Deep Reinforcement Learning-Based Joint Task Offloading and Radio Resource Allocation for Platoon-Assisted Vehicular Edge Computing

Yi Chen[1(✉)], Xinyu Hu[2], Haoye Chai[2], Ke Zhang[2], Fan Wu[2], and Lisha Gu[1]

[1] Research Institute of Highway Ministry of Transport, Mail Address No. 8 Xitucheng Road Haidian District, Beijing, China
{yi.chen,ls.gu}@rioh.cn

[2] School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China
huxinyu97@163.com, haoyechai@163.com, {zhangke,wufan}@uestc.edu.cn

Abstract. Platoons, formed by smart vehicles driving in the same patterns, bring potential benefits to road traffic efficiency while providing a promising paradigm to execute computation tasks with onboard computing resources. However, constrained resources of individual vehicles (IV), limited wireless coverage of vehicular communication nodes as well as high mobility of running platoons pose critical challenges on task scheduling and resource management. To address these challenges, we propose a platoon-based vehicular edge computing mechanism, which exploits computation capabilities of both platoons and edge computing enabled Roadside Units (RSUs), and jointly optimizes task offloading target selection and resource allocation. Taking aim at minimize delay cost and energy consumption of the platoon-based task execution, we leverage deep deterministic policy gradient (DDPG) to design a learning algorithm, which efficiently determines target computation servers and obtains optimized resource scheduling strategies. Numerical results demonstrate that our algorithm significantly reduces delay and energy costs in comparing its performance to that of benchmark schemes.

Keywords: Vehicular edge computing · Task offloading · Resource allocation · Platoon

1 Introduction

The Internet of vehicles (IoV) is a vital application of the Internet of things (IoT) in the automotive industry and is regarded as the information foundation for the next generation of intelligent transportation system with great potential [1]. To

Supported by organization x.

X. Wang et al. (Eds.): 6GN 2020, LNICST 337, pp. 69–85, 2020.
https://doi.org/10.1007/978-3-030-63941-9_6

reduce the traffic accident rate, as well as improve traffic efficiency and traveling convenience, the amount of intelligent transportation applications such as automatic driving, intelligent auxiliary driving for vehicles have been increasing consistently. However, these emerging applications not only require high computational complexity but also have strict delay sensitivity [2]. The current limited computing capability and storage capacity of the on-board equipment may not fully meet the requirements. On the other hand, the vehicular edge computing (VEC) [3] which combines edge computation and vehicle networks is widely considered as a promising approach to handle the computation-intensive tasks. In this case, the computation-intensive tasks of the vehicles can be offloaded to edge severs in proximity to them instead of the remote cloud servers, and thus the processing delay of tasks will be reduced significantly.

Compared with mobile edge computing (MEC), VEC has more challenge due to the high mobility and distributed nature of vehicles [4]. The author of [5] introduced a software-defined vehicular edge computing (SD-VEC) architecture where a controller guides both the vehicles task offloading strategy and the edge cloud resource allocation strategy. They devised a mobility-aware greedy algorithm (MGA) that determines the amount of edge cloud resources allocated to each vehicle. [6] studied the task offloading problem from a matching perspective based on three vehicular mobility models to simulate the movement of vehicles. To minimize the network delay, they proposed a pricing-based one-to-one matching algorithm and pricing-based one-to-many matching algorithms for the task offloading. However, all the mentioned works ignore the fact that compared with the smart mobile phones or tablets, vehicles have much powerful computing capacity, and the aggregate computing capacity will grows with the number of vehicles. Therefore, if the vehicles can be utilized to provide task offloading services, the computing performance of the vehicular networks will be subsequently improve.

Considering how to utilize the computing ability of vehicles, [7] introduced the federated offloading of vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communication in MEC-enabled vehicular networks. They aimed at minimizing the total latency for the moving vehicles. In [8], a computation offloading method named V2X-COM is proposed, which employs vehicle-to-everything (V2X) technology for data transmission in edge computing. Non-dominated sorting genetic algorithm III (NSGA-III) is adopted to generate balanced offloading strategies. The research above address to utilize the excess computing capacity of a single vehicle. As a matter of fact, facing the huge computing requirements for IoV, the computing capacity of a single vehicle is very limited. We have to develop an approach to aggregate the computing capacity of multiple vehicles to fulfill task offloading requirements. So how to centralize the use of multiple vehicles is still a subject to be studied.

With the rapid development of artificial intelligence, more and more researches apply reinforcement learning to solve the problem of offloading. [9] provided two novel approaches termed as distributed deep deterministic policy gradient (DDDPG) and sharing deep deterministic policy gradient (SDDPG)

based on DDPG algorithm, solving the multi-agent learning and non-cooperative power allocation problem in D2D-based V2V communications. The author of [10] formulated the offloading decision as a resource scheduling problem with single or multiple objective function and constraints, proposing a knowledge driven (KD) service offloading framework by exploring the deep reinforcement learning (DRL) model to find the long-term optimal service offloading policy. However, to our best knowledge, there is few research to discuss how to apply the reinforcement learning algorithms to the computing resource management for platoon.

Inspired by the above observations, we exploit a platoon-based VEC to offload part of computing load to platoons. In this case, an IV with requirements can select a proper offloading server, i.e., a RSU with MEC server or a platoon, according to its computation capability and its continuous wireless connection duration with the IV. In addition, in order to develop the computation capability of a platoon, the leader of the platoon is responsible for unified management of the radio resource owned by this platoon and the computing resource distributed across each vehicle within the platoon. To this end, the tasks offloaded to the platoon will be further divided into series of subtasks with different computational complexity and assigned to the platoon members associated with different proportions of the radio resource by the leader of the platoon.

