Hindawi
Mobile Information Systems
Volume 2020, Article ID 7630275, 6 pages
https://doi.org/10.1155/2020/7630275

Research Article

Task Offloading with Power Control for Mobile Edge Computing Using Reinforcement Learning-Based Markov Decision Process

Bingxin Zhang[1,2] **Guopeng Zhang**[1] **Weice Sun**[3] **and Kun Yang**[4]

[1]*School of Information and Control Engineering, China University of Mining and Technology, Xuzhou, China*
[2]*School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China*
[3]*School of Cyber Science and Engineering, Southeast University, Nanjing, China*
[4]*School of Computer Science & Electronic Engineering, University of Essex, Colchester, UK*

Correspondence should be addressed to Guopeng Zhang; gpzhang@cumt.edu.cn

Received 8 October 2019; Revised 7 November 2019; Accepted 16 December 2019; Published 14 February 2020

Academic Editor: Wenchi Cheng

This paper proposes an efficient computation task offloading mechanism for mobile edge computing (MEC) systems. The studied MEC system consists of multiple user equipment (UEs) and multiple radio interfaces. In order to maximize the number of UEs benefitting from the MEC, the task offloading and power control strategy for a UE is optimized in a joint manner. However, the problem of finding the optimal solution is NP-hard. We then reformulate the problem as a Markov decision process (MDP) and develop a reinforcement learning- (RL-) based algorithm to solve the MDP. Simulation results show that the proposed RL-based algorithm achieves a near-optimal performance compared to the exhaustive search algorithm, and it also outperforms the received signal strength- (RSS-) based method no matter from the standpoint of the system (as it leads to a larger number of beneficial UEs) or an individual (as it generates a lower computation overhead for a UE).

1. Introduction

With the radically increasing popularity of mobile UEs, such as smart phones, tablet computers, and Internet of Things (IoT) devices, new mobile applications such as navigation, face recognition, and interactive online gaming are emerging constantly [1]. Nevertheless, the limited computing resources of UEs are incapable of meeting the demand of computation intensive applications and, hence, become the bottleneck for providing satisfactory QoS. Conventional cloud computing systems enable UEs to utilize the powerful computing capability in remote public clouds; however, long latency may be incurred due to the data exchange in wide area networks (WANs). In order to reduce the latency, MEC systems have been proposed to deploy computing resource closer to UEs within the radio access network (RAN) [2].

A typical MEC system is shown in Figure 1 [3]. The edge cloud consists of a number of base band units (BBUs) and a MEC server. Multiple remote radio units (RRUs) working as the radio transceivers of the RAN are connected to the edge cloud via the optical fiber. Each mobile clone in the MEC server is associated with a specific UE. It works as the proxy virtual machine for the UE as it can collect the task input data generated by the UE, produce the analytical results on behalf of the UE, and send the results back to the UE. By offloading computation tasks from UEs to proximate edge clouds, MEC has the potential to reduce computation latency, avoid network congestion, and prolong the battery lifetimes of UEs [4].

In a MEC system, a UE is likely to have many candidate RRHs via in which it can offload the task to the edge cloud. The problem of associating a UE with an appropriate RRH is becoming more important. A conventional approach (which is also suggested by LTE-A) is to select the RRH which offers the highest received signal strength (RSS) to offload the task, but this approach does not consider the interference caused by the UEs associated with the same RRH. There have been many efforts in the literature toward the computation task offloading problems in MEC systems. Liu et al. [5] proposed a resource allocation scheme for the multiuser task

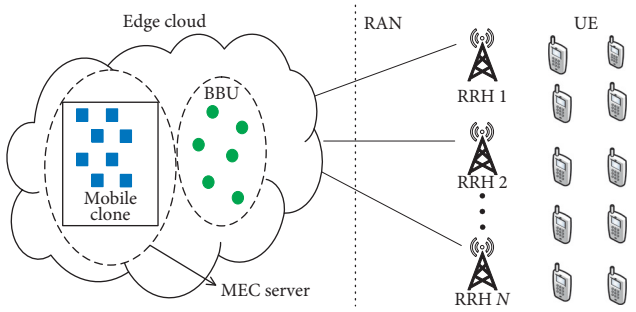

Figure 1: The typical MEC system model.

offloading scenario. The target is to minimize the overall UEs' energy consumption under the latency constraint. Since only one RRH is considered in [5], Zhang et al. [6] extended the work to the multi-RRH scenario. However, the UE-RRH association is predetermined in [6]. A more flexible association scheme is required to balance the signal-to-interference-plus-noise ratios (SINRs) at different RRHs. To address this issue, Chen et al. [7] studied the task offloading problem in a multichannel interference environment. They devised a game theoretic strategy for a UE to determine the channel via which it can offload the task. Unfortunately, optimized power control was not considered in [7]. As reported in [8–10], efficient power control can greatly alleviate the severe SINR of a shared channel, thus leads to a substantial performance improvement for all users.

