HIGH TECHNOLOGY LETTERS | Vol. 29 No. 3 | Sep. 2023 | pp. 295-304

doi:10.3772/j.issn.1006-6748.2023.03.008

Deep reinforcement learning based task offloading in blockchain enabled smart city①

JIN Kaiqi (金凯琦), WU Wenjun, GAO Yang, YIN Yufen, SI Pengbo②

(Faculty of Information Technology, Beijing University of Technology, Beijing 100124, P. R. China)

Abstract

With the expansion of cities and emerging complicated application, smart city has become an intelligent management mechanism. In order to guarantee the information security and quality of service (QoS) of the Internet of Thing (IoT) devices in the smart city, a mobile edge computing (MEC) enabled blockchain system is considered as the smart city scenario where the offloading process of computing tasks is a key aspect infecting the system performance in terms of service profit and latency. The task offloading process is formulated as a Markov decision process (MDP) and the optimal goal is the cumulative profit for the offloading nodes considering task profit and service latency cost, under the restriction of system timeout as well as processing resource. Then, a policy gradient based task offloading (PG-TO) algorithm is proposed to solve the optimization problem. Finally, the numerical result shows that the proposed PG-TO has better performance than the comparison algorithm, and the system performance as well as QoS is analyzed respectively. The testing result indicates that the proposed method has good generalization.

Key words: mobile edge computing (MEC), blockchain, policy gradient, task offloading

0 Introduction

Developed with the help of big data and Internet of Thing (IoT)[1], the smart city is capable of the intelligence management in terms of smart grid, smart community, smart hospitality, smart transportation, smart warehouse and smart healthcare[2]. However, most of the IoT devices in smart city are lack of integrated security mechanisms and vulnerably exposed in the open area, which brings challenges such as data privacy and security into smart city[3-4].

Fortunately, blockchain, which is an intelligent decentralized system using distributed databases to identify, disseminate and record information, makes it possible to guarantee the data security and establish a reliable network system[5]. The core techniques of blockchain are consensus mechanism and smart contract, which are responsible for the development of trust among distributed nodes and autonomic management of the system, respectively. Due to the intervention of blockchain, the security access[6], data privacy and security[7-8] and data integrity [9] can be ensured in smart city.

However, it is wildly acknowledged that the operation of consensus mechanism and smart contract of blockchain requires substantial computation resource. To further facilitate the implementation of blockchain in smart city, mobile edge computing (MEC) servers are always deployed with the access points to provide highly expandable computation resource[10] and reduce the service latency[11].

In the MEC-enhanced smart city, the wireless resource allocation problem has been wildly studied. The dynamic scheduling problem is researched in the Internet of Everything, and a multiple scheduling algorithm is proposed adopting round robin scheduling, proportional fair scheduling and priority based scheduling, enhancing the resource allocation efficiency[12]. The joint user association and data rate allocation problem is modeled to optimize the service delay and power consumption, and correspondingly solved by iterative algorithm and bisection algorithm[13]. Task offloading is the most typical resource management problem in MEC, and it is also widely studied in the scenario of smart city. In a multi-user and multi-server MEC scenario, a

① Supported by the National Natural Science Foundation of China (No. 62001011) and the Natural Science Foundation of Beijing Municipality (No. L192002).

② To whom correspondence should be addressed. E-mail: cathy_gy@outlook.com.

Received on Oct. 8, 2022

joint task offloading and scheduling problem is studied aiming to minimize task execution latency and solved by a heuristic algorithm[14]. Aiming at the maximization of offloading accomplish rate, problems restricted by the power consumption of mobile equipment as well as the deadline of offloading tasks are studied by adopting deep reinforcement learning (DRL) and convex optimization[15-16]. Considering the time cost, energy cost of IoT devices and resource utilization of cloudlets in smart city, the computation offload problem is researched, and a balanced offload strategy is perceived using Pareto optimization[17]. By using game theory, the comprehensive profit of mobile devices and edge service providers are designed as optimization problems, coming up with an offloading solution enhancing both data privacy and system power efficiency [18].

When blockchain is enabled in smart city, the resource management problem is necessarily considered with more variables such as task profit obtained from the blockchain system, service latency and resource occupation. To optimize these multiple optimization objectives with complex restrictions, DRL is efficient by the perception ability of deep learning and the decision-making ability of reinforcement learning[19-20]. In blockchain enabled MEC scenario, task scheduling and offloading problems are studied and modified as Markov decision process (MDP). Policy gradient method is adopted to solve the optimization goal considering the long-term mining reward, system resource and latency cost while deep Q-learning (DQL) optimizes the task execution delay and energy consumption[21-22]. Another DRL method called asynchronous advantage actor-critic (A3C) is applied in the cooperative computation offloading and resource allocation scenario, enhancing the computation rate and throughput as main factors of optimization function[23]. In the intelligent resource allocation mechanism for video services, the system performance is optimized by A3C in terms of throughput and latency[24]. Aiming at the system energy efficiency in Internet of Vehicles (IoV) scenario, A3C is also adopted to solve the optimization function composed of energy and the computation overheads[25].

Specifically, in a related scenario proposed in Ref. [26], where MEC provides users with sufficient resources to reduce the computing pressure in the IoT system supported by blockchain, the node matching between users and edge computing servers in consideration of wireless channel quality and QoS is studied, the simulation results show excellent performance with the help of reinforcement with baseline algorithm. However, different from the common IoT, the smart city has diverse requirements in application layer, and the demand of high restrictions on latency in applications such as high definition (HD) mapping and intelligent driving decision in smart traffic service is one of the most important factors[27]. In this perspective, the application implied in the smart city dabbles in a wider range and ensures more latency restriction in comparison with common IoT services, which has not been fully studied.

