Computer Communications 151 (2020) 485–494

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Intelligent resource allocation management for vehicles network: An A3C learning approach

Miaojiang Chen [a], Tian Wang [b], Kaoru Ota [c], Mianxiong Dong [c], Ming Zhao [a], Anfeng Liu [a,*]

[a] School of Computer Science and Engineering, Central South University, ChangSha 410083, China
[b] College of Computer Science and Technology, Huaqiao University, Xiamen, 361021, China
[c] Department of Information and Electronic Engineering, Muroran Institute of Technology, Muroran 0508585, Japan

ARTICLE INFO

Keywords:
Deep reinforcement learning
Vehicular networks
Markov decision process
Quality of service
Artificial intelligence
Asynchronous advantage actor–critic (A3C)

ABSTRACT

With the increasing demand of users for high-speed, low-delay and high-reliability services in connected vehicles network, wireless networks with communication, caching and computing convergence become the trend of network development in the future. To improve the quality of services of vehicles network, we propose a virtualized framework for mobile vehicle services, which using a learning-based resource allocation scheme. The dynamic change processes are modeled as Markov chains without making assumptions about the optimization goal and reducing the complexity of resource allocation computing. A high performance asynchronous advantage actor–critic learning algorithm is proposed to solve the complex dynamic resource allocation problem. Base on software-defined networking and information-centric networking, the method can dynamic orchestration of computing and communication resources to enhance the performance of virtual wireless networks. Simulation results verify that the proposed scheme can converge at a fast speed and improve the network operator's total rewards.

1. Introduction

In recent years, a large number of researchers have promoted the development of wireless mobile networks, most of which focus on improving the ability of wireless network to provide communication services [1–4]. For early wireless mobile services, such as voice and short message, higher wireless link capacity or system throughput means better service quality and more satisfied users, thus bringing more revenue to mobile service operators. However, with the rapid development of wireless mobile networks, the types of services provided are also expanding rapidly. They have changed from traditional connection-centric services (such as telephone services) to content-centric services (such as live video broadcasting).

The researches of urban Traffic Flows [5], utility-aware data transmission [6], Energy Harvesting [7] in the Internet of things have widely attention. The Cisco Visual Network Index report predicts that video traffic will account for 72% of global network data traffic by 2019 [8]. At present, some popular new mobile applications, such as virtual reality (VR), augmented reality (AR), connected vehicle (CV), Internet of Things (IoT), cognitive Information and other wireless mobile applications [9–13], often require higher data transmission rate, lower transmission delay and stronger computing power. Due to the limitation of wireless spectrum resources, the growth rate of wireless network capacity always lags behind the growth rate of network services. However, congestion and inefficiency of large-scale content delivery in wireless networks can hardly be solved by simply improving the network's communication capability [14]. However, data security is very important in network service and trust computing [15], road networks with privacy [16] and privacy-preserving in vehicle network [17] have been widely studied. Furthermore, some technology [18–21] are proposed to solve above network problems.

Another promising technology, information-centric networking (ICN), which can efficiently reduce the duplicate content transmission in networks [22]. Recent advances in computing also have profound impacts on networks as well [23–25]. In order to cope with the change from wireless network to "content-centric" services demand, a novel design paradigm beyond the current "connection-centric" network architecture is needed, and mobile edge computing (MEC) [26] is used to allocate computing resources closer to terminal mobile devices. In the current mobile network, both at the base station and at the user mobile device, caching and computing capabilities have been widespread. Computing and caching resources are abundant, sustainable and cost-effective, unlike communication energy limited by wireless spectrum resources. These "green resources" have been following Moore's law for decades. Through information-centric network (ICN) technology

* Corresponding author.
E-mail addresses: miaojiangchen@csu.edu.cn (M. Chen), wangtian@hqu.edu.cn (T. Wang), ota@csse.muroran-it.ac.jp (K. Ota), mx.dong@csse.muroran-it.ac.jp (M. Dong), meanzhao@mail.csu.edu.cn (M. Zhao), afengliu@mail.csu.edu.cn (A. Liu).

https://doi.org/10.1016/j.comcom.2019.12.054
Received 14 October 2019; Received in revised form 30 November 2019; Accepted 29 December 2019
Available online 7 January 2020

and mobile edge computing (MEC) technology, making full use of the caching and computing capabilities of edge network nodes (e.g., base stations or user equipment), it is possible to fundamentally change the architecture of future wireless networks.

By integrating the cache into the edge nodes of the wireless network, the system can support flexible and efficient content retrieval, significantly reduce the transmission of duplicate content in the network, and reduce the transmission delay and the energy consumption of the content acquired by users. Meanwhile, the mobility, flexibility and security of green information-centric networking and edge networking in smart city can be improved [27,28]. In addition, convergent computing will enable wireless networks to have powerful data processing capabilities and to execute computationally intensive applications in the network. By migrating all or part of the computing tasks of mobile devices to edge nodes with computing power, compared with migrating to cloud servers with rich remote computing resources, it can significantly reduce the execution time of tasks, reduce the risk of network crash or service interruption, and improve the stability and robustness of the network.

Although the wireless network system which integrates communication, caching and computing has important application value and is the development trend of the future network, there are still many important challenges to be solved before the system is widely deployed and applied. These include how to meet the needs of time delay, bandwidth constraints, different resource interaction interfaces, mobile mobility management, and some non-technical issues, such as government regulations. Especially, in this fusion system, how to manage and allocate three different types of resources, namely, communication, caching and computing, in order to meet the needs of different users and different services efficiently is a key issue of the fusion system. There are mutual influences and constraints among communication, caching and computing. For instance, cache hits can save communication and computing resources. The quality of communication directly affects the benefits of caching and computing decisions. Computing can reduce the use of communication resources and save caching space. Therefore, considering these three kinds of resources and their corresponding functions, and optimizing the allocation of resources together, the total benefit of the end of the network can be optimized.

At present, many resource allocation problems in wireless networks are mainly solved by Jordan optimization, dynamic programming and game theory. Usually these methods need to make assumptions about the objective function or data distribution. As the wireless networks become more and more complex, these assumptions are often quite different from the actual situation, and their reference value to reality will become less and less. Meanwhile, with the large increase of the number of variables in the objective function, a large number of variables pose a serious challenges to the calculation based on mathematical methods and memory space. Deep Reinforcement Learning (DRL) [29–33] is an advanced machine learning algorithm. It combines the perception ability of in-depth learning and the decision-making ability of reinforcement learning. It can solve the decision-making problem in complex high-dimensional state spaces by using function approximation without making assumptions about the optimization goal or reducing the complexity of sub-optimization. Therefore, the use of deep reinforcement learning is expected to solve the problems of end-user management and control in complex, changeable and heterogeneous mobile environments.

Wang [34] et al. has studies the vehicular services in IoT. To expand their researches, we propose a scheme of virtual vehicular network with asynchronous advantage actor–critic algorithm. The main contributions of this scheme are as follows:

(1) Base on software-defined networking and information-centric networking, we introduce a method in which can dynamic orchestration of computing and communication resources to enhance the performance of virtual wireless networks.

(2) Resource allocation problem is essentially an optimization problem. In our model, we defined the resource allocation strategy as a Markov decision process and using function approximation without making assumptions about the optimization goal and reducing the complexity of resource allocation computing.

(3) Because of the complexity of the dynamic system, we use the high performance asynchronous advantage actor–critic learning algorithm to solve the resource allocation problem.

(4) According to experiment theoretical analysis, the experiment results show that our proposed scheme can achieve the better performance and the effectiveness compared with the scheme without learning-based algorithms.

The rest of this paper is organized as follows. In Section 2, the overview of the preliminaries and background is introduced. In Section 3, the system model is presented. In Section 4, the resource allocation problem of network is defined as a DRL Markov decision process. Experiment results are described and analyzed in Section 5. Finally, conclusions and future work are presented in Section 6.

Table 1
A list of variables.

Variables	Term
r	Reward
a	Action
s	State
$Q(s,a)$	Action value function
$V(s)$	State value function
π	Policy
θ	The weight of deep network
D	The experience pool
γ	Discount factor
α	Learning rate

2. Related work

2.1. Markov decision process

In the RL environment, agents make decisions based on signals from environments called environmental states. A reinforcement learning assignment that satisfies the Markov property is called a Markov decision process (MDP). In particular, if the action spaces and state spaces are finite, it is called finite Markov decision process. An MDP is denoted as a tuple $M = (S, A, p, R)$, where S represents a set of states, A is a set of actions, p is a transition probability from state s to s' when executed action a. To be more specific, when an agent executes an action $a \in A$ and receives a reward $r \in R$, the environment transitions from state $s \in S$ to $s' \in S$. R is the reward obtained after action a is executed. If the Markov property holds in the reinforcement learning environment, which means the transition to the next state s' is only determined by the current action a and state s. A list of variables used in the following is also shown in Table 1.