Motivated by the previous works, this paper designs an efficient task offloading mechanism for the MEC systems with multi-UE and multi-RRH settings. In order to maximize the number of the UEs benefitting from the MEC, the optimal task offloading and power control strategy is found in a joint manner. Although the formulated mathematical problem is NP-hard, we can obtain the near-optimal solution by using an alternate RL-based MDP.

2. System Model

The studied MEC system working with the multi-UE and multi-RRH settings is shown in Figure 1. The set of the RRHs is denoted by $\mathcal{K} = \{1, 2, \ldots, K\}$ and the set of the UEs is denoted by $\mathcal{N} = \{1, 2, \ldots, N\}$. The UEs are distributed uniformly in the radio coverage area of the RRHs. It is assumed that the nth ($\forall n \in \mathcal{N}$) UE has a computation task $\mathcal{T}_n$ to be executed which is characterized by a two-tuple of parameters (b_n, c_n), where b_n (in bits) denotes the amount of the task input data and c_n (in CPU cycles/bit) denotes the number of CPU cycles required for computing 1 bit of the data. The values of b_n and c_n depend on the nature of the task and can be obtained through offline measurements [8].

We assume that a UE has Y power levels (corresponding to Y modulating constellations) for data transmission. Let P^1 and P^Y denote, respectively, the minimum and maximum transmit powers at a UE. Let p_n denote the transmit power applied by the nth UE for uploading the task input data to the RRH. For $\forall y \in \mathcal{Y} = \{1, \ldots, Y\}$, we have $p_n \in \{P^1, \ldots, P^y, \ldots, P^Y\}$ for $\forall n \in \mathcal{N}$. MEC enables a UE to perform task offloading by sending the task input data to the edge cloud via an RRH. Let $Z_n = (d_n, p_n)$ denote the task offloading decision for the nth UE, where $d_n \in \{0, 1, \ldots, K\}$. $d_n = 0$ means that the UE chooses to execute the task locally, and $d_n = k\,(\forall k \in \mathcal{K})$ means that the UE chooses to offload the task to the edge cloud via the kth RRH by using transmission power p_n.

2.1. Local Computing Model. If a UE chooses to execute the task locally, the latency for computing the task $\mathcal{T}_n$ can be expressed as

$$t_n^{\mathrm{L}} = \frac{b_n c_n}{f_n^{\mathrm{L}}}, \quad \forall n \in \mathcal{N}, \tag{1}$$

where f_n^{L} denotes the computation capacity of the nth UE that is measured by the number of CPU cycles per second.

Let V_n denote the energy consumption per second for computing at the nth UE. The total energy consumption for computing the task $\mathcal{T}_n$ locally is given by

$$e_n^{\mathrm{L}} = V_n \cdot t_n^{\mathrm{L}}, \quad \forall n \in \mathcal{N}. \tag{2}$$

In this paper, we consider that the UEs may have different QoS demands. That is, some delay sensitive UEs (e.g., mobile phones and surveillance UEs) need lower latency but can bear higher energy consumption, while some energy sensitive UEs (e.g., sensor nodes and IoT devices) require lower energy consumption but is delay insensitive. So, we adopt a composite index, termed as the computation cost in [5], to reflect the QoS satisfactory of a UE for executing a computation task.

In detail, the computation cost for the nth UE to execute task $\mathcal{T}_n$ locally is defined as

$$U_n^{\mathrm{L}} = u_n t_n^{\mathrm{L}} + (1 - u_n) e_n^{\mathrm{L}}, \quad \forall n \in \mathcal{N}, \tag{3}$$

where $u_n\,(0 \le u_n \le 1)$ is the weighting factor used for adjusting the tradeoff between the execution latency and the energy consumption. When a UE is at a low battery state and cares more about the energy consumption, it can set $u_n = 0$. In contrast, when a UE is with sufficient energy and runs some delay sensitive applications, it concerns more about the execution latency and can set $u_n = 1$.