In order to realize the data security and ensure the quality of service (QoS) requirements of smart city applications, both the blockchain and MEC technologies are adopted to enhance the system. The secure process and storage of original data from the smart city applications with latency constraint is considered as the task. The task offloading problem maximizing the long-term reward which consists of the profits of task processing and the cost of task processing latency is the main concern. The task offloading problem is formulated as a MDP, and the DRL method which uses a deep neural network as the decision agent is used to optimize the long-term reward. The episodic simulation is built, and the policy gradient (PG) with baseline algorithm is adopted to train the decision agent. The performance with different environment parameters are tested, which respectively confirm the effectiveness and the generalization ability of the policy gradient based task offloading (PG-TO) algorithm.

The rest of this article is organized as follows. System model is formed in Section 1. The task offloading problem is formulated in Section 2 with the definition of the actions, states and rewards of the MDP. In Section 3, DRL is used to solve the task offloading problem. The training and testing results are given in Section 4. At last, this paper is concluded in Section 5.

1 System model

In this paper, the MEC-enhanced smart city scenario is considered, and the blockchain technology is adopted to ensure the secure process and storage of original data from the smart city applications. As shown in Fig. 1, the system is composed of physical layer and blockchain layer, which is described in detail in this section. Besides, the reputation model and the profit model are also given in the following.

1.1 Physical layer

The physical layer of the proposed MEC-enhanced smart city with blockchain management functions is shown in Fig. 1. There are 3 classes of participants in

the proposed physical layer, which are user equipment (UE), access point (AP), and main access point (MAP).

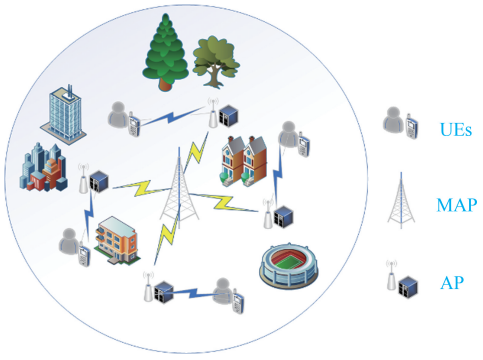


Fig. 1 Physical layer

(1) UE can be IoT device or mobile device which has limited frequency and computing resources. It is the user of various emerging applications in smart city, and the key information of its applications or behaviors are considered as the computation task which needs to be uploaded to the blockchain for secure process and storage. It is capable of operating the preset program according to the smart contract in blockchain but does not participate in the consensus process.

(2) AP is generally the small base station deployed with MEC server. It performs as the blockchain node which takes part in the consensus process in blockchain and is responsible for uploading the offloaded tasks from UEs to the blockchain.

(3) MAP is the central control node which is in charge of assigning the computation tasks form UEs to appropriate APs. Besides, it has the similar property as AP that it can participate in the consensus process and maintain the blockchain system.

1.2 Blockchain layer

As the rules of blockchain nodes running in the smart city, the smart contract defines the detailed responsibility of the nodes. Based on the smart contract, the specific steps of the task offloading process are shown below.

(1) UE adopts various applications in smart city generating computation task and declares the average task fee it can pay for the task.

(2) UE generates task offloading request including information of the average task fee and the latency limitation, and sends the requests to the MAP.

(3) MAP observes the frequency and computing resource occupation of all the APs, the reputation value of the APs and the arrived task offloading requests of the UEs.

(4) MAP assigns the tasks to APs according to the current system observation in step(3).

(5) APs allocate frequency and computing resources to transmit and compute the offloaded tasks.

(6) If the task is offloaded and computed successfully within the tolerable latency, the task is accomplished and task fee is paid to the APs as task profit. Otherwise, AP does not get paid.

(7) MAP updates the reputation value of APs according to the accomplish status of tasks.

1.3 Reputation model

In order to value the reputation and profit for each APs, the corresponding value model is proposed in the following subsections. As the historical reputation of each blockchain node reflects its competitiveness in the consensus process, the credit model formulated in a MEC enabled blockchain system[26] is used in this paper. The reputation is denoted by r_i, which shows the reliability of the i-th AP and is closely related to the probability of accomplishing a task within the tolerable latency. The value of r_i is linear within a restricted range $r_i \in [1,5]$, whose initial value is set as 3. Every time if the task is accomplished by the i-th AP, $(r_i + 0.1)$, otherwise, $(r_i - 0.5)$.

1.4 Profit layer

The profit of APs is only contributed by the task fee paid from the offloaded tasks of UEs. In order to establish an incentive mechanism, it is supposed that APs that have high reputation value will be paid with more profit, which is a reward and approval for the high quality of service. When the i-th AP accomplishes the j-th offload service, the practical profit can be described as

$$\hat{f}_j = f_j\left(\frac{r_i - \bar{r}}{\tilde{r}} + 1\right) \tag{1}$$

where f_j stands for the expected task profit that is defined once the j-th task is generated, and $\hat{f}_j$ represents the actual profit value that the i-th AP gains when the j-th task is done. For easy understanding, use '^' in this article as the practical value of the original estimated value, '~' as the maximum value and '-' as the average value. So, $\bar{r}$ and $\tilde{r}$ respectively represents the average and the maximum reputation value, where $\bar{r} = 3$, $\tilde{r} = 5$. Then, the overall profit that the AP could gain can be described as $\sum_{j \in J^d} \hat{f}_j$, where J^d denotes the set of tasks that are successfully completed by APs.