2.2. Reinforcement learning

There are two types of reinforcement learning: value-based reinforcement learning and strategy-based reinforcement learning. In addition, the actor–critic reinforcement learning method is a combination of value-based reinforcement learning method and strategy-based reinforcement learning method. Furthermore, according to whether the environmental elements (i.e. state transition probability and reward function) are known, reinforcement learning can be divided into model-free reinforcement learning and model-based reinforcement learning. Model-free reinforcement learning has recently been successfully applied to deep neural networks [35–37].

Value-based reinforcement learning. As we know, the Q-learning is a well-known value-based reinforcement learning algorithm, which finding an optimal action-value function $Q(s, a)$. Furthermore, using deep neural network to approximate the action-value function $Q(s, a; \theta)$

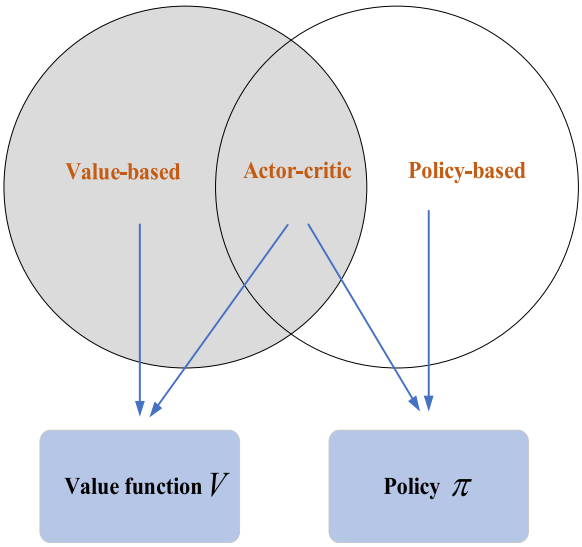


Fig. 1. The relationship of value-based and policy-based.

is called the DNQ [38] and asynchronous Q-learning [39]. The value-based reinforcement learning algorithms update the parameters by minimizing the mean-squared error loss.

Policy-based reinforcement learning. Literature [40,41] introduced the policy-based reinforcement learning. Compare with value-based reinforcement learning, policy-based algorithm can directly optimize the policy. The core idea of policy-based is that it maps a state to an action, and then optimizes the parameters to maximize long-term returns. The preponderance of policy-based algorithms is that they can have stochastic policies, which may lead to optimal policy for some problems. The variations of the policy-based methods include proximal policy optimization (PPO) [40], trust region policy optimization (TRPO) [41], and so on. The relationship of value-based and policy-based is shown in Fig. 1.

For deep Q-learning, the agent will adopt a neural network (NN) to define the Q function, which is written as $Q(s,a;\theta)$. The parameter θ is considered as weights of the NN. Through updating the parameter θ, the Q network is trained at each episode to approximate $Q(s,a)$. Although NNs allow great flexibility, they pay the cost for the stability of Q-learning. Recently, the deep Q network is proposed to replace Q function approximation by deep neural network, and it has been proved that it has more robust ability and higher performance. In order to transform the Q-learning into a deep Q-learning, some conditions should be satisfied:

(1) Using multiple layers convolution networks, because they can extract high-level features from the original input data by adopting hierarchical convolution filter to utilize local spatial relevance. This can make the agent learn from the past experience.

(2) Using experience replay. In deep Q-network, system store the data $e(t) = \{s(t), a(t), r(t), s(t+1)\}$ into a experience pool $D = \{e(1), \ldots, e(t)\}$. When training start, agent randomly samples from D to update the parameters of networks, instead of using the consecutive samples.

(3) Establish a target Q network in the training process, which is used to calculate the loss function. If only one network is used to estimate target Q value and Q value, which will be falling into feedback loops. Therefore, the weights of the target network are fixed and updated regularly to ensure stable training.

3. System model

In this section, we present the reinforcement learning within the MDP framework and then discuss the optimal solutions. In this paper, the asynchronous advantage actor–critic (A3C) algorithm is the product of combining policy and value Function. The advantages and disadvantages of policy-based approach are summarized as follows:

Advantages:

(1) More effective in high dimension and continuous action spaces;

(2) Better convergence properties;

(3) Can learn stochastic policies.

Disadvantages:

(1) Evaluation policies are usually not very efficient and high variance;

(2) Typically converge to a local rather than global optimal.

In the fundamental settings of reinforcement learning, there are agent, environment, action, state, reward and other basic elements. The agent interacts with the environment and generates trajectory, which changes the state $s \rightarrow s'$ by executing action. And agent will receive a reward from environment. By continuing these interactions, agents accumulate more and more experiences, and then update the policy.

For convenience, some defines is given as following:

(1) Stochastic policy π(s) determine agent's action, which means that its output is not a single action, but the distribution of probability over actions.

(2) $\pi(s|a)$ represents the probability of selecting action a with the state s.

We know that the objective of the agent is to maximize the reward. In probability distribution P, the expected return of value X can be written as:

$$E_P(X) = \sum_i P_i X_i \tag{1}$$

where, X_i is the all possible values of X, P_i is the probability of each value X_i. If exist a pool of values X, the probability of which was given by P, and we randomly selected a number of these, we would expect the mean of them to be $E_P(X)$. Let us define value function $V(s)$ of policy π:

$$V(s) = E_{\pi(s)}[R + \gamma V(s')] \tag{2}$$

Meanwhile, the action value function $Q(s,a)$ can be defined as:

$$Q(s,a) = R + \gamma V(s') \tag{3}$$

Furthermore, the Bellman equation recursive relationship of $Q(s,a)$ can be written as:

$$Q(s_t, a_t) = E[r(s_t, a_t) + \gamma E[Q(s_{t+1}, a_{t+1})]] \tag{4}$$

The policy gradient algorithm is probably the most popular reinforcement learning algorithm for continuous action. These algorithms update the parameters θ of the policy in the direction of the performance gradient $\nabla_\theta G(\theta)$. The $\nabla_\theta G(\theta)$ can be written as:

$$\nabla_\theta G(\theta) = E_{s\sim\rho^\pi, a\sim\pi}[\nabla_\theta log\pi_\theta(a|s) Q^\pi(s,a)] \tag{5}$$

where the ρ^π is the state distribution, which depends on the policy parameters. (5) is also called the policy gradient theorem [20]. The theorem simplifies the computation of gradient to a simple expected value, which has important practical value. Furthermore, (5) has been used to derive a variety of policy gradient algorithms [21].

The actor–critic is another architecture reinforcement learning algorithm based on the policy gradient, which consists of actor and critic two components. The actor updates the parameters θ of the stochastic policy $\pi_\theta(s)$ by (5). Instead of the unknown true action-value function $Q^\pi(s,a)$, an action-value function $Q^w(s,a)$ is used with parameter vector w. Using an appropriate policy evaluation, the critic estimates $Q^w(s,a) \approx Q^\pi(s,a)$. That is, the actor executes the action and critic evaluates it, saying that the choice of the action is good or bad. The gradient of actor parameter can be written as:

$$Loss = \frac{1}{N}\sum_t (r + maxQ(s_{t+1}, a_{t+1}) - Q(s_t, a_t))^2 \tag{6}$$

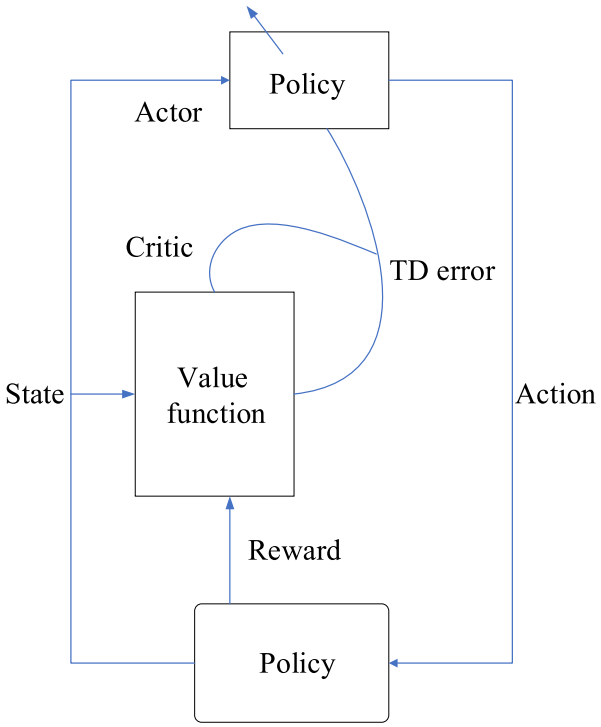


Fig. 2. The execution process of actor–critic.