2.2. Mobile Edge Computing Model. When a UE does not have enough computation or energy resource to process the computation task locally, it can offload the task to the edge cloud. In this case, a UE should select one of the RRHs and then transmit the task input data to the edge cloud via the RRH by consuming communication resource.

For easy analysis, we consider a quasi-static scenario where the set of the active UEs and their wireless channel conditions remain unchanged during a task offloading decision period T (e.g., several hundred milliseconds), while they can change across different periods. We also assume that each RRH holds just one physical channel, and the channels of the RRHs are nonoverlapped. Each UE can thus select a specific RRH to offload the computation task to the edge cloud.

Let w denote the channel bandwidth available for each RRH. Given the decision profile $\mathcal{Z} = (Z_1, \ldots, Z_N)$ of the active UEs, the transmission rate of the nth UE that selects the kth RRH to offload the task can be computed as

$$R_{n,k} = w \log_2\left(1 + \frac{p_n g_{n,k}}{\sigma^2 + \sum_{i \in \mathcal{N}-\{n\} \text{ and } d_i = d_n} p_i g_{i,k}}\right), \quad (4)$$

where σ^2 is the noise variance at the kth RRH, $g_{n,k}$ is the power gain of the channel from the nth UE to the kth RRH, and the term ($i \in \mathcal{N} - \{n\}$ and $d_i = d_n$) denotes the ith UE other than the nth UE that also selects the kth RRH to offload the task to the edge cloud.

Due to the powerful computing capability provided by the edge cloud (as many telecom operators are capable for large scale infrastructure investment), we ignore the latency and energy consumption at the edge cloud for executing the tasks offloaded by the UEs. Additionally, as the computation results are of small size, the feedback delay can also be ignored. Hence, the latency for executing the task $\mathcal{T}_n$ remotely at the edge cloud via the kth RRH can be expressed as

$$t^{\mathrm{M}}_{n,k} = \frac{b_n}{R_{n,k}}, \quad \forall n \in \mathcal{N}, \forall k \in \mathcal{K}. \quad (5)$$

The energy consumption of the UE is mainly generated by the task input data transmission which can be given as

$$e^{\mathrm{M}}_{n,k} = p_n \cdot t^{\mathrm{M}}_{n,k}, \quad \forall n \in \mathcal{N}, \forall k \in \mathcal{K}. \quad (6)$$

When the nth UE selects the kth RRH to offload the task to the edge cloud, we can define the computation cost for the nth UE in terms of the weighted sum of execution latency and energy consumption as

$$U^{\mathrm{M}}_{n,k} = \nu_n t^{\mathrm{M}}_{n,k} + (1 - \nu_n) e^{\mathrm{M}}_{n,k}, \quad \forall n \in \mathcal{N}, \forall k \in \mathcal{K}. \quad (7)$$

3. Problem Formulation

In general, the number of the UEs that attempt to access the edge cloud is much larger than the number of the RRHs (i.e., $K \ll N$). The UEs are ordered to make their task offloading decision simultaneously in each decision period T. Since the wireless channel held by each RRH is a shared medium, if too many UEs select the same RRH to offload their tasks, it would incur severe co-channel interference and high computation cost for the UEs. In such a case, it would be more beneficial for a UE to select another RRH to offload the task or execute the task locally. In addition, it is also shown in [7] that if efficient power control were applied, a UE could achieve a high data rate while at the same time expending a small amount of energy. Hence, it is necessary to coordinate the transmission power of the UEs that selects the same RRH to offload their tasks to the edge cloud.

For the nth UE, the optimal task offloading decision $Z^*_n = (d^*_n, p^*_n)$ should cause the lowest cost of executing task $\mathcal{T}_n$. Particularly, we refer to the nth UE as the MEC benefited UE, if it chooses to offload the task $\mathcal{T}_n$ to the edge cloud rather than executing the task locally. That means $U^{\mathrm{M}}_{n,k} < U^{\mathrm{L}}_n$ ($\exists k \in \mathcal{K}$) and $d_n > 0$ for the nth UE, whereas from the system designer's point of view, the optimal task offloading decision for the UEs, denoted by $\mathcal{Z}^* = (Z^*_1, \ldots, Z^*_N)$, should be able to maximize the number of the MEC benefitted UEs. It can lead to a higher utilization ratio of the MEC infrastructures and bring a higher revenue for providing the MEC service. Mathematically, we can formulate the optimal task offloading problem as

$$\mathcal{Z}^* = \arg\max_{\mathcal{Z}} \sum_{n \in \mathcal{N}} 1_{\mathcal{F}=\{\forall d_n | d_n > 0\}}(d_n), \quad (8)$$