2 Problem formulation

In the blockchain enabled smart city scenario, the frequency and computing resources of the APs are lim-

ited. As the task offloading is a key process which can affect not only the profit of the APs but also the overall system performance, it is formulated as an optimization problem. The target is set to be maximizing the long-term cumulative profit gains from offloaded tasks for APs while considering the limitation of the resources of APs and the service latency constraint for QoS. The optimization problem is formulated as

$$\begin{aligned} &\max_{A} \sum_{j \in J^{\mathrm{d}}} \hat{f}_j(A) \\ &\text{s. t.} \quad \hat{t}_j \leqslant t_j + \tilde{T} \\ &\qquad d_t(A) \leqslant D \\ &\qquad f_t(A) \leqslant F \end{aligned} \tag{2}$$

where $\sum_{j \in J^{\mathrm{d}}} \hat{f}_j(A)$ is the key optimization goal. The optimization variable is the offloading APs index matrix which is defined as $A = (a_1, a_2, \cdots, a_t, \cdots)$, where $a_t = (a_{t,1}, a_{t,2}, \cdots, a_{t,x}, \cdots)$ stands for the offloading action at time step t, and $a_{t,x}$ means the x-th offloading action at time step t. t_j is the expected process latency of the task j and $\tilde{T}$ is the maximum service latency threshold of each task. $d_t(A)$ and $f_t(A)$ represent the computing resource and frequency resource occupation in AP by the offloaded tasks, respectively. D and F respectively denote the maximum resource storage of computing and frequency in each APs.

According to the system model described in Section 1, as MAP only observes the situation of current time step, the task offloading process has the Markov property, which is modeled as a Markov decision process (MDP). The MDP is denoted as (S, A, P, R), where S stands for state space, and A represents the action space. Specifically, P is defined as $P(s' \mid s, a)$ to describe the probability of the system state transition from $s \in S$ to $s' \in S$ after taking action $a \in A$, while R is defined as $R(a, s)$ which is the reward from taking action a when state is s. The detailed definition of state, action and reward is shown below.

2.1 State

The design of state should consider those elements including S_A, S_R, S_O and S_B. S_A represents the state of the APs, which includes the frequency and computing resource occupation situation of the current time and the next $(T-1)$ time steps. S_R denotes the state of the processing tasks in the system, including the index and reputation value of the AP where the tasks are offloaded, and the practical profit gain for AP when tasks are accomplished. S_O and S_B give the state of tasks which have arrived and are waiting to be offloaded. S_O denotes the tasks which can be observed in detail, including information of the required frequency resource in each APs if offloaded to, the related transmission latency and the expected profit. And S_B denotes the extra unobservable tasks that have arrived at the system, including the arrival time, expected profit and the expected service latency of these tasks. The state structure detail is shown in Fig. 2.

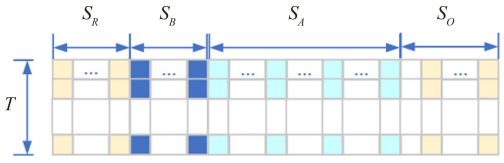


Fig. 2 The state observation of each time step t

2.2 Action

In the researched scenario, as MAP is capable of observing state information of APs, the definition of action is to select the suitable APs for UEs to offload their tasks. In time step t, if the await offloaded tasks number is D_t and the APs amount is N^{A}, the action space size is $(N^{\mathrm{A}}+1)D_t$, for the reason that the task could be either offloaded to any of the APs or not. Therefore, the action space is so huge that the efficiency of the proposed algorithm would decrease. As a result, simplify the action space by observing the first K number of await tasks at each time step and offload them adopting first-in-first-out (FIFO) method.

As the intellectual property of task offloading, the optimization of long-term accumulation value by serial decision could be realized the same as the parallel decision. Hence, it is considered that the parallel action decisions are decomposed into serial decisions in the same time step. The decision times in time step t is denoted as X_t and all the action in the time step t is denoted as

$$a_t = (a_{t,1}, a_{t,2}, \cdots, a_{t,x}, \cdots, a_{t,X_t}) \tag{3}$$

In this method could the action space be reduced, whose space size is $N^{\mathrm{A}}+1$, denoting as $\{1, 2, \cdots, N^{\mathrm{A}}, \varnothing\}$, where $a_{t,x} = i$ means that the first await task will be processed by the i-th AP in the x-th step at time t. Specifically, if the first task is the j-th task in the system task process list, $a_{t,x} = i$ stands for $a_{t,ji} = 1$.

2.3 Reward

As the optimization problem is defined in Eq. (3), the computing and frequency resources are easily implemented that MAP will not offload tasks to the APs whose frequency and computing resources are insufficient. As for the service experience, the service latency restriction of tasks in Eq. (3) can be defined as a pen-

alty function in the reward to guarantee the accomplish rate of the scheme. Therefore, the reward can be defined based on Eq. (3) as

$$\max_{A} g(A) = \alpha \sum_{j \in J^{\mathrm{d}}} \hat{f}_j(A) - \beta \sum_{j \in J^{\mathrm{a}}} \left(\frac{\hat{t}_j^{\mathrm{f}} + t_j^{\mathrm{d}}}{t_j^{\mathrm{f}} + t_j^{\mathrm{d}}} \right) \tag{4}$$

where α and β are weight integers of task profit and latency cost, respectively. J^{a} is the set of tasks that are scheduled by APs. The practical transmission latency of the j-th tasks that is allocated is denoted as $\hat{t}_j^{\mathrm{f}}$, and t_j^{f}, t_j^{d} respectively stands for the expected transmission and computation latency. As for the MDP, the objective function can be decomposed into rewards k_t in each time step.

$$k_t(a_t) = \alpha \sum_{j \in J^{\mathrm{d}}(t)} \left[\frac{\hat{f}_j(a_t) - 1}{\hat{t}_{j,d} + t_{j,f}} \right] - \beta \sum_{j \in J^{\mathrm{a}}(t)} \left[\frac{1}{t_{j,d} + t_{j,f}} \right] \tag{5}$$

where $J_1(t)$ denotes the tasks that have been scheduled and will be completed successfully at t, and $J_2(t)$ denotes all the tasks that have been scheduled at t. Hence, the optimization function could also be expressed as a cumulative function of each time step starting from initial state s_0 described as

$$g(A) = \sum_{t=0}^{T^{\mathrm{M}}} \delta^t k_t(a_t) \tag{6}$$

where the discount factor is $\delta = 1$, and T^{M} denotes the maximum time step number of one episodic simulation.