In this paper, we jointly optimize the task offloading selection, inner-platoon radio resource and computation resource allocation to minimize latency and energy consumption in the platoon-based VEC. In summary, the main contributions of this paper include:

1) We devise a platoon-based VEC system model for the tasks offloaded from multiple IVs to reduce system task-offloading delay and the energy consumption. Many practical conditions and constraints for the vehicular networks are considered in our proposed model, especially including the duration of communication connection between the vehicle and a platoon or an RSU, the radio resource limitation of each platoon, and the difference of computing capability of each platoon member.
2) We formulate an optimization problem for joint task offloading target selection, inner-platoon radio resource and computing resource allocation with the objective of minimizing the overall system average offloading cost, which is defined as the weighted sum of average task-offloading delay and energy consumption.
3) Since the formulated problem is a well-known NP-hard problem (i.e., maximum cardinality bin packing problem [11]), and we have no priori-knowledge of arrival distribution of computation tasks, channel state and vehicles locations in practical. We propose a DDPG-based joint task offloading decision algorithm by integrating deep neural networks and reinforcement learning approaches.

The remainder of this paper is organized as follows. Section 2 gives the system model. In Sect. 3, we introduce a DDPG-based approach for task offloading and resource allocation. Simulation results and discussions are given in Sect. 4. Finally, we conclude this paper and propose some future works in Sect. 5.

2 System Model

In this section, we first describe the multi-lane scenario and present the system model for platoon-based VEC, including network model, communication model and computation model. Then the problem of jointly optimizing the offloading and resource allocation decision is formulated in details.

2.1 Network Model

As illustrated in Fig. 1, we consider a vehicular network consisting of m RSUs and n platoons and v IVs that do not belong to any platoon. A platoon consists of a group of vehicles travelling in the same lane and maintaining constant relative velocity. Vehicles belonging to the same platoon referred to as platoon members and each platoon has a platoon leader, which is responsible for the resource allocation within the platoon. The RSU provides computation resources (e.g. CPU cycles per second) for vehicles within its radio coverage. Let $\mathcal{S}erver = \left\{ \underbrace{S_1, S_2, ..., S_m}_{RSU}, \underbrace{S_{m+1}, S_{m+2}, ..., S_{m+n}}_{Platoon} \right\}$ denote the set of servers. The coordinates of of server i at time slot t is (a_i^t, b_i^t), a_i^t is the horizontal coordinates while b_i^t is the vertical coordinate. As for platoon, (a_i^t, b_i^t) is the coordinates of leader. The computational capability of each server is f_i^t. The platoon size and platoon length for platoon i is N_i and L_i, respectively. The j-th vehicle in platoon i is denoted by V_i^j and the leader is V_i^0. The computational capability of V_i^j at time slot t is $f_{i,j}^t$.

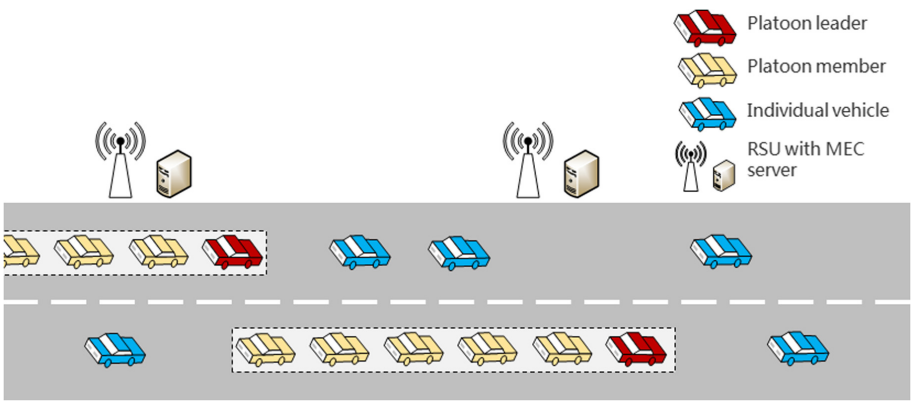

Fig. 1. The multi-lane scenario.

We consider the task offloading procedure for IVs. We assume that the vehicles running on the road follows a Poisson process with rate λ per unit time

interval. At each time slot there are $U(t)$ vehicles which have to offload, defined as $\mathcal{U}^t = \left\{V_1^t, V_2^t, ..., V_{U(t)}^t\right\}, U(t) < v$. The coordinate of the vehicle u at the time slot t is (a_u^t, b_u^t). Suppose the vehicle's velocity in each time slot remains unchanged, we use v_u^t to represent the velocity of V_u^t and use $v_i^{t,P}$ to represent the velocity of platoon i. The velocity is assumed to follow constrained Gaussian distribution independently and identically with probability distribution function (PDF)[12]

$$\hat{f}(v) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(v-\mu)^2}{2\sigma^2}}, v \in [v_{\min}, v_{\max}], \tag{1}$$

where $v_{\min}$ and $v_{\max}$ are lower and upper bound of velocity, respectively; μ and σ respectively represent mean and standard deviation of velocity.

Let r_i^L and r_i^{Mem} denote the transmission ranges of the platoon leaders and members, respectively. Assume that in each platoon the member's transmission range is the same, and $r_i^{Mem} < r_i^L$. Platoon leaders are trucks with higher-placed antennas in order to cover all the members. Since the transmitting power of each RSU is different, the wireless coverage range of each RSU is different. Let r_i denote the coverage range of RSU i.

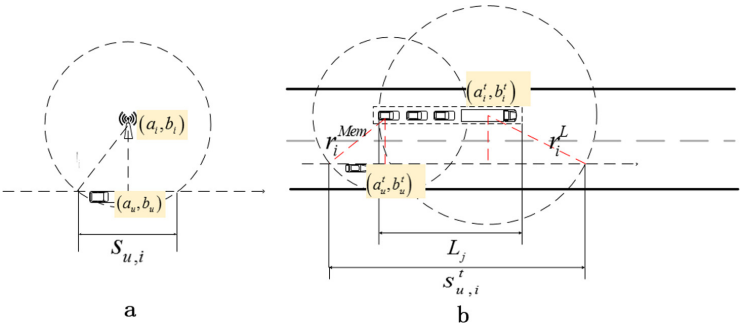

Fig. 2. A topology of the vehicle within the range of the RSU and the platoon.