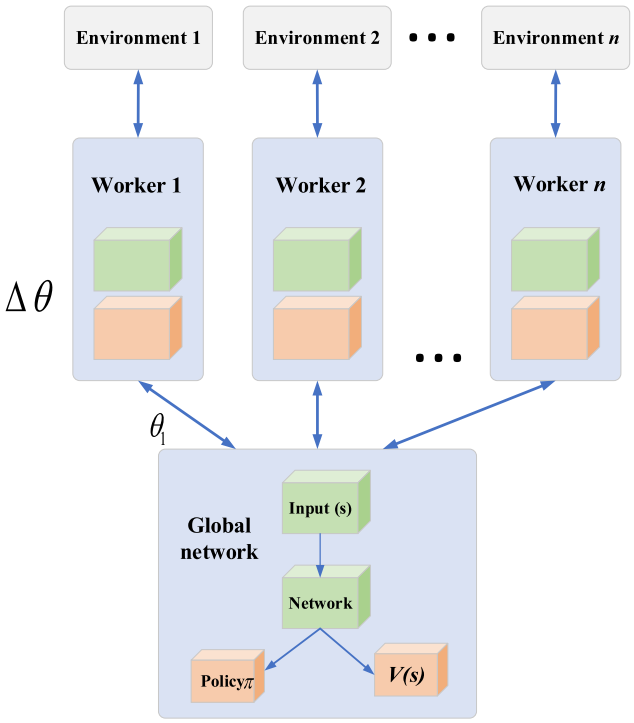


Fig. 3. The model of asynchronous advantage actor–critic.

However, if only a single agent is used to sample the samples, the samples we get are highly correlated, which will make machine learning model out of order.

The execution processes of actor–critic as shown in Fig. 2. The structure of actor net is the same as that defined by policy gradient. It is a double-layer full connected neural network. In order to achieve higher reward, actor needs a strategy: input state and output action. Critic net needs to feed back a TD value to the actor to determine whether the action selected by the actor is good or bad. If the TD value is large, it means that the surprise degree of the action selected by the current actor is high, and more appearances are needed to reduce the TD value.

Because the condition of machine learning is that sample is independent distributed and cannot be highly related. Therefore, a baseline $b(s) \approx V(s)$ is added to the value function to get a much lower variance estimate of the policy gradient. When using the approximation function as baseline, the system need two networks to estimate $V(s)$ and $Q^{\pi}(s,a)$. The $r_t - b_t$ used to scale the policy gradient, which can be considered as an estimate of the advantage of a_t in s_t. That is:

$$A\left(s_t, a_t\right) = Q\left(s_t, a_t\right) - V(s_t) \tag{7}$$

As we know, the iteration speed of policy gradient is very slow. In order to make full use of computing resources, the asynchronous Advantage Actor–Critic (A3C) method is used to maintain a policy π and a value function $V(s)$. According to (7), then:

$$A\left(s_t, a_t\right) = Q\left(s_t, a_t\right) - V\left(s_t\right) = r + \gamma V\left(s_{t+1}\right) - V(s_t) \tag{8}$$

As [42], we consider a virtualized vehicular network. The vehicular network mainly include MEC servers, vehicles and routers, road side units (RSUs), base stations (BSs), which are managed by the infrastructure providers (InPs). Let B be the sets of BSs and R be the sets of RSUs. $U = B \cup R = \{1, \ldots, u\}$ is the set of all the BSs and RSUs. $V = \{1, \ldots, v\}$ is the set of vehicles. $C = \{1, \ldots, c\}$ and $M = \{1, \ldots, m\}$ are the sets of caches and MEC servers, respectively.

An example of a use case in this network model is as follows. Suppose the vehicle sends video content requests to its associated virtual base station. In the first step, according to the vehicle and content information, the base station will check whether its associated cache has the requested content. If the information is yes, the cache will further check whether the version of the cached video content can be played and matched with the vehicle. The model of A3C is shown in Fig. 3.

As shown in Fig. 3, the system model of A3C has a main network and multiple workers. Each worker net is also a net of A2C. A3C mainly has two operations: pull and push. Pull action gives the parameters of the main network directly to the network in the workers. Push action uses the gradients in each worker to update the parameters of the main network. The global network has the global parameters and every worker has local net parameters. Local net can push the updated parameters to global net at a fixed time, and then get the comprehensive updated parameters from global net at a fixed time.

If the information is still yes, the base station sends the content of the request from the cache to the vehicle. Otherwise, the base station will select the current transmission content create a new task according to the size of content. Furthermore, the base station sends the new task to the MEC to finish the task. When the task is completed, the base station transfers transcoded content to the end user. Supposing that cache cannot search the match contents, the base station must search the content from internet, which will consume some backhaul resources. The resource request process is shown in Fig. 4

Virtual vehicular networks must be established logically. Each network contains of MEC servers, BSs, caches, RSUs. They can provide the high-speed computing capabilities for vehicles. Supposing that every wireless network belongs to different the infrastructure providers, and the license spectrum of different the infrastructure providers is orthogonal to eliminate interference.

The service providers (SPs) manage all the networks resource. SPs is denoted as $S = \{1, \ldots, s\}$. For a service provider, V_s is the set of vehicle, and one vehicle is allocated from V_s is denoted as v_s. All V_s are controlled by service providers. However, given a time period, one v_s only subscribes to one service provider. That is, $V = \cup_s V_s$ and $V_s \cap V_{s'} = \emptyset$, when $s \neq s'$. At time t, $a_{v_s,u}(t)$ is the association between

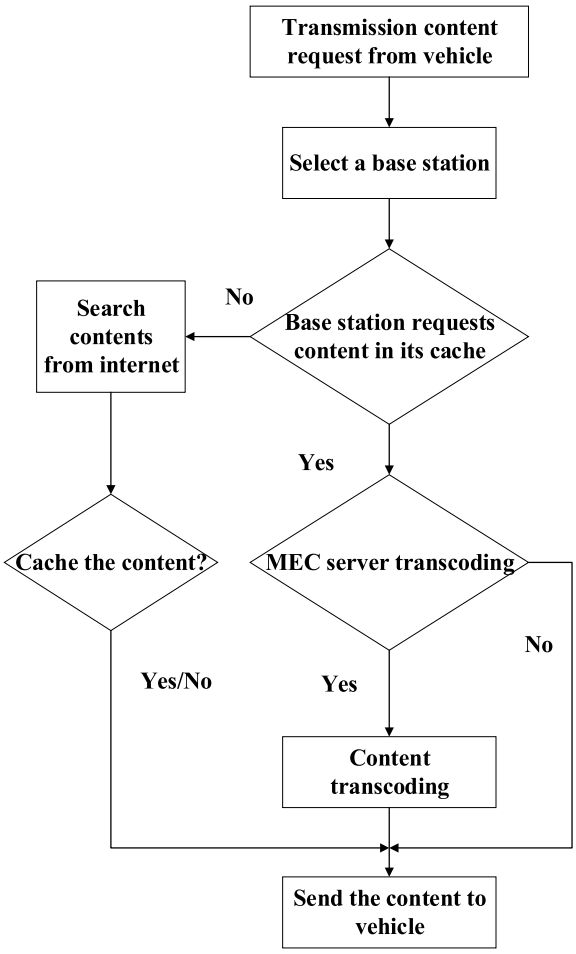


Fig. 4. The resource request process of vehicle network.

vehicle v_s and BS u. If $a_{v_s,u}(t) = 1$, that means v_s connects to k to request, otherwise, $a_{v_s,u}(t) = 0$. However, for every v_s can only access to one RSUs or BS at time t, thus:

$$\sum_{u \in U} a_{v_s,u}(t) = 1, if\, s \in S, v_s \in V_s \tag{9}$$

As we know, the network model consists of computation, communication and cache. Now, we describe the computation model. Let $W_{v_s} = \{o_{v_s}, q_{v_s}\}$ is the computation assignment, which is executed on v_s associated with MEC server m_k. o_{v_s} represents the size of the transfer content and q_{v_s} is the required number of CPU cycles. After the computation, base station k sent the transfer content back to v_s.