3 Policy gradient based task offloading algorithm

This section introduces the PG with baseline which is one of the DRL algorithms, to develop the policy gradient-based task offloading algorithm (PG-TO) for the proposed optimization problem. The PG-TO algorithm is illustrated in detail including the adoption of the PG with baseline method and the episodic simulation process in the following.

3.1 The adoption of PG with baseline

The adoption of PG with baseline algorithm in the proposed optimization problem and the data flow of state, action and reward in the training process are shown in Fig. 3. The PG agent, which is the policy network π_θ, outputs the probability distribution of action according to the input state, and then randomly chooses the action based on the probability distribution. The network parameter θ in π_θ is continuously optimized after each training iteration. In this way, the probability distribution of actions under each state can be established and approach the optimal strategy of probability distribution of action.

Through the training, Q samples will be generated as worksheets of the offloaded tasks, which will be respectively episodically simulated E times at each iterative training. Hence, a set of episodic simulation trajectories denoted as $\{s_{i,t,x}, a_{i,t,x}, r_{i,t,x}\}$ can be generated in each iteration, where $i \in [1, I]$, $I = E \times Q$, $t \in [0, T^{\mathrm{M}}]$, $x \in [1, X_t]$ and X_t is the total number of decision step in one time step t. When $x = X_t$, $r_{i,t,x}$ can be calculated by Eq. (5), otherwise $r_{i,t,x} = 0$. The I trajectories obtained by the E times of episodic simulation for each sample are regarded as training samples. Considering a series of decisioning process in each time step, introduce a new decisioning step index $l \in L_M$ and $L_M = \sum_{t=0}^{T^{\mathrm{M}}} X_t$, which turns $\{s_{i,t,x}, a_{i,t,x}, r_{i,t,x}\}$ into $\{s_{i,l}, a_{i,l}, r_{i,l}\}$. The l-th decision value of the i-th trajectory is $v_{i,l} = \sum_{t=1}^{L_M} \gamma^{t-l} r_{i,l}$, where γ is the discount factor and $\gamma = 1$. According to the PG with baseline, the update equation of the neural network parameter θ after each training iteration is described as follows

$$\theta = \sum_{l=0}^{L_M} \sum_{i=1}^{I} \nabla_\theta \ln \pi_\theta(s_{i,l}, a_{i,l}) (v_{i,l} - b_{\lceil i/E \rceil, l}) \tag{7}$$

where the baseline is denoted as $b_{\lceil i/E \rceil, l}$.

$$b_{\lceil i/E \rceil, l} = \frac{b_{q,l}}{E} \sum_{i=E(q-1)+1}^{E_q} v_{i,l} \tag{8}$$

As the average profit of each time step for the q-th sample, the baseline is calculated for each E times of episodic simulation in one iteration. The variance is reduced and the efficiency of policy training is enhanced by the deduction of baseline from $v_{i,l}$.

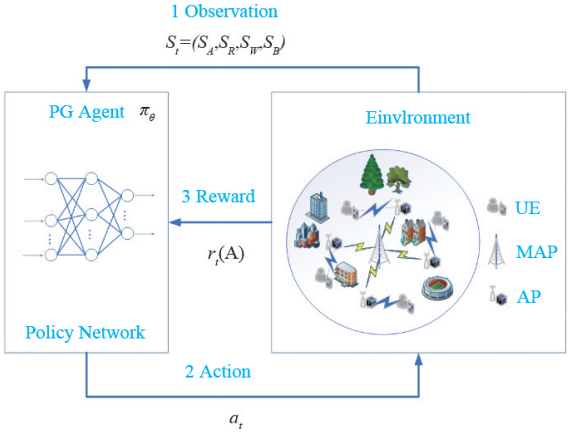


Fig. 3 The data flow in the training of the proposed PG-TO algorithm

3.2 Simulation design

The overall episodic simulation design of the proposed PG-TO algorithm is shown as Algorithm 1. The total iteration number is set as P and the policy network

π_θ is trained based on the PG with baseline.

Algorithm 1 PG-TO

Establish UEs, APs and MAP in the simulation environment, initialize the related system parameters.
Generate Q worksheets.
Randomly initialize the neural network parameters θ by normal distribution.
for iteration $p < P$ **do**
  for episodic simulation $i < I$ **do**
    get the initial state $s_{i,l} = s_{i,0}$
    while time step $t < T^{\mathrm{M}}$ **do**
      based on π_θ and $s_{i,l}$ get $a_{i,l}$
      if action $a_{i,j} = \varnothing$ **then**
        calculate k_t according to Eq. (5)
        update $t = t + 1$
      else
        Offload the j-th task to the $a_{i,l}$-th APs
        $k_t = 0$
        acquire $\hat{f}_j$, $\hat{t}_{f,j}$
      end if
      store $s_{i,j}, a_{i,j}, r_{i,j}$ in the trajectory
      update $J_1(t)$, $J_2(t)$
      update state $s_{i,l}$
    end while
  end for
  Calculate the baseline $b_{\lceil i/E \rceil, l}$ by Eq. (8)
  Update parameters of network by Eq. (7)
end for

In the beginning of each time of episodic simulation, the policy network π_θ takes observation of current simulation environment getting $s_{i,l}$ and output the probability distribution of action. Then, action $a_{i,l}$ is selected randomly according to the output probability distribution of action. If $a_{i,l} = \varnothing$, there is no task offloaded to any APs at current time step, is obtained according to Eq. (5) and then the system time step moves on. If the related task is offloaded to the $a_{i,l}$-th AP for processing, while $\hat{f}_j$ and $\hat{t}_{\mathrm{f},j}$ are counted, and $r_{i,l} = 0$. Then, $\{s_{i,l}, a_{i,l}, r_{i,l}\}$ is recorded as one sample in the trajectory. Afterwards, the related task setsare updated and the next state $s_{i,l+1}$ is obtained. The above steps are repeated to obtain a complete trajectory sample of a whole episodic simulation as long as $t < T^{\mathrm{M}}$.