As shown in Fig. 2a, at time slot t, the distance traveled by a vehicle within the coverage of RSU i is $s_{u,i}^t = 2\sqrt{r_i^2 - (b_u^t - b_i^t)^2}$. The duration of IV staying within the coverage is

$$t_{u,i,t}^{stay} = \begin{cases} \frac{s_{u,i}^t}{2} + \left(a_i^t - a_u^t\right) \\ \hline v_u^t \\ 0 \end{cases} \begin{array}{l} , \left|a_i^t - a_u^t\right| \leq \frac{s_{u,i}^t}{2} \\ , \left|a_i^t - a_u^t\right| > \frac{s_{u,i}^t}{2} \end{array}. \tag{2}$$

As shown in Fig. 2b, at time slot t, the distance traveled by a vehicle within the coverage of platoon i is $s_{u,i}^t = L_i + 2\left(\sqrt{\left(r_i^L\right)^2 - (b_u^t - b_i^t)^2} + \sqrt{\left(r_i^{Mem}\right)^2 - (b_u^t - b_i^t)^2}\right)$, where $v_{rel} = \left|v_i^{t,P} - v_u^t\right|$ is the relative velocity. The duration of IV staying within the coverage is

$$t_{x,i,t}^{stay} = \begin{cases} 0, & |a_i^t - a_u^t| > \sqrt{\left(r_i^L\right)^2 - \left(b_u^t - b_i^t\right)^2} \,\&\, |a_i - a_x| < L + \sqrt{\left(r_i^{Mem}\right)^2 - \left(b_u^t - b_i^t\right)^2} \\ \frac{s_{u,i} - \left(a_i^t - a_u^t + \sqrt{\left(r_i^L\right)^2 - \left(b_u^t - b_i^t\right)^2}\right)}{v_{rel}}, & v_i^{t,P} \geq v_u^t \\ \frac{a_i^t - a_u^t + \sqrt{\left(r_i^L\right)^2 - \left(b_u^t - b_i^t\right)^2}}{v_{rel}}, & v_i^{t,P} < v_u^t \end{cases}. \tag{3}$$

We assume that when a vehicle enters the communication range of the platoon, it is deemed that it can establish a connection with the leader.

2.2 Computation Model

We focus on the widely used task model $Q_u^t = \{I_u^t, C_u^t\}$, where I_u^t and C_u^t stand for the size of computation input data and the total number of CPU cycles needed to accomplish the task, respectively. In this paper, we consider these two parts can be split proportionally.

In general, a computation task can be executed at an RSU or a platoon. Let $\rho_{u,i}^t \in (0,1)$ denote the offloading decision, $\rho_{u,i}^t = \begin{cases} 1, & \text{task } u \text{ offloads to server } i \\ 0, & \text{others} \end{cases}$, we have

$$\sum_{i=1}^{m+n} \rho_{u,i}^t = 1, \forall u \in \mathcal{U}^t, t \in \mathcal{T}, \tag{4}$$

which means one task can only offload to one server. Moreover, since the computation capacity of each server is limited, the following constraint must be held,

$$\sum_{u=1}^{u(t)} \rho_{u,i}^t C_u^t \leq f_i^t, \forall u \in \mathcal{U}^t,\ i \in \mathcal{S}erver,\ t \in \mathcal{T}. \tag{5}$$

Let $\theta_{u,i,j}^t \in (0,1)$ denote the task allocation proportion for the j-th vehicle in platoon i when offloading task u, and the corresponding offloading decision of task u is denoted by $\bar{\theta}_{u,i}^t = \left\{\theta_{u,i,1}^t, \theta_{u,i,2}^t, ..., \theta_{u,i,N_i}^t\right\}$,

$$\sum_{j=1}^{N_i} \theta_{u,i,j}^t = 1, \forall u \in \mathcal{U}^t, t \in \mathcal{T}, i \in [m+1, m+n] \tag{6}$$

Also, the computation capacity of each vehicle is limited,

$$\sum_{u=1}^{u(t)} \rho_{u,i}^t \theta_{u,i,j}^t C_u^t \leq f_{i,j}^t, \forall u \in \mathcal{U}^t, i \in [m+1, m+n], j \in \{1, 2, ..., N_i\}, t \in \mathcal{T}. \tag{7}$$

After time slot t, the computation capacity of each vehicle and each RSU becomes

$$\begin{cases} f_{i,j}^{t} = f_{i,j}^{t-1} - \sum_{u=1}^{u_{t-1}} \sum_{j=1}^{N_i} \varphi_{u,i}^{t} \rho_{u,i}^{t} \theta_{u,i,j}^{t} C_u^t \\ f_{i}^{t} = f_{i}^{t-1} - \sum_{u=1}^{u_{t-1}} \sum_{j=1}^{N_i} \varphi_{u,i}^{t} \rho_{u,i}^{t} C_u^t \end{cases}, \tag{8}$$

where $\varphi_{u,i}^{t} = 1$ indicates that task u offloaded to server i can not be completed in this time slot.

2.3 Communication Model

We consider that one IV only accesses to one server in a time slot for the data transmission. Let $\left|h_{u,i}^{t}\right|^2$ and $d_{u,i}^{t}$ denote the coefficient of the effective channel power gain and the distance from vehicle u to server i, respectively. In this paper, we consider the scenario that the users move very slowly during the data offloading, so $\left|h_{u,i}^{t}\right|^2$ can be seen as a constant in a time slot and can change over different time slots (i.e., block fading channel). Hence, the corresponding channel power gain can be given as $\left|h_{u,i}^{t}\right|^2 = G \cdot \left(d_{u,i}^{t}\right)^{-\alpha} \cdot |h_0|^2$, where G is the power gain constant introduced for the amplifier and antenna, and $h_0 \sim CN(0,1)$ represents the complex Gaussian variable that represents Rayleigh fading [13].