We define the computation capability of base station k as $\varphi^k_{v_s}$, which can be weighted by CPU computing power. In our network model, all MEC servers are assigned to base stations dynamically. Furthermore, multiple vehicles can share the same MEC server and access to the same base station at time t. Because of above two points, however, the network system cannot accurately predict the computation capability for v_s at time $t + 1$. Therefore, $\varphi^k_{v_s}$ is defined as a random variable, which denoted by $\varepsilon = \{\varepsilon_0, \ldots, \varepsilon_{N-1}\}$, where N is the number of available computation capability states. Furthermore, the computation capability realization of $\varphi^k_{v_s}$ denoted as $\Psi^k_{v_s}(t)$ at time t. The state transition probability matrix of base station is defined as:

$$\Theta^k_{v_s}(t) = [l_{xy}(t)]_{N \times N} \tag{10}$$

where $l_{xy}(t) = \Pr(\Psi^k_{v_s}(t+1) = y | \Psi^k_{v_s}(t) = x)$, and $x, y \in \varepsilon$.

The computation time of W_{v_s} at base station k is defined as:

$$T_{v_s,k} = \frac{q_{v_s}}{\Psi^k_{v_s}(t)} \tag{11}$$

Furthermore, the bit computed per second (also called the computation rate) is defined as:

$$r_{comp}(t) = a_{v_s,u} \frac{o_{v_s}}{T_{v_s,k}} = a_{v_s,u} \frac{\varphi^k_{v_s} o_{v_s}}{q_{v_s}} \tag{12}$$

However, the computing resources of server is limited, thus the constraint must be observed:

$$\sum_{s \in S} \sum_{v_s \in V_s} a_{v_s,u}(t), o_{v_s} \le O_u, u \in U \tag{13}$$

where O_u is the maximum content sizes, which simultaneously run on the MEC server.

Consider that the communication channels are realistic time-varying channels, thus we can define them as the finite-state Markov channels. As [43,44], the finite-state Markov channels are widely used to characterize the correlation structure of the fading process. Meanwhile, the finite-state Markov channels enable significant performance improvement compared with memoryless channel.

In the finite-state Markov channels, the received signal-to-noise ratio (RSNR) is an appropriate parameter, which enabled to represent the channel quality. Meanwhile, the RSNR associated with v_s and base station k can model as a random variable $\gamma^k_{v_s}$. Furthermore, $\gamma^k_{v_s}$ can be quantized and partitioned into L discrete levels: $L_0, L_2, \ldots, L_{u-1}$, which are satisfied:

$$\begin{cases} L_0, & if\ \gamma^*_0 \le \gamma^k_{v_s} \le \gamma^*_1 \\ L_1, & if\ \gamma^*_1 \le \gamma^k_{v_s} \le \gamma^*_2 \\ & \cdots \\ L_{u-1}, & if\ \gamma^k_{v_s} \ge \gamma^*_{u-1} \end{cases}$$

Each level corresponds to a Markov chain state, let $D = \{D_0, D_2, \ldots, D_{u-1}\}$ is the u-length state spaces. The instantaneous value of $\gamma^k_{v_s}$ can be defined as $\upsilon^k_{v_s}$. Furthermore, assume that the network communication has T time slots, which starts when v_s sends a request and ends when the result is obtained. Let $t \in \{0, 1, 2, \ldots, T-1\}$ be the time instant. Based on the transition probabilities, the RSNR $\upsilon^k_{v_s}$ varies from one state to another state. Let $\psi_{g_s h_s}(t)$ be the transition probabilities, that is, the probability from state g_s transfer to another state h_s at time t. The state transition probability matrix is defined as:

$$\Psi^k_{v_s}(t) = [\psi_{g_s h_s}(t)]_{L \times L} \tag{14}$$

where $\psi_{g_s h_s}(t) = Pr\left(Y^k_{v_s}(t+1) = h_s | Y^k_{v_s}(t) = g_s\right)$, and $g_s, h_s \in D$.

Furthermore, supposing the spectrum bandwidth is E Hz, and the bandwidth allocated to the base station k is E_k Hz. The backhaul capacity is Y bps, and the capacity allocated to the base station k is Y_k bps. $B^k_{v_s}$ is denoted as the spectrum bandwidth, which is no interference to different vehicles. Let $e_{v_s,k}(t)$ be the achievable spectrum efficiency. The communication rate can be written as:

$$r_{comm}(t) = a_{v_s,u}(t)\, B^k_{v_s}(t) e_{v_s,k}(t) \tag{15}$$

However, the sum rate of v_s cannot exceed its backhaul capacity, thus:

$$\sum_{s \in S} \sum_{v_s \in V_s} r_{comm}(t) \le Y_k \tag{16}$$

Similarly, the data rate of v_s cannot exceed the total backhaul capacity, thus:

$$\sum_{k \in U} \sum_{s \in S} \sum_{v_s \in V_s} r_{comm}(t) \le Y \tag{17}$$

4. The asynchronous advantage actor–critic algorithm

In this section, according to the system network model, we model the resource allocation problem as a DRL Markov decision process. In the system, the network has B base stations, E MEC servers. They are managed by a mobile virtual network operator.

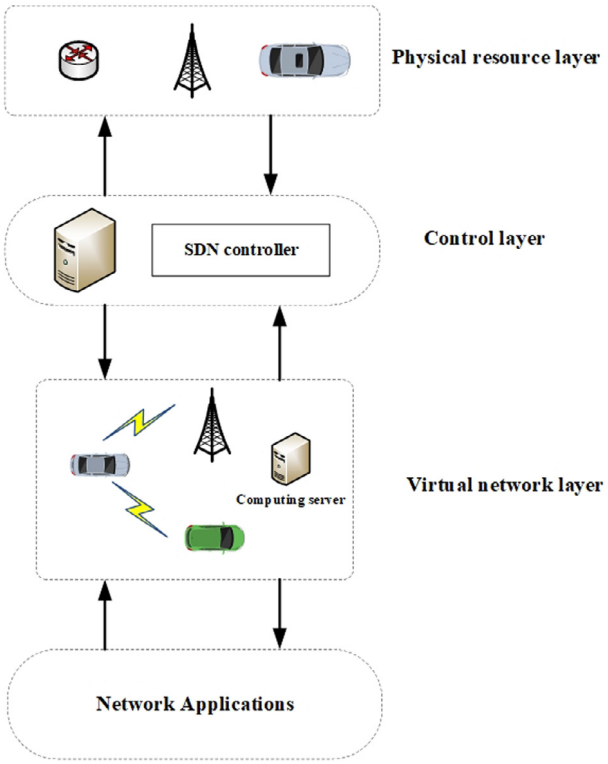


Fig. 5. The model framework of integrated communication and computation with virtualized and software-defined networks.

Based on realistic situation, the computation abilities of MEC servers and downlink channel conditions of BSs are all dynamically change, and the mobile virtual network operator has a large number of system states, which has to make a decision on resource allocation according to the system's current state. However, it is difficult to solve this complex task by traditional methods. The asynchronous advantage actor–critic is a new technique that is capable of receiving complex high-dimensional data as input, and generates the optimal action for each input data. By taking the advantages of the asynchronous advantage actor–critic, the mobile virtual network operator can manage the network system resource effectively. The model framework of integrated communication and computation with virtualized and software-defined networks is show in Fig. 5.

As we know, the network virtualization can be become an effective strategy to manage infrastructure and spectrum resources. According to different service functions, QoS and wireless network requirements, the physical resource layer network can be virtualized into multiple virtual networks through hypervisor. Furthermore, because virtualization can share the network resources, the operation cost can be significantly reduced. Therefore, It is beneficial to manage computation and communication resources in virtual networks based on flexibility and principles of software-defined networking.

In Fig. 5 model, a large amount of security data and information will be exchanged between RSU and vehicle transceivers in various vehicle applications. Due to low quality network links and high mobility, it is a challenge to use traditional host-centric IP-based methods to transmit large amounts of data in vehicle networks. Recent advances in information-centric networking can be used to solve this issue. Particularly, the cache transmission content at network edge (such as, road side units and base station) is becoming one of the key technologies in next generation virtual networks.

Algorithm 1: Asynchronous advantage actor-critic algorithm in the virtualized network.