$$\text{s.t.} \quad U^{\mathrm{M}}_{n,k} < U^{\mathrm{L}}_n, \quad \forall d_n > 0, \forall n \in \mathcal{N}, \forall k \in \mathcal{K}, \quad (9)$$

$$p_n \in \left\{P^1, \ldots, P^y, \ldots, P^Y\right\}, \quad \forall n \in \mathcal{N}, \quad (10)$$

$$d_n \in \{0, 1, \ldots, K\}, \quad \forall n \in \mathcal{N}, \quad (11)$$

where $1_{\mathcal{F}=\{\forall d_n | d_n > 0\}}(d_n)$ is an indicator function defined as

$$1_{\mathcal{F}=\{\forall d_n | d_n > 0\}}(d_n) = \begin{cases} 1, & \text{if } d_n \in \mathcal{F}, \\ 0, & \text{Otherwise.} \end{cases} \quad (12)$$

However, it can be proved that problem (8) of finding the optimal decision profile $\mathcal{Z}^*$ is NP-hard as it is an instance of the Mixed Integer Nonlinear Programming (MINLP) problem (which is known to be NP-hard [11]). The proof is omitted here due to limited space. In order to ease the heavy burden of complex computing at the MEC server, we next model the task offloading decision process as a Markov decision process (MDP). Consequently, a reinforcement learning- (RL-) based algorithm is developed to find the solution to the MDP.

4. Markov Decision Process (MDP)

In the MEC system, the number of the UEs that attempt to access the edge cloud is much larger than the number of the RRHs, and the UEs are ordered to make their task offloading decision simultaneously in each decision period. These all involve an interaction between a UE (as a decision maker) and the environment (the interference incurred by the co-channel UEs), within which the UE seeks to achieve a goal as minimizing the computation cost despite uncertainty about the environment. The UEs' actions are permitted to affect the future states of the environment (the interference levels at the RRHs), thereby affecting the options and opportunities available to the UEs at later time steps. In such a situation, where outcomes are partly random and partly under the control of a decision maker, MDPs [12] provide a mathematical framework to model and analyze the decision-making process.

More precisely, the task offloading decision process is modelled as a MDP which is substantially a discrete time stochastic control process. An agent-environment interaction of the MDP is termed as an episode, which equals to a task offloading period T. An episode is further broken into several discrete time steps. At each time step t ($t = 1, 2, \ldots$), a UE is in some state s. An episode of the MDP starts from a random initial state and ends in a terminal state. A UE acting as a decision maker must choose any action a that is available

in state s; thus, the MDP responds at the next time step by moving the UE into a new state s' and giving the UE a corresponding reward r. In the proposed MDP, future states only depend on the current state instead of the former ones; thus, the memoryless Markov property is guaranteed. The actions, states, and reward functions of the proposed MDP are formally defined as below.

States: at any time step t, if a UE offloads the computation task via the kth RRH by using transmission power p^y, we say that the UE is in state $\varphi_{k,y}$ for $\forall k \in \mathcal{K}$ and $\forall y \in \mathcal{Y}$. Otherwise, if it executes the computation task locally, we say that the UE is in state $\varphi_{0,0}$. The set of states of the MDP can thus be given by $\mathcal{S} = \{\varphi_{0,0}, \varphi_{1,1}, \ldots, \varphi_{K,Y}\}$.

Actions: at each time step t, a UE must take an action a according to the current state s for $\forall s \in \mathcal{S}$, which also implies a transition from the current state s to the next state s' ($\forall s' \in \mathcal{S}$). We define $\mathcal{A} = \{\phi_{0,0}, \phi_{1,1}, \ldots, \phi_{K,Y}\}$ as the action set of the MDP, where $a = \phi_{0,0}$ implies that a UE selects local computation, and $a = \phi_{k,y}$ ($\forall k \in \mathcal{K}$ and $\forall y \in \mathcal{Y}$) implies that the UE select the kth RRH to offload the computation task by using transmission power p^y.

Reward functions: after the agent-environment interaction in each time step t, a UE obtains a reward which represents the optimization objective. The reward function just maps a pair of state and action into stochastic rewards. Since we take the objective to minimize the *computation cost* of a UE, the reward function of the nth UE is defined as

$$r(s,a) = \begin{cases} \dfrac{\lambda_1}{U_n^{\mathrm{L}}}, & \text{if } a = \phi_{0,0}, \\ \dfrac{\lambda_2}{U_{n,k}^{\mathrm{M}}}, & \text{if } a = \phi_{k,y}, \\ \forall s \in \mathcal{S}, \\ \forall a \in \mathcal{A}, \end{cases} \tag{13}$$

where λ_1 and λ_2 are variables for normalization.