Without loss of generality, $d_{u,i}^{t}$ denotes the distance between the server (the RSU center or the vehicle leader) and vehicle at the beginning of each time slot and $d_{u,i}^{t}$ is $d_{u,i}^{t}=\sqrt{\left(a_i^t - a_u^t\right)^2 + \left(b_u^t - b_i^t\right)^2}$. After a time slot the distance changes to

$$d_{u,i}^{t+1} = \begin{cases} \sqrt{\left(a_i^t - a_u^t - v_u^t \Delta t\right)^2 + \left(b_u^t - b_i^t\right)^2}, & i \in (1,m) \\ \sqrt{\left(a_i^t - a_u^t - v_{rel} \Delta t\right)^2 + \left(b_u^t - b_i^t\right)^2}, & i \in (m+1, m+n) \end{cases}. \tag{9}$$

The signal noise ratio (SNR) of task u transmission from V_u^t to server i at time slot t is expressed as

$$\gamma_{u,i}^{t} = \frac{P_u \left|h_{u,i}^{t}\right|^2}{\sigma^2}. \tag{10}$$

where P_u, σ^2 are the uplink transmit power of vehicle u, the noise power, respectively. The uplink data rate of a vehicle that chooses to offload its task to the server via a wireless link can be expressed as

$$R_{u,i}^{t}=B\log_2\left(1+\gamma_{u,i}^{t}\right), \tag{11}$$

where B is the available spectrum bandwidth.

On the occasion of a vehicle decide to offload the task to a platoon, the task will first be offloaded to the leader, then the leader completes the secondary offloading of the task according to the characteristics of the platoon and the allocation of computing and communication resources in the platoon.

Let $\omega_{u,i,j}^t \in (0,1)$ denote the spectrum allocation proportion for the j-th vehicle in platoon i when transferring task u, and the corresponding offloading decision of task u is denoted by $\bar{\omega}_{u,i}^t = \left\{\omega_{u,i,1}^t, \omega_{u,i,2}^t, ..., \omega_{u,i,N_i}^t\right\}$,

$$\sum_{j=1}^{N_i} \omega_{u,i,j}^t = 1, \forall u \in \mathcal{U}^t, i \in \mathcal{S}erver, t \in \mathcal{T}. \tag{12}$$

According to the above, when the leader sends task u to v_i^j, the SNR can be expressed as

$$\gamma_{u,i,j}^t = \frac{P_i^L \left|h_{u,i,j}^t\right|^2}{\sigma^2}. \tag{13}$$

The data rate can be expressed as

$$R_{u,i,j}^t = \omega_{i,j}^t B \log_2\left(1 + \gamma_{u,i,j}^t\right), \tag{14}$$

where P_i^L, $\left|h_{u,i,j}^t\right|^2$ and B are the transmit power of the leader of platoon i, the effective channel power gain from leader to V_i^j and the available spectrum bandwidth, respectively.

2.4 Problem Formulation

In this subsection, the optimal problem formulation will be described in detail. The server needs to make a joint optimization of the offloading proportion, communication resource and computation resource allocation. For each task generated by a vehicle, it can either be computed by a platoon or an RSU. According to different computation and resource allocation strategies, the latency and energy consumption of the task may be different.

For in-vehicle applications, latency must be taken into consideration seriously. On the other hand, due to the limited battery of the vehicles energy consumption should also be taken into account. However, minimize both energy consumption and the latency are conflicting. For example, the vehicle can save the energy by setting the lowest frequency all the time, but this will certainly increase the computing time. Therefore, we consider a trade-off analysis between the energy consumption and the execution delay for the offloading decision. Next we will introduce the formulas for calculating energy consumption and time delay.

When the task is offloaded to RSU, the communication time between the vehicle task generator and the RSU can be expressed by

$$T_{u,i}^{t,R,com} = \frac{I_u^t}{B \log_2\left(1 + \gamma_{u,i}^{t,R}\right)}. \tag{15}$$

The computation execution time of task u completed by RSU can be expressed as

$$T_{u,i}^{t,R,cmp} = \frac{C_u^t}{f_i^t}. \tag{16}$$

Then, the communication and computation energy consumption of task u by accomplishing at RSU can be calculated as

$$E_{u,i}^{t,R,com}=P_u T_{u,i}^{t,R,com}. \tag{17}$$

$$E_{u,i}^{t,R,cmp} = P_i^{t,R} T_{u,i}^{t,R,cmp}, \tag{18}$$

respectively, where $P_i^{t,R}$ indicate the computation power of RSU i.

When the task choose to offload at platoon, the communication time between the vehicle which has task and the leader is defined as follows,

$$T_{u,i}^{t,P,com1}(\omega) = \frac{I_u^t}{B\log_2\left(1+\gamma_{u,i}^{t,P}\right)}. \tag{19}$$

And the communication time within the platoon, i.e. the transmission between the leader and the member, can be expressed as

$$T_{u,i,j}^{t,P,com2}(\theta,\omega) = \frac{\theta_{u,i,j}^t I_u^t}{\omega_{u,i,j}^t B\log_2\left(1+\gamma_{u,i,j}^t\right)}. \tag{20}$$

The computation execution time of task u completed by each vehicle assigned to the task is

$$T_{u,i,j}^{t,P,cmp}(\theta) = \frac{\theta_{u,i,j}^t C_u^t}{f_{i,j}^t}. \tag{21}$$

Then, the communication energy consumption of task offloaded to the leader can be calculated as

$$E_{u,i}^{t,R,com1}(\omega) = P_u T_{u,i}^{t,P,com1}(\omega), \tag{22}$$

and the intra-platoon communication and computation energy consumption are

$$E_{u,i,j}^{t,R,com2}(\theta,\omega) = P_i^L T_{u,i,j}^{t,P,com2}(\theta,\omega), \tag{23}$$

$$E_{u,i,j}^{t,P,cmp}(\theta,\omega) = P_{i,j}^t T_{u,i,j}^{t,P,cmp}(\theta) \tag{24}$$

respectively, where $P_{i,j}^t$ indicates the computation power of member j in platoon i.

