in \mathcal{E}} \sum_{e_j \in \mathcal{E}} x_{i,k}[t] y_{j,k}[t] \Big[\max\{D_{j',j,k}^M[t], D_{i,k}^T[t]\} + D_{j,k}^C[t] + D_{i,j,k}^B[t] \Big], \tag{14}$$

where ϖ is the weight parameter of offloading throughput. Due to the small amount of result data, the downlink delay of returning results from ESs to VUEs is negligible compared to the uplink delay of offloading tasks from VUEs to ESs.

We consider a problem that jointly optimizes task offloading and service migration in the proposed VEC network. We aim to maximize the weighted sum throughput of all VUEs across the communication and computing network in $\mathcal{T}$ while limiting the latency of all VUEs in each time slot to a certain range. Therefore, it is necessary to select the appropriate computational ES for all VUEs, the deployment location of the RIS and its reflecting beamforming matrix. As such, the optimization problem is formulated as

$$\begin{aligned} (P) \ & \max_{\mathbf{\Theta}, \mathbf{Y}, \mathbf{c}_r} \sum_{t \in \mathcal{T}} \sum_{k \in \mathcal{K}} R_k[t], \\ s.t. \ & D_k[t] \leq D^{\max}, \\ & y_{j,k}[t] \in \{0, 1\}, \ \forall k \in \mathcal{K}, \ \forall e_j \in \mathcal{E}, \\ & \sum_{e_j \in \mathcal{E}} y_{j,k}[t] = 1, \forall k \in \mathcal{K}, \\ & \theta_1[t], \theta_2[t], \ldots, \theta_{N_r}[t] \in [0, 2\pi), \ \mathbf{c}_r \in \mathcal{C}_r, \end{aligned} \tag{15}$$

where $D^{\max}$ is the maximum latency that VUEs can tolerate, $\mathbf{c}_r$ is the deployment location of the RIS, $\mathcal{C}_r$ denotes the discrete set of all possible deployment locations for the RIS. $\mathbf{Y} = [[\mathbf{y}[1]], [\mathbf{y}[2]], \ldots, [\mathbf{y}[T]]]$ denotes the choice of computational ES for all VUEs, where $\mathbf{y}[t] = [\mathbf{y}_1[t], \mathbf{y}_2[t], \ldots, \mathbf{y}_K[t]], \forall t \in \mathcal{T}$, $\mathbf{y}_k[t] = [y_{k,1}[t], y_{k,2}[t], \ldots, y_{k,E}[t]], \ \forall k \in \mathcal{K}$.

Obviously, it is challenging to obtain the optimal solution to (P), since the problem is non-concave because $\mathbf{\Theta}$, $\mathbf{Y}$ and $\mathbf{c}_r$ are coupled. To avoid the complex derivations and transformations of traditional convex optimization methods, we can model the problem as a Markov Decision Process (MDP) model and solve it by deep reinforcement learning (DRL) algorithms.

III. PDQN-based Joint Task Offloading and Service Migration Optimization

In this section, the MDP model is formulated and an optimization algorithm based on PDQN [9] is proposed to solve (P).

A. Markov Decision Process Model

To solve the established optimization problem using DRL first requires converting the formulated problem into an MDP model, where the key elements are defined as follows.

The MDP has five basic components $< \mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma >$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, r is the immediate reward obtained from the environment, $\mathcal{P}$ is the transition probability, and $\gamma \in [0, 1]$ is the discounted factor.

1) State space: Since all BSs with ESs are able to communicate with each other using backhaul links and the delay in transmitting optimization decisions between BSs within each time slot is much smaller than the total latency of any VUE, the BS associated with the most number of servers is able to

play the role of agent. We define the state space of the agent in t as

$$s[t] = \{x_{i,k}[t], R_k[t], D_k[t] | \ \forall e_i \in \mathcal{E}, \forall k \in \mathcal{K}\} \in \mathcal{S}, \quad (16)$$

2) Action space: we define the action space of the agent in time slot t as

$$a[t] = \{\mathbf{\Theta}[t], \mathbf{c}_r[t], \mathbf{y}_k[t] | \forall k \in \mathcal{K}\} \in \mathcal{A}, \quad (17)$$