Input: the state s and action a of environment
Initialize the Environment Env;
Initialize Global parameters θ, θ_v;
Initialize Global shared counter $T=0$;
Initialize Thread-specific parameters θ', θ'_v;
Initialize thread counter $t \rightarrow 1$;
Output: optimal policy π
repeat
Reset cumulative gradients: $d\theta \leftarrow 0, d\theta_v \leftarrow 0$
Synchronize parameters: $\theta' \leftarrow \theta, \theta'_v \leftarrow \theta_v$
$t_{start} = t$
Get observation from Env
Get state $x_{v_s}(t)$
repeat
Policy choice: $a_{v_s}(t)$ according to:
$\pi(a_{v_s}(t)|x_{v_s}(t); \theta')$
By (20) Receive reward $R_{v_s}(t)$ and obtain the next state $x_{v_s}(t+1)$
t = t + 1
T = T + 1
until $x_{v_s}(t)$ terminal or $t - t_{start} = t_{max}$
If $x_{v_s}(t)$ is the terminal state **then**
$R_{v_s}(t) = 0$
else
$R_{v_s}(t) = V(x_{v_s}(t)|\theta'_v)$
end if
for $i \in \{t-1, \ldots, t_{max}\}$ **do**
update:
$R_{v_s}(t) \leftarrow (r_{comm}(t) + r_{comp}(t) + \gamma * R_{v_s}(t))$
Accumulate gradient θ':
$d\theta \leftarrow d\theta + \nabla_{\theta'} log\, \pi(a_{v_s}(t)|x_{v_s}(t); \theta')\,(R_{v_s}(t) - V(x_{v_s}(t)|\theta'_v))$
Accumulate gradient θ'_v:
$d\theta_v \leftarrow d\theta_v + \partial(R_{v_s}(t) - V(x_{v_s}(t)|\theta'_v))^2/\partial\theta'_v$
end for
Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$
until $T > T_{max}$

The mobile virtual network operator collects the status from each BS and MEC server. Meanwhile, it combines the whole information into a system state. Furthermore, the mobile virtual network operator sends the state to the agent, and receives a feedback of the optimal policy, that is arranges a certain vehicle from the resource pools. When selecting the action, the mobile virtual network operator sends information which virtualized network resource is free and it can be used.

In the A3C algorithm, the experience of agents will be stored in the replay memory. The net parameters θ are updated at every time instant t. The training algorithm of A3C is shown in Algorithm 1.

To gain the optimal policy of the virtual network system, we must identify the reward, state and action function in our A3C model.

4.1. System state

Let $k \in \{1, 2, \ldots, K\}$ be the state of an available BS and $m \in \{1, 2, \ldots, M\}$ be the available MEC server, which is determined by $Y^k_{v_s}(t)$ of the random variable $\gamma^k_{v_s}$, $\Psi^k_{v_s}(t)$ of the random variable $\varphi^k_{v_s}$. Therefore, the state vector can be described as follows:

$$x_{v_s}(t) = [Y^0_{v_s}(t), Y^1_{v_s}(t), \ldots, Y^K_{v_s}(t), \Psi^0_{v_s}(t), \Psi^1_{v_s}(t), \ldots, \Psi^K_{v_s}(t)] \quad (18)$$

4.2. System action

In virtual vehicle network system, the agent must determine which base station is assigned to the vehicle, and whether the computation assignment should be offloaded to the MEC server.

The current composite action $a_{v_s}(t)$ can be defined as:

$$a_{v_s}(t) = \{a^{comm}_{v_s}(t), a^{comp}_{v_s}(t)\} \quad (19)$$

where $a_{v_s}^{comm}, a_{v_s}^{comm}$ are defined as following:

(1) Define row vector

$a_{v_s}^{comm} = [a_{v_s,1}^{comm}(t), a_{v_s,2}^{comm}(t), \ldots, a_{v_s,K}^{comm}(t)]$, where $a_{v_s}^{comm}(t)$ denotes the communication control of the kth BS for v_s. Let $a_{v_s,k}^{comm}(t) \in \{0,1\}$, where $a_{v_s,k}^{comm}(t) = 0$ represents the BS k is not connected at time t, on the contrary, $a_{v_s,k}^{comm}(t) = 1$ represents the BS is connected. Note that, $\sum_{k\in K} a_{v_s,k}^{comm}(t) = 1$.

(2) Define row vector

$a_{v_s}^{comp}(t) = [a_{v_s,1}^{comp}(t), a_{v_s,2}^{comp}(t), \ldots, a_{v_s,M}^{comp}(t)]$, where $a_{v_s}^{comp}(t)$ denotes the offloading control of the mth MEC server for v_s. Let $a_{v_s,m}^{comp}(t) \in \{0,1\}$, where $a_{v_s,m}^{comp} = 0$ represents the assignment is not offloaded to server in time t, on the contrary, $a_{v_s,m}^{comp}(t) = 1$ represents the assignment is offloaded to mth MEC server. Note that, $\sum_{m\in M} a_{v_s,m}^{comm}(t) = 1$.

4.3. Reward model

For the network system, we defined the total revenue of the mobile virtual network operator as the system's rewards. The mobile virtual network operator rents the backhaul bandwidth and wireless spectrum from the infrastructure providers, and assigns the resources to virtual service providers. The mobile virtual network operator is required pay wireless spectrum cost to infrastructure providers, which is represented as δ_k per Hz for base station k. In addition, the mobile virtual network operator also required to pay the computation cost to infrastructure providers when a resource allocation task is performed. For mth MEC server, the energy consumption cost can be denoted as η_m per Joule.

Furthermore, the mobile virtual network operator charges v_s for accessing to the virtual network systems can be denoted as τ_{v_s}. If the vehicles pay the fee, they can offload the task by accessing to the virtual network. In addition, the cost for v_s can be defined as ϕ_{v_s}. The backhaul cost is denoted as κ_{v_s}.

The reward is the mobile virtual network operator's revenue, which can model as the function of RSNR of the computation capability and the access wireless link state. Hence, the reward function is written as:

$$\begin{aligned} R_{v_s}(t) &= \sum_{m=1}^{M} R_{v_s,m}^{comp}(t) + \sum_{k=1}^{K} R_{v_s,k}^{comm}(t) \\ &= \sum_{m=1}^{M} a_{v_s,m}^{comp}(t)\left(\phi_{v_s} r_{comp}(t) - \eta_m q_{v_s} e_m\right) + \\ &\quad \sum_{k=1}^{K} a_{v_s,k}^{comm}(t)\left(\tau_{v_s} r_{comm}(t)\left(1 - w a_{v_s,k}^{comm}(t)\right) - \delta_k B_{v_s,k}\right) \\ &= \sum_{m=1}^{M} a_{v_s,m}^{comp}(t)\left(\phi_{v_s}\frac{\Psi_{v_s}^{k} o_{v_s}}{q_{v_s}} - \eta_m q_{v_s} e_m\right) + \\ &\quad \sum_{k=1}^{K} a_{v_s,k}^{comm}(t)\left(\tau_{v_s} B_{v_s}^{k}(t) e_{v_s,k}(t)\left(\left(1 - w a_{v_s,k}^{comm}(t)\right) - \delta_k B_{v_s,k}\right)\right) \end{aligned} \tag{20}$$

where $\sum_{m=1}^{M} R_{v_s,m}^{comp}(t)$ is the computation revenue, $\phi_{v_s}\frac{\Psi_{v_s}^{k} o_{v_s}}{q_{v_s}}$ represents the revenue of the mobile virtual network operator for execute offloading on MEC servers. $\eta_m q_{v_s} e_m$ represents the energy consumption cost. $\sum_{k=1}^{K} R_{v_s,k}^{comm}(t)$ is the communication revenue, $\tau_{v_s} r_{comm}\left(1 - w a_{v_s,k}^{comm}(t)\right)$ represents the revenue of the mobile virtual network operator allowed vehicles access virtual base stations and charge for providing services, however, if the transmitted content performs transcoding operation, the revenue decreases. $R_{v_s}(t)$ is the immediate reward of the system at time t, which represents the revenue of the mobile virtual network operator when the agent of system select action $a_{v_s}(t)$ in state $x_{v_s}(t)$. The objective of using A3C algorithm is to obtain an optimal policy to maximize the total reward of mobile virtual network operator. Thus the cumulative reward can be defined as:

$$R_{v_s}^{T} = max E\left[\sum_{t=0}^{t=T-1} R_{v_s}(t)\right] \tag{21}$$

Table 2
Setting of simulation parameters.