4 Simulation and performance evaluation

4.1 Training performance

The training parameters are defined in Table1. Refer to the scenario setting in Ref. [26], which adopts NB-IoT service model specified in 3GPP 36.752 [28]. The bandwidth is set as 180 kHz, and the number of frequency resource units is 48 as 3.75 kHz single-tone is adopted. The size of task and the corresponding profit for APs are classified into small and big referring to Ref. [26]. The transmission frequency resource requirements of tasks are also defined according to the subchannel classifications in NB-IoT. The observable future time T is set as 300 times steps as the system performance indicators are more reasonable when the system reaches a steady state and to avoid excessive redundant simulation experiments. The number of APs N^{A} in the simulation makes trade-off between system state complexity and reasonable service pressure. The numerical setting of N^{T} makes a reasonable size of input state space while the number of N^{W} comprehensively considers the APs' numbers, overall system tasks quantity and job density.

Table1 Training parameters

Symbol	Parameter	Setting
D	Maximum frequency resource units number	48
F	Maximum computing resource units number	64
(D_t, F_t)	Frequency resource requirement and transmitting time of j-th data package in each time step (big)	{(4,8), (1,8), (48,1)}
	Frequency resource requirement and transmitting time of j-th data package in each time step (small)	{(24,2), (12,4), (4,2)}
T	Time steps that can be observed to the future	300
f_j	Expected task profit of the j-th task (big)	[10, 15]
	Expected task profit of the j-th task (small)	[15, 20]
$\tilde{T}$	Maximum latency of tasks	25
N^{A}	Number of APs	5
N^{T}	Maximum number of observable processing tasks of each AP	25
N^{W}	Maximum storage number for await allocate tasks arrived of each AP	125
α	Weight for task profit	1
β	Latency cost weight for each task	5
λ	Tasks arrival rate	0.8
R	Proportion of small tasks in all tasks	0.8

The policy network used by PG-TO method has 2 fully hidden layers, each of which has 32 neurons. Moreover, max-SINR, max-credit, max-resource and ran-

dom are set as the comparison methods. Max-SINR always selects the AP which has the maximum SINR towards UEs. The max-credit prefers the AP with the maximum reputation value r_i while the max-resource tends to offload tasks to APs with the maximum available frequency and computing resources.

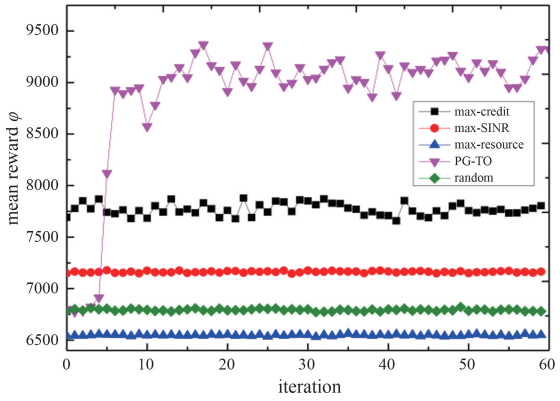


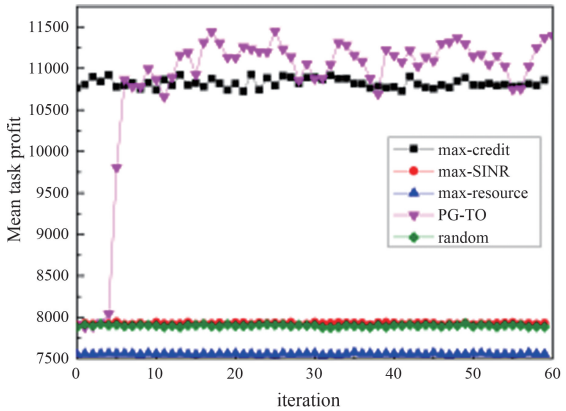
Fig. 4 Mean reward φ

Fig. 4 shows the average value of mean reward φ which is the average $g(a)$ of I samples in each iteration. For training efficiency, the learning rate is set as 0.001. And then, it is decreased to 0.0005 after 20-th iteration for better convergence performance. The φ of PG-TO rises rapidly and surpasses the value of the max-credit after only 5 iterations, and finally converges to about 9100. The φ of PG-TO is about 16.6% better than the second-best max-credit strategy, while max-SINR strategy is in the third place, about 7180. The random strategy and the max-resource strategy perform worst, both of which range from 6500 to 6800.

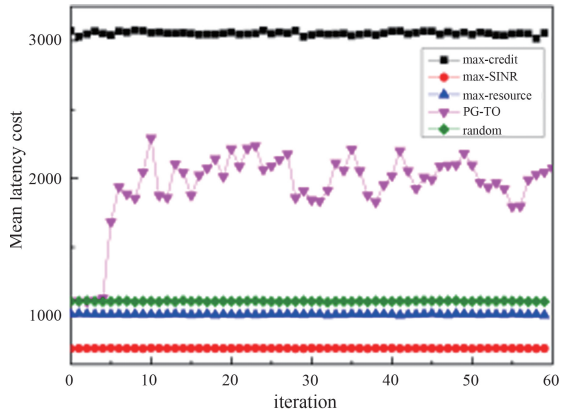
As two important components in φ, the average task profit and the average latency cost of APs are shown in Fig. 5, and their calculation formula is shown in Table 2. In Fig. 5(a), the max-credit strategy obtains the largest income among all the comparison algorithms as it always picks the AP with the highest reputation and earns higher task profit for each completed task according to Eq. (1). Meanwhile, max-SINR obtains the lowest latency cost due to the good quality of wireless transmission referring to Fig. 5(b). In Fig. 5 and Fig. 4, it can be found that there is a contradiction between the profit of APs and the latency performance of UEs. However, the advantage in task profit surpasses the shortage of latency cost, so the max-credit reaches the second-best strategy. Specifically for PG-TO, the optimal performance in φ is contributed by a little bit better task profit and about 40% less latency cost compared with max-credit.