3) Reward function: considering that the objective function of (P) is to maximize the weighted sum throughput of all VUEs while the objective of MDP is to maximize long-term cumulative rewards, we define the reward function of the agent in time slot t as

$$r[t] = \begin{cases} \sum\limits_{k \in \mathcal{K}} R_k[t], & if \ D_k[t] \leq D^{\max}, \forall k \in \mathcal{K}, \\ -1, & otherwise. \end{cases} \quad (18)$$

B. Proposed Algorithm Based on PDQN

Considering that the action space of MDP contains both continuous and discrete variables, it is not possible to directly use the common DRL algorithm as we need to preprocess the current hybrid action space. Therefore, we propose a PDQN based optimization algorithm. The PDQN-based algorithm combines the actor-critic structure and deep neural network (DNN), which consists of three sections, namely, a Parameterized Actor network, a Q Actor network and a replay memory. Besides, Parameterized Actor and Q Actor networks consist of two DNNs i.e., an online network and a target network, separately. The algorithm based on PDQN combines actor-critic structure and fully connected neural network, including Parameterized Actor network, Q Actor network, and the replay memory. In addition, both Parameterized Actor and Q Actor networks are composed of online network and target network. Base on the proposed algorithm, we split $a[t]$ into $a^{param}[t] = \{\mathbf{\Theta}[t], \tilde{\mathbf{y}}_{i,k}[t] | \forall k \in \mathcal{K}\} \in \mathcal{A}^{param}$ and $a^d[t] = \{\mathbf{c}_r[t]\} \in \mathcal{A}^d$, where $\tilde{\mathbf{y}}_{i,k}[t]$ is the relaxation value of the discrete variable $\mathbf{y}_{i,k}[t]$. There are only two discrete values for $\mathbf{y}_{i,k}[t]$, so there is a condition for continuity, and it needs to be rounded based on $\tilde{\mathbf{y}}_{i,k}[t]$ in subsequent calculation. The detailed procedure of PDQN-based algorithm is shown as follows.

Initially, based on equation (16) to obtain environment state information, the online Parameterized Actor network is able to output parameterized actions, which is described as

$$a^{param}[t] = \mu\left(s[t] | \theta\right) + \mathcal{N}, \quad (19)$$

where θ is the parameters of the online Parameterized Actor network. By adding Gaussian noise $\mathcal{N}$ to the selected action, the exploration can be realized [10].

The online Q Actor network is used to calculate the state-action value function $Q\left(s, a^{param}, a^d\right)$ to evaluate the effect of continuous actions selected by the online Parameterized Actor network and select appropriate discrete actions based on the evaluation value, namely

$$a^d[t] = \arg\max_{\mathcal{A}^d} Q(s[t], a^{param}[t], a^d | \omega), \quad (20)$$

where ω is the parameters of the online Q Actor network. By performing the current actions $a^{param}[t]$ and $a^d[t]$ in the VEC network and updating the vehicle running trajectory, we can obtain the instant reward $r[t]$ and the next states $s[t+1] = s'[t]$ according to Equations (16) and (18). After that, the sequence $(s[t], a^{param}[t], a^d[t], r[t], s'[t])$ is saved into the replay memory $\mathcal{D}$ for subsequent training of online network. In order to eliminate the correlation between samples for training, a mini-batch of samples with the size of Z is usually randomly selected from $\mathcal{D}$.

The target networks are used to generate target values for updating online network parameters as follow

$$\begin{aligned} \theta' &\leftarrow \tau\theta + (1-\tau)\,\theta', \\ \omega' &\leftarrow \tau\omega + (1-\tau)\,\omega', \end{aligned} \quad (21)$$

where θ' and ω' are the parameters of the target Parameterized Actor network and the target Q Actor network, $\tau \ll 1$ is the soft replacement parameter to ensure the high stability in training [10].

The online Parameterized Actor network is updated by policy gradient (PG) method [11] with respect to the actor parameters θ, which is expressed as

$$\begin{aligned} \nabla_\theta L^\mu = \mathop{\mathbb{E}}_{z \in \mathcal{Z}} [\nabla_\theta \mu\left(s|\theta\right)|_{s=s_z} \times \\ \nabla_{a_{param}} Q\left(s, a^{param}, a^d|\omega\right)|_{s=s_z, a^{param}=\mu(s_z|\theta), a^d=a_z^d}] \end{aligned} \quad (22)$$

where $\mathcal{Z}$ stands for the mini-batch dataset and z represents the mini-batch sample index.