Parameter	Value
Data transmission frequency $B_{v_s,k}$	30 MHz
The unit charging-price τ_{v_s}	30 unit/Mbps
The unit paid-price δ_k	5 unit/MHz
The unit charging-price ϕ_{v_s}	15 unit/MHz
The CPU cycles q_{v_s}	0.5 G
The content size o_{v_s}	2 M bits
The spectrum efficiency	[1,3] bps/Hz
The unit energy consumption of server η_m	50 units/J
The effect factor w	0.3
The unit energy consumption of CPU e_m	1 W/GHz
The computation capability $\Psi_{v_s}^{k}$	[1,3,5,7] GHz

5. Experiment and analysis

Reinforcement learning tasks are usually described by Markov decision process. That is, the goal of reinforcement learning is to obtain the optimal decision for a given state s.

Constructing a practical problem as a reinforcement learning task often needs to be divided into different number of Markov states according to specific needs. For instance, in LTE system, the base station attempts to provide the user with the highest correct decodable communication rate. Hence, user needs to feedback a 4 bit information to base station, i.e., channel quality indicator (CQI), which represents a number between 0 and 15, indicating a downlink rate value suitable for user. The Markov model of LTE system channel can be expressed by 16 states.

In this paper, the number of states in the Markov model of communication, computation depends on the actual requirements. All vehicles and base stations are randomly distributed. In this experiment, there are 4 service providers, one mobile virtual network operator, 4 base stations, 4 MEC servers. For convenience, each base station bandwidth will be normalized. The network communication channels follow the Markov decision process. The partial parameters are summarized in Table 2.

For reinforcement learning, we use a graphics processing unit (GPU), which is Nvidia GTX 1060. The CPU is i7-9750H. The programming environment is tensorflow-gpu-1.11.1, python-3.6 on Linux system.

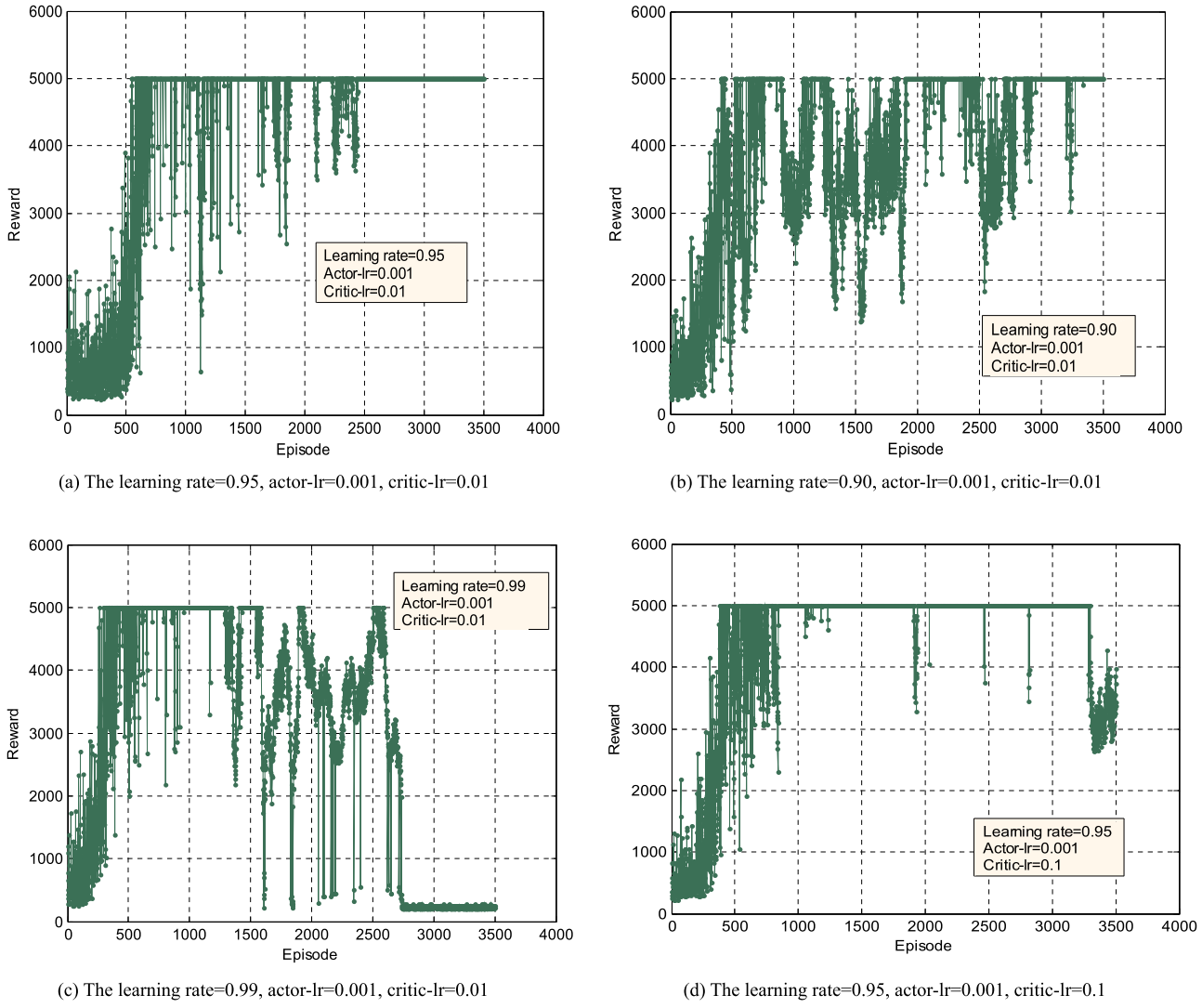
In this simulation, the channel states consist of two kinds: good (the spectrum efficiency $e_{v_s,k}(t) = 3$) or bad (the spectrum efficiency $e_{v_s,k}(t) = 1$). The transition probability of system state invariance is 0.8, otherwise is 0.2. Meanwhile, the MEC servers computing state also follow the Markov decision process. Let the computing power state of MEC servers is divided into four states. The state transition probability matrix is written as:

$$\Theta = \begin{bmatrix} 0.6 & 0.2 & 0.15 & 0.05 \\ 0.05 & 0.6 & 0.2 & 0.15 \\ 0.15 & 0.05 & 0.6 & 0.2 \\ 0.2 & 0.15 & 0.05 & 0.6 \end{bmatrix}$$

5.1. Parametric analysis

The influence of different parameters of the asynchronous advantage actor–critic (A3C) learning algorithm are shown in Fig. 5. The influence of the learning rate of A3C can be inferred from Fig. 6(a), (b), (c).

From Fig. 6(a), (b) and (c), we can see that the actor-lr and critic-lr are on the whole the same. When the learning rate is 0.95, we can see that although the result has a slight fluctuations, it has a good convergence effect. When the learning rate is 0.90, the reward curve is always in a state of fluctuation and difficult to converge. However,



(a) The learning rate=0.95, actor-lr=0.001, critic-lr=0.01

(b) The learning rate=0.90, actor-lr=0.001, critic-lr=0.01

(c) The learning rate=0.99, actor-lr=0.001, critic-lr=0.01

(d) The learning rate=0.95, actor-lr=0.001, critic-lr=0.1

Fig. 6. The influence of different parameters of the asynchronous advantage actor–critic (A3C) learning algorithm.

when the learning rate is 0.99, the reward curve is always in a state of fluctuation and cannot converge eventually.

Therefore, we can conclude that the learning rate in A3C reinforcement learning algorithm cannot be too large or too small, otherwise the system cannot learn the optimal policy.

In this paper, due to the algorithm consists of actor network and critic network. Therefore, we consider the influence of the parameters of actor-lr and critic-lr. From Fig. 6(a) and (d), we can see that the learning rate is the same, and the critic-lr are the 0.01 and 0.1, respectively. However, when the critic-lr too large, the result is easy to fall into local optimal solution and cannot obtain the optimal policy.

5.2. Performance analysis

In this paper, to describe the experiment, 2 schemes are proposed:

(1) The system state contains MEC computing and communication: The network states of MEC servers are considered to be static and will not be changed dynamically.

(2) The virtual network allocation strategy without MEC servers: the scheme does not consider MEC offloading, and the vehicles can only execute computational tasks locally.

The convergence performance of above two schemes is show in Fig. 7. According to Fig. 7, we can see that the reward of the virtual network system is very low at the beginning. However, with the increase of the iterations, the average reward will reach a stable value. In the scheme, which the system state contains MEC computing and communication, the stable value is 5000. And if the scheme without the MEC offloading, the stable value is 4000. Fig. 6 shows the convergence performance of the integrated communication and computation in our work. Furthermore, we also can see that the reward of the scheme without MEC offloading is less.