Generally, the amount of data returned are very small, so the time delay of the results back to the vehicle can be ignored.

The total energy consumption and time delay are shown to be

$$E_{u,i}^t = \begin{cases} E_{u,i}^{t,R,com} + E_{u,i}^{t,R,cmp}, & i \in (1,m) \\ E_{u,i}^{t,R,com1} + \sum\limits_{j=1}^{N_i}\left[E_{u,i,j}^{t,R,com2} + E_{u,i,j}^{t,P,cmp}\right], & i \in (m+1,m+n) \end{cases}, \tag{25}$$

$$D_{u,i}^{t}=\begin{cases} T_{u,i}^{t,R,com}+T_{u,i}^{t,R,cmp}+\varphi_{u,i}^{t-1}T_{wait}, & i\in(1,m) \\ T_{u,i}^{t,P,com1}+\max\limits_{j=1}^{N_i}\left[T_{u,i,j}^{t,P,com2}+T_{u,i,j}^{t,P,cmp}\right]+\varphi_{u,i,j}^{t-1}T_{wait}, & i\in(m+1,m+n) \end{cases}, \tag{26}$$

If the task can't be completed in this time slot, as for RSU, $\varphi_{u,i}^{t}=1$ means $T_{u,i}^{t,R,cmp}>\Delta t-T_{u,i}^{t,R,com}$. As for platoon, $\varphi_{u,i,j}^{t}=1$ when $T_{u,i}^{t,P,cmp}>\Delta t-T_{u,i}^{t,P,com1}-T_{u,i}^{t,P,com2}$. So the time delay to complete the remaining tasks in the previous slot is

$$T_{wait}=\left(C_{u}^{t-1}-f_{u,i}^{t-1}T_{u,i}^{t-1,R,cmp}\right)/f_{u,i}^{t}, \tag{27}$$

$$T_{wait}=\left(\theta_{u,i,j}^{t-1}C_{u}^{t-1}-f_{u,i}^{t-1}T_{u,i}^{t-1,P,cmp}\right)/f_{u,i}^{t}, \tag{28}$$

on the occasion of offloading to RSU and platoon respectively.

Then we consider an optimization problem about the offloading decision, communication resource and computation resource allocation. The aim is to provide optimal computation offloading decision $\rho_{u,i}^{t}$, task allocation proportion $\bar{\theta}_{u,i}^{t}$ and spectrum allocation proportion $\bar{\omega}_{u,i}^{t}$ for all vehicles such that the energy consumption and the maximal task completion time is minimized. The corresponding optimization problem can be formulated as follows,

$$\min_{\rho_{u,i}^{t},\theta_{u,i,j}^{t},\omega_{u,i,j}^{t}}\frac{1}{T}\sum_{t=1}^{T}\sum_{u=1}^{u(t)}\sum_{i=1}^{m+n}\left\{\rho_{x,i}^{t}\left[\alpha E_{u,i}^{t}\left(\theta,\omega\right)+\beta\min\left[D_{u,i}^{t}\left(\theta,\omega\right),\Delta t\right]\right]\right\} \tag{29}$$

$$\begin{aligned} s.t.\ & C1:(4) \\ & C2:(6) \\ & C3:(5),(7) \\ & C4:(12) \\ & C5:\left\lceil\theta_{u,i,j}^{t}\right\rceil\odot\left\lceil\omega_{u,i,j}^{t}\right\rceil=1,\forall u\in\mathcal{U}^{t},\ i\in\mathcal{S},\ j\in\{1,2,...,N_i\},t\in\mathcal{T} \\ & C6:P_u\geq P_{th},\rho_{u,i}^{t}P_{i}^{L}\geq P_{th}\ ,\forall u\in\mathcal{U}^{t},i\in\mathcal{S},t\in\mathcal{T} \\ & C7:\gamma_{u,i}^{t}\geq\gamma_{th},\rho_{u,i}^{t}\theta_{u,i,j}^{t}\gamma_{u,i,j}^{t}\geq\gamma_{th},\forall u\in\mathcal{U}^{t},i\in\mathcal{S},j\in\{1,2,...,N_i\},t\in\mathcal{T}. \end{aligned}$$

where α, β are weighting factor.

Constraint C5 means that only platoon members with tasks will have the allocated spectrum. Constraint C6 ensures that the transmission power of each vehicle is lower than the threshold P_{th}. C7 is the SNR requirement for successful transmission.

3 Deep Reinforcement Learning for Task Offloading and Resource Allocation

Deep reinforcement learning combines the perception ability of deep learning with the decision-making ability of reinforcement learning, which can be controlled directly according to the input information. In our study, we used DDPG to learn optimal task offloading and resource allocation strategies.

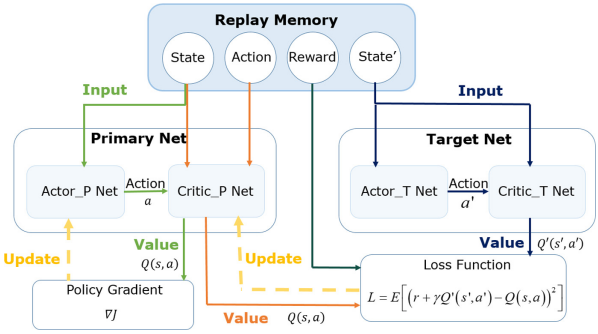

Fig. 3. The framework of deep deterministic policy gradient learning algorithm.