The online Q Actor network is updated by minimizing the loss function L^ω with the stochastic gradient descent (SGD) method shown as follows

$$L^\omega = \mathop{\mathbb{E}}_{z \in \mathcal{Z}} \left[\left(y_z - Q\left(s_z, a_z^{param}, a_z^d|\omega\right)\right)^2\right], \quad (23)$$

where y_z is the current target Q value based on the target Q Actor network which is given by

$$y_z = r_z + \gamma \max Q'\left(s_z, \mu'\left(s_z'|\theta'\right), a_z^d|\omega'\right). \quad (24)$$

Therefore, the proposed PDQN-based joint optimization algorithm for task offloading and service migration is summarized in Algorithm 1.

IV. Performance Evaluation

In this section, we use numerical simulation to show the performance of the proposed PDQN-based joint optimization algorithm in our RIS-aided VEC network. Since the proposed algorithm adopts DRL framework, we implement the simulation in the PyTorch framework with Python 3.8.

A. System Parameters

For simulation scenario setup, we assume that the width of all streets is 20m, the side length of all square blocks is 500m, and the coverage of each BS is 368m. BSs are deployed on the center of the square blocks. RIS can be deployed in the edge area of the BS service area and the optional deployment locations include the corner position of the street intersection

Algorithm 1 PDQN-Based Joint Optimization Algorithm for Task Offloading and Service Migration.

Randomly initialize online Parameterized Actor network and online Q Actor network with θ and ω;
Initialize θ' and ω' by $\theta' \leftarrow \theta, \omega' \leftarrow \omega$;
Initialize discounted factor γ, replay memory $\mathcal{D}$, mini-batch size Z, soft replacement parameter τ;
for $episode = 1$ to $Iterations$ **do**
  Initialize a random process $\mathcal{N}$;
  Receive initial states $s[0]$ from the environment;
  for $t = 0$ to T **do**
    select parameterized actions $a^{param}[t] = \mu(s[t])+\mathcal{N}$;
    select discreate actions $a^d[t] = \arg\max_{\mathcal{A}^d} Q(s[t], a^{param}[t], a^d|\omega)$;
    Execute $a^{param}[t]$ and $a^d[t]$ and updating the vehicle running trajectory to obtain $r[t]$ and $s[t+1] = s'[t]$;
    Store $(s[t], a^{param}[t], a^d[t], r[t], s'[t])$ in $\mathcal{D}$;
    Update the states: $s[t] \leftarrow s[t+1]$;
    Sample a random mini-batch samples $(s_z, a_z^{param}, a_z^d, r_z, s'_z)$ from $\mathcal{D}$;
    Update online Q Actor network with SGD by (23) and (24);
    Update online Parameterized Actor network with PG by (22);
    Update target networks by (21).
  end for
end for

and the midpoint of two corner positions. For the BS-RIS, RIS-VUE and BS-VUE links, we adopt α^{BR}, α^{RU}, α^{BU} to be 2.2, 2.4 and 3.8, respectively. Furthermore, it is assumed that the BS-RIS is the line-of-sight (LOS) channel, the BS-VUE and RIS-VUE links are the Rayleigh fading channel. In order to conform to the actual design, we maintain the deployment location of the RIS unchanged during an epoch, that is, the fixed deployment location of the RIS for this round is the one that has the most occurrences of all the theoretical deployment locations output during the previous training round.

Besides, we set the travel speed range, the size of offloading tasks and the service need to be migrated of VUE as [10,12.5) m/s, [500, 1000) bits and [5000,10000) bits, respectively. We assume that $f_{i,k} \in [2, 0.5, 0.5, 0.5]$ Mbps, $d_i = 0.25$, $\eta_{mig} =$ 5Mbps, $\eta_b = 5$Mbps, $\sigma_{mig} = 1\mu$s/m, $\sigma_b = 0.2\mu$s/m, $\varpi =$ 0.7. Other parameters related to the environment are provided in Table I while the parameters related to PDQN are shown in Table II.

B. Performance Comparisons

Based on the above simulation scenario and parameters, we evaluate the convergence of the proposed algorithm. From Fig. 2, it can be observed that the cumulative reward of the RIS assisted VEC network converges around 1000 iterations and remains stable in subsequent training.