Table 2 Calculation formula of other indicators

Indicators	Calculation formula	Figure
Task profit	$\alpha \sum_{j \in J^{d}(t)} \left[\frac{\hat{f}_j(a_t)}{\hat{t}_{f,j} + t_{d,j}} \right]$	Fig. 5(a)
Latency cost	$\beta \sum_{j \in J^{a}(t)} \left[\frac{1}{t_{f,j} + t_{d,j}} \right]$	Fig. 5(b)
Overtime rate	$\sum_{j \in J^{a}(t)} (\hat{t}_{f,j} + t_{d,j}) / (t_{f,j} + t_{d,j})$	Fig. 6(c)
Overfee rate	$\sum_{j \in J^{d}(t)} \hat{f}_j / f_j$	Fig. 7



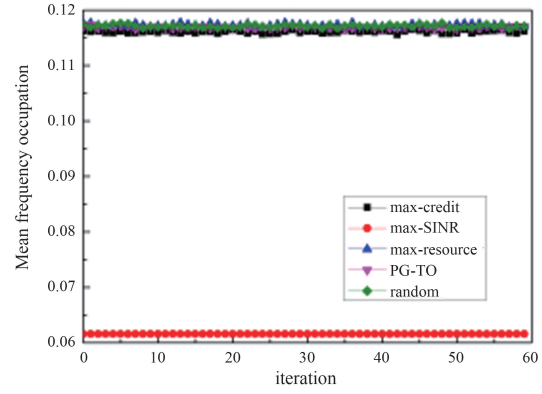
(a) Mean task profit



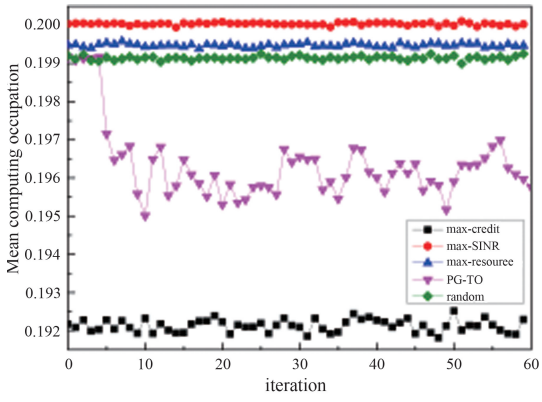
(b) Mean latency cost

Fig. 5 Mean task profit and latency cost

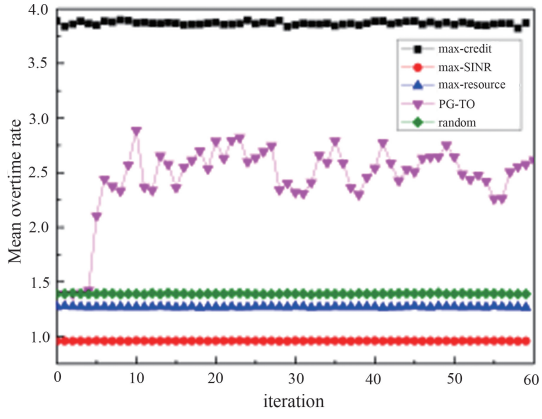
Moreover, the indicators which closely related to the system performance are also calculated and analyzed in Fig. 6, and the related calculation formula of Fig. 6(c) is shown in Table. 2. As shown in Fig. 6(a), the frequency resource occupation of PG-TO is on the same level as the comparison strategies except for max-SINR. This is because the max-SINR strategy causes load imbalance among APs, and thus restricts the utilization rate of frequency resources.



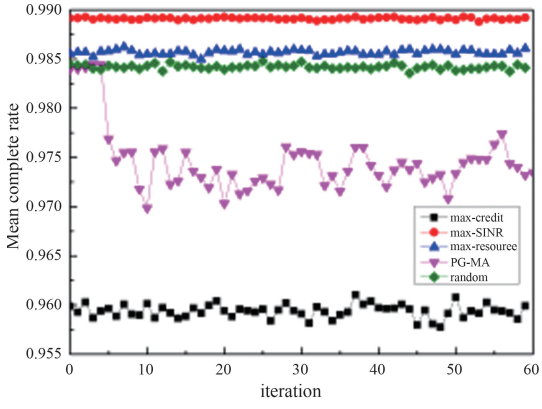
(a) Mean frequency occupation



(b) Mean computing occupation



(c) Mean overtime rate



(d) Mean complete rate

Fig. 6 System performance

As shown in Fig. 6(b) and Fig. 6(d), the performances of computing resource occupation and the complete rate are consistent. Specifically, in Fig. 6(d), the mean complete rate is calculated in the simulation program, which is the ratio of the number of finished tasks to the total number of tasks in the system. It can be observed that the max-credit strategy completes the fewest tasks while the other comparison algorithms complete more amount of tasks and consumes more amount of computing resources.

Further, the overtime rate is defined as $\sum_{j\in J^{a}(t)}(\hat{t}_j + t_j^{d})/(t_j^{f} + t_j^{d})$, which is the practical service latency and the expected service latency ratio. According to the performance of overtime rate shown in Fig. 6(c), it can be known that PG-TO smartly offloads a moderate number of tasks to trade off the acquired task profit and latency cost.