The effect of the content size is shown in Fig. 8. From Fig. 8, we can see that the reward of the proposed reinforcement learning algorithm without MEC server computation deceases with the increase of o_{v_s}. This is because the larger content sizes increasing the cost of transmission content, resulting in lower gain for caching utilities, thereby reducing the reward. In contrast, the reward of the scheme with MEC server increases when the content size o_{v_s} is increasing, because the larger content size will increase the computing charge on MEC server, resulting in higher returns of computing revenue, thereby increases the system reward.

The reward effect of τ_{v_s} is shown in Fig. 9. From Fig. 9, we can observed that the disparity without MEC server. When the vehicles access to the virtualized vehicle wireless networks, the MEC server will reduce the cost. When τ_{v_s} increasing, virtualization is accounts for an increasing proportion of total revenue. Therefore, MEC offloading is less executed. On the contrary, if without MEC server, the reward cannot increasing with the increase of τ_{v_s}.

The reward effect of the unit price for performing MEC ϕ_{v_s} is shown in Fig. 10. According to Fig. 10, we can know that the reward of

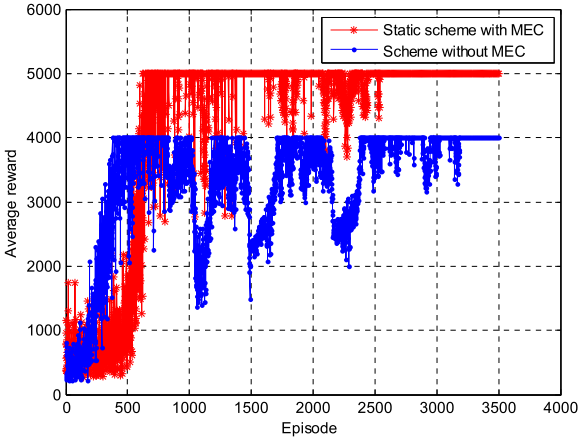


Fig. 7. Convergence performance of different schemes.

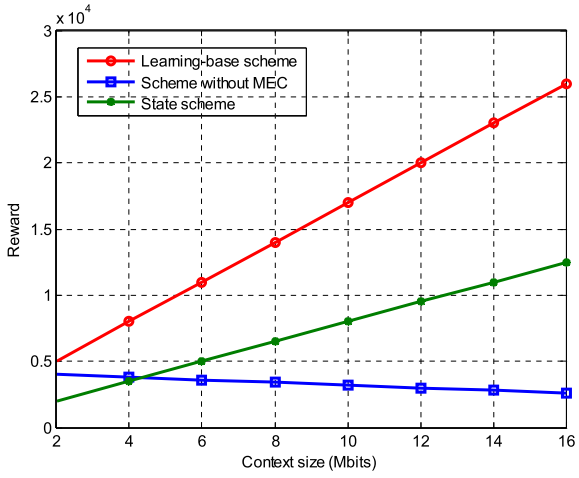


Fig. 8. The reward effects of o_{v_s}.

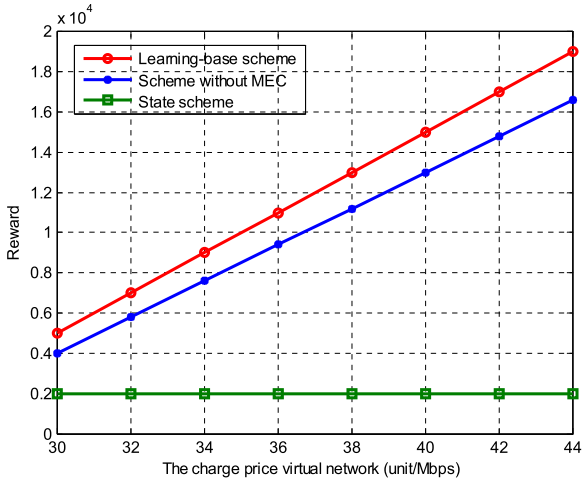


Fig. 9. The reward effect of τ_{v_s}.

learning-base scheme and static scheme keeps improving when the ϕ_{v_s} is increasing, and the gap between them is gradually increasing. This is because the MEC service price ϕ_{v_s} increases, and the computation revenue increases accordingly. Because learning-base scheme has network virtualization, it can provide users with better communication quality network, thus achieving higher communication benefits.

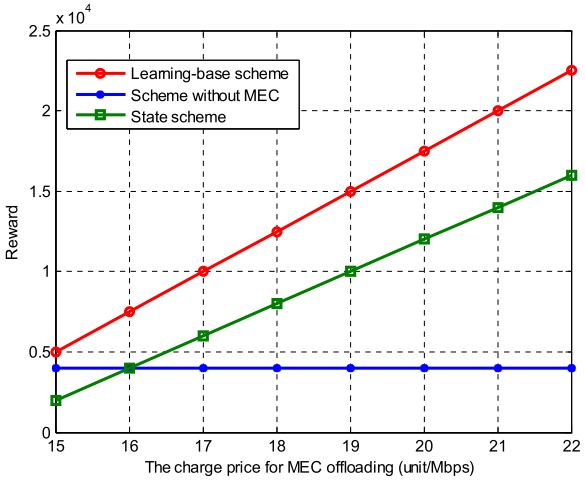


Fig. 10. The reward effect of ϕ_{v_s}.

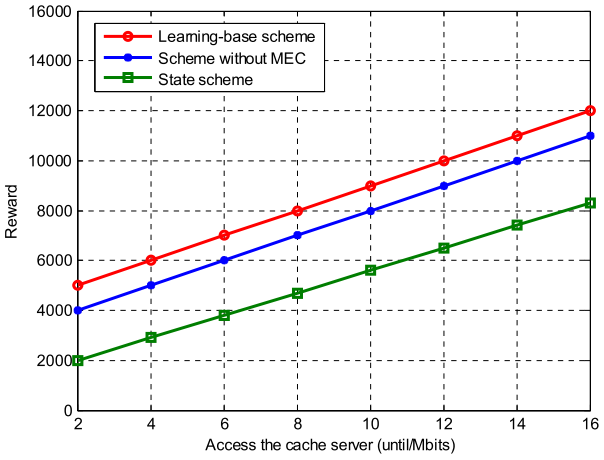


Fig. 11. The reward effect of using the cache.

The reward effect of using the cache is shown in Fig. 11. From Fig. 11, we can see that the reward of all schemes is decreasing when without using the cache server. However, the reward curve of learning-base scheme, without MEC scheme and static scheme is parallel increasing, which means the increasing of cache has not effect on without MEC and static scheme.

6. Conclusion and future work

We studied the learning-based network traffic control and distribution scheme of vehicular networks. Based on the asynchronous advantage actor–critic (A3C) learning algorithm, we proposed an integrated scheme, which enable allocate communication and computing resources to improve the performance of next generation wireless networks. In this scheme, we modeled the resource allocation strategy as a reinforcement learning Markov decision processes and using function approximation without making assumptions about the optimization goal or reducing the complexity of resource allocation computing. The experiment results under different parameters are given to verify the effectiveness of our proposed algorithm. Future work is considering another novel and effective reinforcement learning algorithm to solve the next generation network communication problem.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Miaojiang Chen: Data curation, Software, Visualization, Writing - original draft. **Tian Wang:** Conceptualization. **Kaoru Ota:** Funding acquisition, Validation. **Mianxiong Dong:** Project administration, Resources. **Ming Zhao:** Supervision, Writing - review & editing. **Anfeng Liu:** Conceptualization, Methodology, Funding acquisition.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (61772554, No. 61572526, 61572528), by JSPS KAKENHI Grant Numbers JP16K00117, JP19K20250, Leading Initiative for Excellent Young Researchers, MEXT, Japan and KDDI Foundation.

References

[1] M. Huang, W. Liu, T. Wang, A. Liu, S. Zhang, A cloud-MEC collaborative task offloading scheme with service orchestration, IEEE Internet Things J. (2019) http://dx.doi.org/10.1109/JIOT.2019.2952767.

[2] M. Chen, Y. Hao, C. Lai, D. Wu, Y. Li, K. Hwang, Opportunistic task scheduling over co-located clouds in mobile environment, IEEE Trans. Serv. Comput. 11 (3) (2018) 549–561.

[3] C. Liang, F.R. Yu, X. Zhang, Information-centric network function virtualization over 5G mobile wireless networks, IEEE Netw. 29 (3) (2015) 68–74.