TABLE I
PARAMETERS RELATED TO THE ENVIRONMENT

Parameters	Explanation	Value
D_{th}	Maximum latency that VUEs can tolerate	0.2s
N_t	Number of BS receiving antennas	8
N_r	Number of RIS reflecting units	16
K_l	Number of VUEs	8
C_0	Path loss	-30dB
σ_i^2	Noise variance	-80dBm
t	Unit time slot size	1.0s
K	Number of VUEs	8
p_k	Transmission power of VUE k	0.2W

TABLE II
PARAMETERS RELATED TO PDQN TRAINING

Parameters	Explanation	Value
$layers$	Number of hidden layers	2
γ	Discounted factor	0.99
α_{param}, α_q	Learning rate	0.0002
τ	Soft replacement parameter	0.01
B	Replay memory size	1e6
Z	Mini-batch size	256
$activation$	Activation function	Relu, Tanh
$units$	Number of neuros	400, 300
T	Update steps in an epoch	100

To verify the effectiveness of the proposed algorithm, we use two comparative schemes – the Deep Deterministic Policy Gradient (DDPG) based scheme [10] and the NRIS scheme. The difference between DDPG and PDQN lies in the processing of discrete action space. The former chooses to scale to continuous variables, while the latter chooses to use Q-network for discrete learning. The two comparative schemes are as follows.

1) **DDPG-based scheme:** the action becomes $a[t] = \{\tilde{\mathbf{c}}_r[t], \tilde{\mathbf{y}}_{i,k}[t] | \forall k \in \mathcal{K}\} \in \mathcal{S}, \forall t$, where $\tilde{\mathbf{c}}_r[t]$ is the relaxation value of the discrete variable $\mathbf{c}_r[t]$ which needs to be rounded based on $\tilde{\mathbf{c}}_r[t]$ in subsequent calculation.
2) **NRIS scheme:** there is not a RIS to assist the VEC network. Therefore, all elements about the RIS in (P) are removed, the action space becomes $a[t] = \{\tilde{\mathbf{y}}_{i,k}[t] | \forall k \in \mathcal{K}\} \in \mathcal{S}, \forall t$ and the algorithm structure keeps the same as that in DDPG-based scheme.

The relationship between cumulative rewards and the number of RIS reflecting beamforming units is shown in Fig. 3. When the number of RIS reflected beamforming units increases from 4 to 24, the RIS reflected beamforming units will not affect the NRIS scheme. Compared with the MAPPO-base scheme in VEC network without multi-RIS assistance, the proposed scheme has a 18.1% increase in the system throughput. Besides, the performance improvement of the proposed algorithm compared to the NRIS scheme is 249% of that of the DDPG-based scheme compared to the NRIS scheme when the number of RIS reflective units is 16. The results show the advantages of deploying more RIS reflecting beamforming units. That is because the RIS has the capability to improve signal propagation with high transmission rate, so

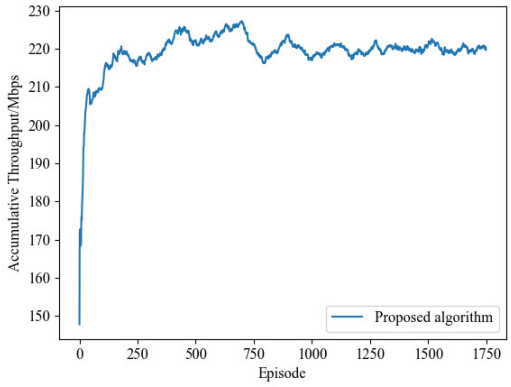

Fig. 2. Convergence of the accumulated rewards.

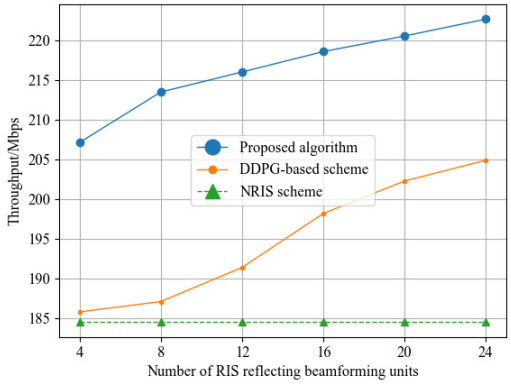

Fig. 3. Comparison of accumulated rewards with different numbers of RIS reflecting beamforming units.