Besides, the ratio of practical task profit to expected task profit is further counted as the overfee rate. Accordingly, the overfee rate range of each strategy is shown in Fig. 7, which marks the corresponding maximum and minimum overfee rate of each iteration and the related calculation formula is shown in Table. 2. Generally, the mean overfee rate of PG-TO algorithm fluctuates in the range of 1.16 to 1.27, and its mean overfee rate is 14.2% higher than the second-best max-credit strategy, lower than any other comparison strategies. Moreover, it is easily spotted that the variance overfee rate of max-credit and that of PG-TO are at a significantly high level compared with other strategies. The reason is that max-credit strategy always selects the AP with the maximum reputation value, which causes tasks backlogs in the high credit value APs whose reputation declines due to the timeout of tasks. However, as shown in Fig. 5(a) and Fig 6(d), max-credit strategy has greater advantages than other comparison strategies with the lowest complete rate. Hence,

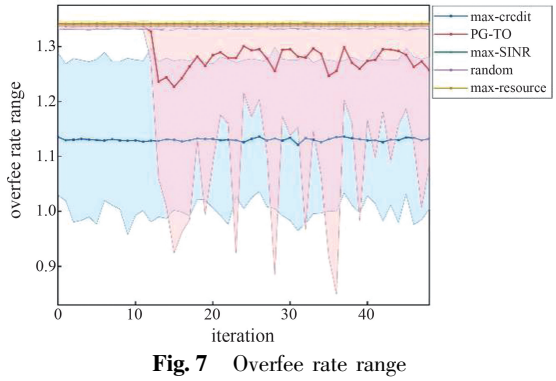


Fig. 7 Overfee rate range

it could be indicated that tasks with relatively high initial task fee are opportunistically offloaded to the higher reputation APs while tasks with lower initial task fee are selectively abandoned. Similarly, the intelligent task offloading scheme learned by PG-TO is shown by the training results that it can gradually learn the advantages of the max-credit and alleviate the huge cost of latency as shown in Fig. 5(b).

4.2 Test performance

For the policy network trained under the situation that the proportion of small tasks is 0.8, the performance with proportion of small tasks chaning is tested. In the test of each small tasks proportion, 100 samples different from the training samples are generated for simulation.

Fig. 8 shows the test results in the proportion of small tasks from 72% to 88% with 2% interval. As the small tasks proportion increases, the total task profit decreases, then φ values of random strategy, max-resource strategy, max-SINR strategy and PG-TO all show a downward trend as a whole accordingly except for the max-credit strategy.

For the max-credit strategy, φ is positively related to the increase proportion of the small tasks. That is because the overtime problem caused by the unbalanced offloading scheme can be slightly alleviated when the overall load of the system reduces, and then the advantage of choosing the maximum reputation to get the maximum practical task profit becomes obvious. Additionally, PG-TO obtains the highest φ when the small tasks proportion ranges from 0.72 to 0.86, which indicates a good generalization.

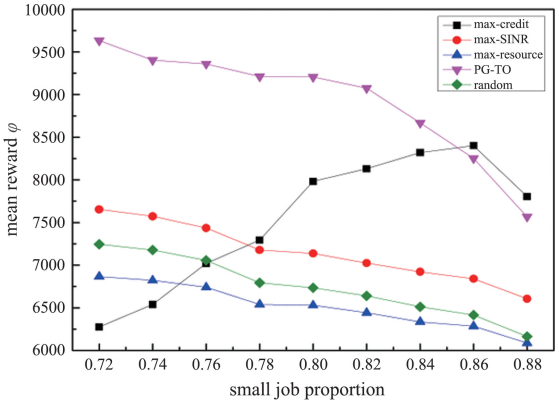


Fig. 8 Test result of mean reward φ

In all, the proposed PG-TO algorithm has a significant advantage in the optimization goal over the other 4 comparison algorithms. The PG-TO algorithm is capable of providing APs with more task profit while maintaining an acceptable latency cost. The resource occupation of PG-TO is at an average value, and the overtime rate along with complete rate are well balanced. Moreover, PG-TO shows a good generalization facing with different level of system tasks load. This means that the proposed PG-TO algorithm is capable of intelligently selecting proper APs to have the tasks offloaded in comprehensive consideration of APs' state, wireless environment, and tasks set while ensuring the latency requirements.

5 Conclusion

This article researches the task offloading process in a MEC-enhanced smart city with blockchain management functions. The task offloading process is modeled as a MDP and an optimization problem is developed focusing on the profit gain for APs and QoS requirement of UEs. The proposed optimization problem is solved by the PG method using the reinforce with baseline algorithm, and its training performance is 16.7% better than the second-best comparison strategy. The test performance with various small task proportion indicates that the proposed PG-TO algorithm has a good generalization.

References

[1] TALEBKHAH M, SALI A, MARJANI M, et al. IoT and big data applications in smart cities: recent advances, challenges, and critical issues[J]. IEEE Access, 2021, 9(55): 465-484.

[2] SILVA B N, KHAN M, HAN K. Towards sustainable smart cities: a review of trends, architectures, components, and open challenges in smart cities[J]. Sustainable Cities and Society, 2018, 38(6): 697-713.

[3] PRADEEP PAI T, SHASHIKALA K L. Smart city services-challenges and approach[C]//2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). Faridabad: IEEE, 2019: 553-558.

[4] ZAFEIRIOU I. IoT and mobility in smart cities[C]//The 3rd World Symposium on Communication Engineering. Virtual: WSCE, 2020: 91-95.