[4] T. Wang, H. Ke, X. Zheng, K. Wang, A. Sangaiah, A. Liu, Big data cleaning based on mobile edge computing in industrial sensor-cloud, IEEE Trans. Ind. Inf. (2019) http://dx.doi.org/10.1109/TII.2019.2938861.

[5] Z.P. Cai, X. Zheng, J.G. Yu, A differential-private framework for urban traffic flows estimation via taxi companies, IEEE Trans. Ind. Inf. (2019) http://dx.doi.org/10.1109/TII.2019.2911697.

[6] F. Xiao, X. Xie, Z. Jiang, L. Sun, R. Wang, Utility-aware data transmission scheme for delay tolerant networks, Peer-to-Peer Netw. Appl. 9 (5) (2016) 936–944.

[7] X. Liu, A. Liu, T. Wang, K. Ota, M. Dong, Y. Liu, Z. Cai, Adaptive data and verified message disjoint security routing for gathering big data in energy harvesting networks, J. Parallel Distrib. Comput. 135 (2020) 140–155.

[8] M. Tao, W. Yu, W. Tan, et al., Communications, caching, and computing for content-centric mobile networks: part 1 [guest editorial], IEEE Commun. Mag. 54 (8) (2016) 14–15.

[9] M. Chen, Y. Hao, H. Gharavi, V. Leung, Cognitive information measurements: A new perspective, Inf. Sci. 505 (2019) 487–497.

[10] K. Xie, J.N. Cao, X. Wang, J.G. Wen, Optimal resource allocation for reliable and energy efficient cooperative communications, IEEE Trans. Wireless Commun. 12 (10) (2013) 4994–5007.

[11] M. Chen, Y. Hao, Label-less learning for emotion cognition, IEEE Trans. Neural Netw. Learn. Syst. (2019) http://dx.doi.org/10.1109/TNNLS.2019.2929071.

[12] T. Degris, P.M. Pilarski, R.S. Sutton, Model-free reinforcement learning with continuous action in practice, in: American Control Conference, 2012, pp. 2177–2182.

[13] Q. Li, A. Liu, T. Wang, M. Xie, N. Xiong, Pipeline slot based fast rerouting scheme for delay optimization in duty cycle based m2m communications, Peer-to-Peer Netw. Appl. 12 (6) (2019) 1673–1704.

[14] S. Andreev, O. Galinina, A. Pyattaev, et al., Exploring synergy between communications, caching, and computing in 5G-grade deployments, IEEE Commun. Mag. 54 (8) (2016) 60–69.

[15] Y. Liu, X. Liu, A. Liu, N. Xiong, F. Liu, A trust computing based security routing scheme for cyber physical systems, ACM Trans. Intell. Syst. Technol. (TIST) 10 (6) (2019) http://dx.doi.org/10.1145/3321694.

[16] Q. Liu, P.L. Hou, G.J. Wang, T. Peng, S.B. Zhang, Intelligent route planning on large road networks with efficiency and privacy, J. Parallel Distrib. Comput. 133 (2019) 93–106.

[17] Z.B. Xiong, W. Li, Q.L. Han, Z.P. Cai, Privacy-preserving auto-driving: a GAN-based approach to protect vehicular camera data, in: 19th IEEE International Conference on Data Mining (ICDM), 2019.

[18] D.Y. Zhang, Y. Qiao, L. She, R.Y. Shen, J. Ren, Y.X. Zhang, Two time-scale resource manageme0nt for green internet of things networks, IEEE Internet Things J. 6 (1) (2019) 545–556.

[19] T. Wang, D. Zhao, S. Cai, W. Jia, A. Liu, Bidirectional prediction based underwater data collection protocol for end-edge-cloud orchestrated system, IEEE Trans. Ind. Inf. (2019) http://dx.doi.org/10.1109/TII.2019.2940745.

[20] Y. Liu, A. Liu, X. Liu, M. Ma, A trust-based active detection for cyber-physical security in industrial environments, IEEE Trans. Ind. Inf. 15 (12) (2019) 6593–6603.

[21] Q. Liu, Y. Tian, J. Wu, T. Peng, G.J. Wang, Enabling verifiable and dynamic ranked search over outsourced data, IEEE Trans. Serv. Comput. (2019) http://dx.doi.org/10.1109/TSC.2019.2922177.

[22] Q. Zheng, K. Zheng, H. Zhang, V.C.M. Leung, Delay-optimal virtualized radio resource scheduling in software-defined vehicular networks via stochastic learning, IEEE Trans. Veh. Technol. 65 (4) (2016) 7857–7867.

[23] X. Deng, J. Luo, L. He, et al., Cooperative channel allocation and scheduling in multi-interface wireless mesh networks, Peer-to-Peer Netw. Appl. 12 (1) (2019) 1–12.

[24] H. Xiong, Y. Zhao, L. Peng, H. Zhang, K. Yeh, Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing, Future Gener. Comput. Syst. 97 (2019) 453–461.

[25] Y. Liu, Z. Zeng, X. Liu, X. Zhu, M. Bhuiyan, A novel load balancing and low response delay framework for edge-cloud network based on SDN, IEEE Internet Things J. (2019) http://dx.doi.org/10.1109/JIOT.2019.2951857.

[26] F.R. Yu, C. Liang, Y. He, N. Zhao, Energy-efficient resource allocation in software-defined mobile networks with mobile edge computing and caching, in: Proc. IEEE Information Workshops, 2017.

[27] C. Fang, F.R. Yu, T. Huang, et al., A survey of green information-centric networking: research issues and challenges, IEEE Commun. Surv. Tutor. 17 (3) (2015) 1455–1472.

[28] L. Hu, A. Liu, M. Xie, T. Wang, UAVs joint vehicles as data mules for fast codes dissemination for edge networking in smart city, Peer-to-Peer Netw. Appl. 12 (6) (2019) 1550–1574.

[29] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. van den Driessche, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.

[30] S. Sukhbaatar, R. Fergus, et al., Learning multiagent communication with back propagation, in: Advances in Neural Information Processing Systems, 2016, pp. 2244–2252.

[31] R. Lowe, Y. Wu, A. Tamar, et al., Multi-agent actor-critic for mixed cooperative-competitive environments, in: Advances in Neural Information Processing Systems, Vol. 30, 2017.

[32] G. Tesauro, Extending q-learning to general adaptive multi-agent systems, in: Advances in Neural Information Processing Systems, 2014, pp. 871–878.

[33] J.N. Foerster, N. Nardelli, G. Farquhar, P.H.S. Torr, P. Kohli, S. Whiteson, Stabilising experience replay for deep multi-agent reinforcement learning, 2017, CoRR, abs/1702.08887.

[34] J.B. Wang, Z.P. Cai, J.G. Yu, Achieving personalized k-anonymity based content privacy for autonomous vehicles in CPS, IEEE Trans. Netw. Sci. Eng. (2019) http://dx.doi.org/10.1109/TII.2019.2950057.

[35] Z. Wang, T. Schaul, M. Hessel, et al., Dueling network architectures for deep reinforcement learning, 2015, Arxiv preprint arxiv:1511.06581.

[36] M. Hausknecht, P. Stone, Deep reinforcement learning in parameterized action space, 2015, Ariv preprint arxiv:1511.04143.

[37] T.P. Lillicrap, J.J. Hunt, A. Pritzel, et al., Continuous control with deep reinforcement learning, 2015, Arxiv preprint arxiv:1509.02971.

[38] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.

[39] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, Asynchronous methods for deep reinforcement learning, in: Proc. Int. Conf. Mach. Learn., 2016, pp. 1928–1937.

[40] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv:1707.06347, [Online]. Available: https://arxiv.org/abs/1707.06347.

[41] J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz, Trust region policy optimization, in: Proc. Int. Conf. Mach. Learn., 2015, pp. 889–1897.

[42] Y. He, N. Zhao, H.X. Yin, Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach, IEEE Trans. Veh. Technol. 67 (1) (2018) 44–55.

[43] Y. Wei, F.R. Yu, M. Song, Distributed optimal relay selection in wireless cooperative networks with finite-state Markov channels, IEEE Trans. Veh. Technol. 59 (5) (2010) 2149–2158.

[44] H. Wang, F.R. Yu, L. Zhu, T. Tang, B. Ning, Finite-state Markov modeling for wireless channels in tunnel communication-based train control (CBTC) systems, IEEE Trans. Intell. Transp. Syst. 15 (3) (2014) 1083–1090.