3.1 DDPG-Based Learning Model

The DDPG algorithm is a combination of Deterministic Policy-Gradient [14] Algorithms, the Actor-Critic [15] Methods, and the Deep Q-Network (DQN). It follows the target network and experience replay technology in the DQN algorithm, using two deep-Q Networks in the algorithm: one is Actor network used to approximate the policy function and the other is Critic network used to approximate the value function. The Actor realizes the output of continuous action values, and the Critic evaluates the execution effect of the action. The framework of DDPG algorithm is demonstrated in Fig. 3.

In reinforcement learning, agents find optimal strategies through interaction with the environment and trial-and-error learning. States, actions and rewards are three key factors of reinforcement learning and we formulate the task as a Markov decision process (MDP). We define a 4-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P} \rangle$, where they represent the agent state space, action space, reward function $r = \mathcal{R}(s, a, s')$ and the transition function $\mathcal{P}^a_{ss'}$, respectively. And $s, s' \in \mathcal{S}$, $a \in \mathcal{A}$, $\mathcal{P}^a_{ss'}$ is the probability of a transition from state s to state s' when taking action a.

Apply it to the task offloading and resource allocation problem, we define that each IV and each platoon is an agent, but the decisions are made from a central server. Generally, an agent observes a state $s^t \in \mathcal{S}$ at each time t, then accordingly takes action $a^t \in \mathcal{A}$ based on the policy π. By taking the action a^t, the agent receives a reward r^t and the environment transits to the next state s^{t+1}. Then we define the state spaces, action spaces and reward function of the joint optimization of task offloading and resource allocation as follows:

State Spaces: At the beginning of each time period, the agent obtains environmental information and vehicular information. Specifically, the system state space contains:

- $Q_u^t = \{I_u^t, C_u^t\}$: Current information of task u.
- f_i^t, $f_{i,j}^t$: The maximum computation capability of the i-th server and the j-th vehicle in platoon i.
- v_u^t, $v_i^{t,P}$: The velocity of the u-th vehicle and the i-th platoon.

- (a_u^t, b_u^t), (a_i^t, b_i^t): The coordinates of the u-th vehicle and the i-th platoon.
- $R_{u,i}^t$, $R_{u,i,j}^t$: The transmission rate between the server and the user, the leader and the member.

Let $s^t \in \mathcal{S}$ denote the system state in our system, i.e.,

$$s^t = \left[Q_u^t, f^t, v^t, \left(a^t, b^t\right), R^t\right]. \tag{30}$$

Action Spaces: After receiving the environment and vehicular information, the central server decides which task should be offload to which server and how to allocate the resources. Let $a^t \in \mathcal{A}$ denote the action space in our system, i.e.,

$$a^t = \left[\rho_{u,i}^t, \bar{\theta}_{u,i}^t, \bar{\omega}_{u,i}^t\right] \tag{31}$$

Reward Function: Our objective is to minimize the total delay and energy consumption of the network by interacting with the environments. Thus, we design a reward function $\mathcal{R}_t$ to get immediate return by executing action a^t as

$$\mathcal{R}_t = \sum_{u=1}^{u(t)} \sum_{i=1}^{m+n} \left\{\rho_{x,i}^t \left[\alpha E_{u,i}^t\left(\theta, \omega\right) + \beta \min\left[D_{u,i}^t\left(\theta, \omega\right), \Delta t\right]\right]\right\} \tag{32}$$

The agent can achieve the optimal results by adjusting Q value according to the updated rule

$$Q\left(s, a\right) \leftarrow Q\left(s, a\right) + \alpha\left[r + \gamma \max_{a' \in \mathcal{A}} Q\left(s', a'\right) - Q\left(s, a\right)\right] \tag{33}$$

where α and γ are the learning rate and the discount factor, respectively, $\alpha, \gamma \in [0, 1]$.

The policy gradient can be computed as (32) to update the actor's primary network,

$$\begin{aligned} \nabla_{\theta^\mu} J &\approx \tfrac{1}{|\mathbb{D}|} \sum_t \left[\nabla_{\theta^\mu} Q\left(s, a|\theta^Q\right)|_{s, a=\mu(s|\theta_\mu)}\right] \\ &= \tfrac{1}{|\mathbb{D}|} \sum_t \left[\nabla_a Q\left(s, a|\theta^Q\right)|_{s, a=\mu(s)} \nabla_{\theta^\mu} \mu\left(s|\theta_\mu\right)|_s\right]. \end{aligned} \tag{34}$$

The value function is donated as $V^\pi(s)$. There is an optimal value $V^*(s)$ corresponding to an optimal policy $\pi*$ of all the possibility of the value function $V^\pi(s)$. By choosing the action to maximize the reward, the optimal policy $\pi*$ can be retrieved from optimal value function $V^*(s)$. So we can define $V^*(s)$ and the optimal policy as

$$V^*(s) = V^{\pi*}(s) = \max_{a \in \mathcal{A}} \left\{r\left(s, a\right) + \gamma \sum_{s' \in \mathcal{S}} p\left(s'|s, a\right) V^*\left(s'\right)\right\} \tag{35}$$

$$\pi*(s) = \arg\max_{a \in \mathcal{A}} \left\{r\left(s, a\right) + \gamma \sum_{s' \in \mathcal{S}} p\left(s'|s, a\right) V^*\left(s'\right)\right\} \tag{36}$$

We use the primary deep neural network with the parameters θ^{μ} and θ^{Q} Besides, the parameters of the target policy network and the target value network are respectively represented as $\theta^{\mu'}$ and $\theta^{Q'}$ which are approached to the primary network parameters with a small amount periodically by (36),

$$\begin{cases} \theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\,\theta^{\mu} \\ \theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\,\theta^{Q} \end{cases}, \tag{37}$$

where τ is the update coefficient, $\tau \in [0, 1]$.