[5] MISHRA P, THAKUR P, SINGH G. Enabling technologies for IoT based smart city[C]//2021 6th International Conference on Image Information Processing (ICIIP). Near Shimla: IEEE, 2021: 99-104.

[6] RIVERA R, ROBLEDO J G, LARIOS V M, et al. How digital identity on blockchain can contribute in a smart city environment[C]//2017 International Smart Cities Conference (ISC2). Wuxi: IEEE, 2017: 1-4.

[7] MAJDOUBI D E, BAKKALI H E, SADKI S. Towards smart blockchain-based system for privacy and security in a smart city environment[C]//2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech). Marrakesh: IEEE,

2020:1-7.
[8] QIAN Y, LIU Z, YANG J, et al. A method of exchanging data in smart city by blockchain[C]//2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/Smart City/DSS). Exeter: IEEE, 2018:1344-1349.
[9] MORA O B, RIVERA R, LARIOS V M, et al. A use case in cyber security based in blockchain to deal with the security and privacy of citizens and smart cities cyber infrastructures[C]//2018 IEEE International Smart Cities Conference (ISC2). Kansas City: IEEE, 2018:1-4.
[10] SPINELLI F, MANCUSO V. Toward enabled industrial verticals in 5G: a survey on MEC-based approaches to provisioning and flexibility[J]. IEEE Communications Surveys Tutorials, 2021, 23(1):596-630.
[11] KHAN L U, YAQOOB I, TRAN N H, et al. Edge-computing-enabled smart cities: a comprehensive survey[J]. IEEE Internet of Things Journal, 2020, 7(10): 200-232.
[12] JAMIL S U, KHAN A M, REHMAN S U. Intelligent task off-loading and resource allocation for 6G smart city environment[C]//2020 IEEE 45th Conference on Local Computer Networks (LCN). Sydney: IEEE, 2020:441-444.
[13] FENG J, YU F R, PEI Q, et al. Joint optimization of radio and computational resources allocation in blockchain-enabled mobile edge computing systems[J]. IEEE Transactions on Wireless Communication, 2020, 19(6):4321-4334.
[14] YANG T, CHAI R, ZHANG L. Latency optimization-based joint task offloading and scheduling for multi-user MEC system[C]//2020 29th Wireless and Optical Communications Conference (WOCC). Newark: IEEE, 2020: 1-6.
[15] ALE L, ZHANG N, FANG X, et al. Delay-aware and energy-efficient computation offloading in mobile-edge computing using deep reinforcement learning[J]. IEEE Transactions on Cognitive Communications and Networking, 2021, 7(3):881-892.
[16] MAHMOOD A, HONG Y, EHSAN M K. Optimal resource allocation and task segmentation in IOT enabled mobile edge cloud[J]. IEEE Transactions on Vehicular Technology, 2021, 70(12):13294-13303.
[17] HUANG H, PENG K, XU X. Collaborative computation offloading for smart cities in mobile edge computing[C]// 2020 IEEE 13th International Conference on Cloud Computing (CLOUD). Beijing: IEEE, 2020:176-183.
[18] PENG K, HUANG H, LIU P, et al. Joint optimization of energy conservation and privacy preservation for intelligent task offloading in MEC-enabled smart cities[J]. IEEE Transactions on Green Communications and Networking, 2022, 6(3):1671-1682.
[19] ARULKUMARAN K, DEISENROTH M P, BRUNDAGE M, et al. Deep reinforcement learning: a brief survey[J]. IEEE Signal Processing Magazine, 2017, 34(6): 26-38.
[20] LUONG N C, HOANG D T, GONG S, et al. Applications of deep reinforcement learning in communications and networking: a survey[J]. IEEE Communications Surveys Tutorials, 2019, 21(4):3133-3174.
[21] GAO Y, WU W, NAN H, et al. Deep reinforcement learning based task scheduling in mobile blockchain for IoT applications[C]//2020 IEEE International Conference on Communications (ICC). Dublin: IEEE, 2020:1-7.
[22] SAMY A, ELGENDY I A, YU H, et al. Secure task offloading in blockchain-enabled mobile edge computing with deep reinforcement learning[J]. IEEE Transactions on Network and Service Management, 2022, 19(4): 4872-4887.
[23] FENG J, YU F R, PEI Q, et al. Cooperative computation offloading and resource allocation for blockchain-enabled mobile-edge computing: a deep reinforcement learning approach[J]. IEEE Internet of Things Journal, 2020, 7(7):6214-6228.
[24] JIANG X, YU F R, SONG T, et al. Intelligent resource allocation for video analytics in blockchain-enabled Internet of autonomous vehicles with edge computing[J]. IEEE Internet of Things Journal, 2022, 9(16):14260-14272.
[25] YE X, LI M, YU F R, et al. MEC and blockchain-enabled energy-efficient Internet of vehicles based on A3C approach[C]//2021 IEEE Global Communications Conference (GLOBE-COM). Madrid: IEEE, 2021:1-6.
[26] GAO Y, WU W, DONG J, et al. Deep reinforcement learning based node pairing scheme in edge-chain for IoT applications[C]//2020 IEEE Global Communications Conference. Taibei: IEEE, 2020:1-6.
[27] DONG J, WU W, GAO Y, et al. Deep reinforcement learning based worker selection for distributed machine learning enhanced edge intelligence in Internet of vehicles [J]. Intelligent and Converged Networks, 2020, 1(3): 234-242.
[28] The 3rd Generation Partnership Project (3GPP). Narrow band Internet of things (NB-IoT); study on NB-IoT RF requirement for coexistence with CDMA (release 14) [R]. 3GPP, 2017.

JIN Kaiqi, born in 1998. He is currently pursuing the M. S. degree at Faculty of Information Technology, Beijing University of Technology. He received the B. S. degree in communication engineering from Beijing University of Technology in 2020. His research interests include network slicing and mobile edge computing.