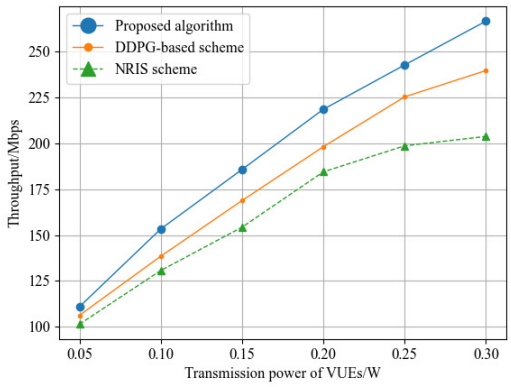

Fig. 4. Comparison of accumulated rewards with different transmission power of VUEs.

adding RIS reflecting units can improve system performance.

Fig. 4 shows the relationship between the accumulated rewards and the transmission power of VUEs. With the increase of the transmission power of VUEs from 0.05W to 0.30W, the effectiveness of all three schemes has been improved, but the cumulative rewards of the proposed scheme are significantly higher than the other two schemes. For instance, when $p_k = 0.15$W, our proposed scheme has a 20.4% increase in the accumulated rewards compared with the NRIS scheme, and the performance improvement of the proposed algorithm compared to the NRIS scheme is 217% of that of the DDPG-based scheme compared to the NRIS scheme. The comparison between the proposed scheme and the NRIS scheme demonstrates the advantages of RIS in rate improvement and delay reduction.

V. Conclusion

In this paper, with the objective of maximizing the weighted sum throughput of all VUEs, a throughput-maximization problem was studied in the RIS assisted VEC network. To solve the problem, we propose a PDQN based algorithm jointly optimizing the deployment location of the RIS and its reflecting beamforming matrix. Simulation results demonstrate that our RIS assisted VEC network is able to achieve higher sum throughput than the conventional VEC network without the deployment of RIS, and the proposed algorithm performs effectively than DDPG-based algorithm. Our future work will focus on joint optimization of beamforming and VUEs trajectory in multiple RIS assisted VEC network.

References

[1] Q. Yuan, B. Chen, G. Luo, J. Li and F. Yang, "Integrated route planning and resource allocation for connected vehicles," in *China Commun.*, vol. 18, no. 3, pp. 226-239, March 2021.

[2] Z. Liang, Y. Liu, T. -M. Lok and K. Huang, "Multi-cell mobile edge computing: joint service migration and resource allocation," in *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 5898-5912, Sept. 2021.

[3] Q. Yuan, J. Li, H. Zhou, T. Lin, G. Luo and X. Shen, "A joint service migration and mobility optimization approach for vehicular edge computing," in *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9041-9052, Aug. 2020.

[4] S. Huang, S. Wang, R. Wang, et al, "Reconfigurable intelligent surface assisted mobile edge computing with heterogeneous learning tasks," in *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 2, pp. 369-382, June 2021.

[5] M. Mukherjee, V. Kumar, S. Kumar, et al, "Reconfigurable intelligent surface-assisted edge computing to minimize delay in task offloading," in *2021 IEEE Global Communications Conference (GLOBECOM)*, 2021,pp. 01-06.

[6] J. Wang, J. Hu, G. Min, Q. Ni and T. El-Ghazawi, "Online service migration in mobile edge with incomplete system information: a deep recurrent actor-critic learning approach," in *IEEE Trans. Mobile Comput.*, Nov. 2023.

[7] S. Huang, S. Wang, R. Wang, M. Wen and K. Huang, "Reconfigurable intelligent surface assisted edge machine learning," in *ICC 2021-IEEE International Conference on Communications*, 2021, pp. 1-6.

[8] D. Bruneo, "A stochastic model to investigate data center performance and QoS in IaaS cloud computing systems," in *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 3, pp. 560–569, Mar. 2014.

[9] Xiong J, Wang Q, Yang Z, et al. "Parametrized deep Q-networks learning: reinforcement learning with discrete-continuous hybrid action space," 2018. [Online]. Available: https://arxiv.org/abs/1810.06394.

[10] T. P . Lillicrap et al, "Continuous control with deep reinforcement learning," in *Proc. 4th Int. Conf. Learn. Representations (ICLR)*, San Juan, Puerto Rico, 2019.

[11] D. Silver et al., "Deterministic policy gradient algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 387–395.