3.2 DDPG-Based Joint Task Offloading Decision Algorithm (JTO)

The process of DDPG-based JTO Algorithm is summarized in Algorithm 1. We construct a replay memory to store a series of historical experiences. By randomly choosing a mini-batch sample from replay memory, the network parameters will be updated.

The algorithm parameters include the replay memory $\mathbb{D}$, the total time slot T, the number of servers and vehicles which have task to offload, discount factor and learning rates. After initializing the parameters, the decision of task offloading and resource allocation is executed at the beginning of each episode t. The system will choose an action with random noise N_t and then receive the reward and turn to the next state. Furthermore, the parameters of the network will be updated.

Algorithm 1: DDPG-based JTO Algorithm

Input: $\mathbb{D}$, K, S, T, V, γ, σ, s_0

initialize replay memory $\mathbb{D}$ to capacity D^*, $\mathbb{D}(K) = \emptyset$;

initialize networks θ_μ and θ_Q with random weight, target networks $\theta_{\mu'} = \theta_\mu$ and $\theta_{Q'} = \theta_Q$;

for *episode* $= 0 \rightarrow T - 1$ **do**

  initialize multi-lane scenario,task assignment and resource allocation process as the state $s_0 = \{Q, F, v, (a, b), R\}$;

  for $t = 0 \rightarrow l - 1$ **do**

    perform action $a = \theta_\mu + N_t$;

    observe reward r and s', **execute** action a, based on (32), (33) and (36);

    Store transition (s, a, r, s') in $\mathbb{D}$;

    if *replay memory* $\mathbb{D}$ *is full* (D^*) **then**

      sample a random batch of K transitions (s, a, r, s') from $\mathbb{D}$;

      Set $\triangle_t = r + \upsilon Q(s', a'|\theta_Q)$;

      Update the parameter of critic network θ_Q by minimizing the loss: $L = E^2[\triangle_t - Q(s, a|\theta_{Q'})]$;

      Update the parameter of actor network θ_μ by using the sampled policy gradient in (34) ;

      Update target network $\theta_{\mu'}$ and $\theta_{Q'}$ using (37);

return θ_μ and θ_Q;

4 Performance Evaluation

In this section, we use Python to build a simulation environment for the multi-lane platoon system. Furthermore, we use Tensorflow platform to implement the DDPG-based JTO scheme. Our implementation is based on the open-source package DDPG [16].

For performance comparison, we present two benchmark schemes: RSU task offloading scheme (RTO) and other vehicles task offloading scheme (VTO),

1) *RTO:* Each IV only chooses to offload the task to RSU by V2I.
2) *VTO:* Each IV only chooses to offload the task to other vehicles by V2V.

The simulation parameters are given in Table 1.

Table 1. Simulation parameters

System parameter	Value/description
Number of IVs (tasks), u	10
Number of RSUs, m	4
Number of platoons, n	5
Number of vehicles in a platoon, N_i	4
Bandwith, B	20 MHz
Computation power P_i^t	1 W
Transmission power P_u	125 mW
Time slot Δt	90 s
Size of computation input data I_u^t	[250, 600] MB
Total number of CPU cycles C_u^t	[20, 30] cycle
Computation frequency f_i^t	[5, 20] cycles/s
Transmission rate R_i^t	[5, 20] Mbps
α, β	1

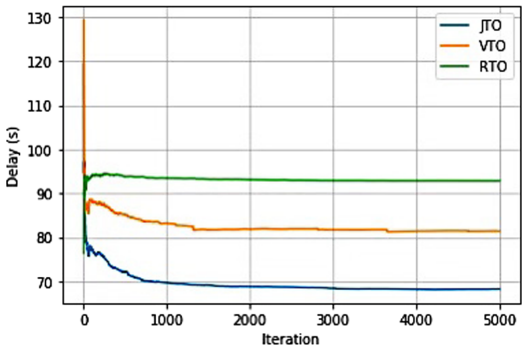

Fig. 4. The task execution delay achieved by different task offloading schemes.

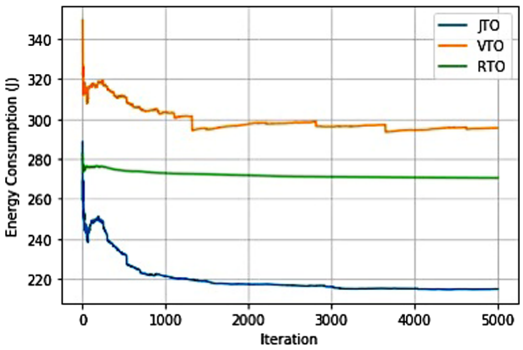

Fig. 5. The energy consumption achieved by different task offloading schemes.

In Fig. 4, we compare the performance of task execution delay of different task offloading schemes based on the DDPG learning algorithm. It can be easily seen that all the three task offloading schemes can approach their stable point as the number of episodes increases. We can draw the following observations from Fig. 5. Firstly, RTO has the highest delay because the RSU layout on the road may be sparse, and the distance is longer than that between vehicles. The channel quality is more easily affected too. Secondly, VTO can reduce latency since the possibility of connection interruption during transmission is less due to the relative speed between vehicles. Moreover, our proposed JTO schemes can yield the lowest delay as compared to the other benchmark schemes.

Figure 5 shows the comparison of the three schemes' energy consumption. Different to the task execution delay, the highest energy consumption is caused by VTO. The reason is that the computation capability of the vehicle is much smaller than that of RSU, so only using the V2V would require sending tasks to multiple vehicles, which would consume a lot of energy. However, the proposed JTO scheme combines the advantages of the other two schemes, so both the delay and the energy consumption are the lowest. As for the sudden rise for JTO when iteration is between 100 and 200, it is because the system tentatively assigning more tasks to the IVs, and then learning other schemes after increasing energy consumption in the learning process.

Figure 6 depicts the overall cost of those three proposed schemes, including the delay and the energy consumption with different amounts of tasks in the system. As the number of tasks increases, the energy consumption and delay of the whole system increase under these three scenarios. When there are fewer tasks, VTO must be lower than RTO because it uses vehicle collaboration. However, as the number of tasks increases, many tasks in V2V have to be offloaded to vehicles with poor communication quality, so its consumption increases rapidly. However, the algorithm proposed by us can reduce the consumption because it can choose RSUs and platoons adaptively.

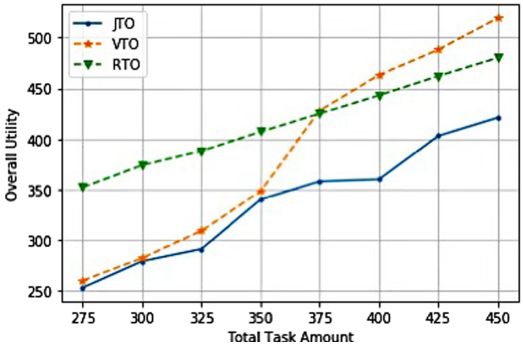

Fig. 6. The overall cost of different task offloading schemes.

5 Conclusion

In this paper, we have introduced a platoon-based VEC mechanism, which makes use of computing capabilities of both platoons and edge computing enabled RSU. Then we make a jointly optimization of task offloading target selection and inner-platoon resource allocation. We aims at minimizing the offloading cost, including task execution delay and energy consumption. In order to solve the problem, a DDPG-based algorithm has been proposed. Compared with other benchmark schemes, our proposed scheme significantly reduces task offloading latency and energy consumption, obtaining the optimized resource scheduling strategy, and especially in the case with a large number of tasks.

References

1. Li, W., Zhu, C., Leung, V.C.M., Yang, L.T., Ma, Y.: Performance comparison of cognitive radio sensor networks for industrial IoT with different deployment patterns. IEEE Syst. J. **11**(3), 1456–1466 (2017). https://doi.org/10.1109/JSYST.2015.2500518
2. Alam, K.M., Saini, M., Saddik, A.E.: Toward social internet of vehicles: concept, architecture, and applications. IEEE Access **3**, 343–357 (2015)
3. Huang, C., Chiang, M., Dao, D., Su, W., Xu, S., Zhou, H.: V2V data offloading for cellular network based on the software defined network (SDN) inside mobile edge computing (MEC) architecture. IEEE Access **6**, 17741–17755 (2018). https://doi.org/10.1109/ACCESS.2018.2820679
4. Zhou, H., et al.: TV white space enabled connected vehicle networks: challenges and solutions. IEEE Netw. **31**(3), 6–13 (2017). https://doi.org/10.1109/MNET.2017.1600049NM
5. Choo, S., Kim, J., Pack, S.: Optimal task offloading and resource allocation in software-defined vehicular edge computing. In: 2018 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, pp. 251–256 (2018). https://doi.org/10.1109/ICTC.2018.8539726

6. Liu, P., Li, J., Sun, Z.: Matching-based task offloading for vehicular edge computing. IEEE Access **7**, 27628–27640 (2019). https://doi.org/10.1109/ACCESS.2019.2896000
7. Wang, H., Li, X., Ji, H., Zhang, H.: Federated offloading scheme to minimize latency in MEC-enabled vehicular networks. In: 2018 IEEE Globecom Workshops (GC Wkshps), Abu Dhabi, United Arab Emirates, 2018, pp. 1–6 (2018). https://doi.org/10.1109/GLOCOMW.2018.8644315
8. Xu, X., Xue, Y., Li, X., Qi, L., Wan, S.: A computation offloading method for edge computing with vehicle-to-everything. IEEE Access **7**, 131068–131077 (2019). https://doi.org/10.1109/ACCESS.2019.2940295
9. Nguyen, K.K., Duong, T.Q., Vien, N.A., Le-Khac, N., Nguyen, L.D.: Distributed deep deterministic policy gradient for power allocation control in D2D-based V2V communications. IEEE Access **7**, 164533–164543 (2019). https://doi.org/10.1109/ACCESS.2019.2952411
10. Qi, Q., et al.: Knowledge-driven service offloading decision for vehicular edge computing: a deep reinforcement learning approach. IEEE Trans. Veh. Technol. **68**(5), 4192–4203 (2019). https://doi.org/10.1109/TVT.2019.2894437
11. Loh, K.H., et al.: Solving the maximum cardinality bin packing problem with a weight annealing-based algorithm. Oper. Res. Cyber Infrastruct. **47**, 147–164 (2009)
12. Roess, R.P., Prassas, E.S., McShane, W.R.: Traffic Engineering, 3rd edn. Pearson/Prentice Hall, New Jersey (2004)
13. Rappaport, T.S.: Wireless Communications: Principles and Practice, 2nd edn. Prentice Hall: Englewood Cliffs, New Jersey (1996)
14. Jan, P., Stefan, S.: Reinforcement learning of motor skills with policy gradients. Neural Netw. **21**(4), 682–697 (2008). https://doi.org/10.1016/j.neunet.2008.02.003
15. Konda, V.R., Tsitsiklis, J.N.: Actor-critic algorithm. In: NIPS, vol. 13, pp. 1008–1014 (1999)
16. Reimplementation of DDPG (Continuous Control with Deep Reinforcement Learning) based on OpenAI Gym + Tensorflow. https://github.com/songrotek/DDPG. Accessed Dec 2019