5. RL-Based Solution Method

MDPs are a wide range of optimization problems which can be solved via dynamic programming (DP) and RL methods [12]. The RL method is an area of machine learning concerned with how an agent takes actions in an environment so as to maximize the cumulative reward. The main difference between DP methods and RL methods is that the latter do not assume knowledge of an exact mathematical model of the MDP and they target large scale MDPs where DP methods become infeasible.

Next, we develop an RL-based algorithm to solve the MDP. First, we assume that the decision of the nth UE for choosing an action a in a given state s is determined by a policy:

$$\pi(s) = a, \quad \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{14}$$

Then, we need to find the state-action value function $Q(s,a)$, also called the Q-value function for the MDP [12], which represents the expected return (cumulative discounted reward) that the nth UE is to receive when taking action a in state s and behaving according to the policy $\pi(s)$ afterwards. In the RL method, the Q-function is learned by the interaction between the decision makers and their environment, thus can approximate the optimal state-action value function directly, independent of the policy being followed. Next, we define the updating rule of the Q-value function as

$$\begin{aligned} Q_{t+1}(s,a) = Q_t(s,a) + \beta\Big\{ & r_{t+1}(s,a) \\ & + \gamma \max_{a \in \mathcal{A}} Q_{t+1}(s',a) - Q_t(s,a)\Big\}, \end{aligned} \tag{15}$$

where γ is the reward decay and β is the learning rate. If the optimal $Q(s,a)$ were known, the optimal policy can be found by [12]

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q(s,a). \tag{16}$$

Finally, the number of the MEC benefitted UEs can be obtained. The overall learning procedure is summarized in Algorithm 1.

In Algorithm 1, we use the ε-greedy policy [12] for the sake of discovering an effective action. In detail, the UEs perform *exploration* with probability ε ($0 < \varepsilon < 1$) at each time step, and they *exploit* stored Q-values with probability $1 - \varepsilon$. It is noted that the algorithm should be performed at the MEC server, i.e., the control center of the system. Since the MEC server has full knowledge of the RRHs and UEs and powerful computation capacity, the solely required information is only the channel state information (CSI) of the UEs. The task input data and the CSI of UEs can be conveyed to the MEC server by the RRHs. Subsequently, the scheduling information can be fed back to the UEs also via the RRHs.

6. Simulation Results

In the simulation, we set up a MEC system, as shown in Figure 1. The coverage range of the system is 1 km and multiple UEs are scattered randomly over the region. The available channel bandwidth for each RRH is $w = 1$ MHz. The power-law path loss of the wireless channels is modelled as $g_{n,k} = l_{n,k}^{-\alpha}$, where $l_{n,k}$ is the distance between the nth UE and the kth RRH and $\alpha = 4$ is the path-loss factor. The background noise variance is set as $\sigma^2 = 10^{-14}$W. The set of transmission powers for each UE is $\{50, 100, 150, 200, 250\}$ mW. We take the face recognition applications [13] as the computation tasks. For the nth UE, we set the size of the task input data as $b_n = 2$ kB, the number of the required CPU cycles per bit $c_n = 20$ cycles, and the power for local computing $V_n = 0.1$ W. Due to the heterogeneity of the mobile UEs, we assume that the CPU computational capability f_n^{L} of the nth UE is randomly selected from the set $\{0.1, 0.15, 0.2\}$ MHz, and the QoS

Initialization:
$t \longleftarrow 0$; $s \longleftarrow 0$, $\forall s \in \mathcal{S}$; $Q(s,a) \longleftarrow 0$, $\forall s \in \mathcal{S}$ and $\forall a \in \mathcal{A}$;
while $t \leq t^{\max}$ ($t^{\max}$ is the is the maximum number of iterations)
  for each UE in $\mathcal{N}$
    if exploration
      chooses an action a arbitrarily with probability ε, $0 < \varepsilon < 1$;
    else exploitation
      chooses an action $a = \arg\max_{a \in \mathcal{A}} Q(s,a)$;
    end if
    perform a and get a reward $r(s,a)$ and a successor state s';
    update the Q-value function according to equation (15);
    $s \longleftarrow s'$;
  end for
  $t \longleftarrow t+1$;
end while

Algorithm 1: Q-learning algorithm with ε-greedy policy.

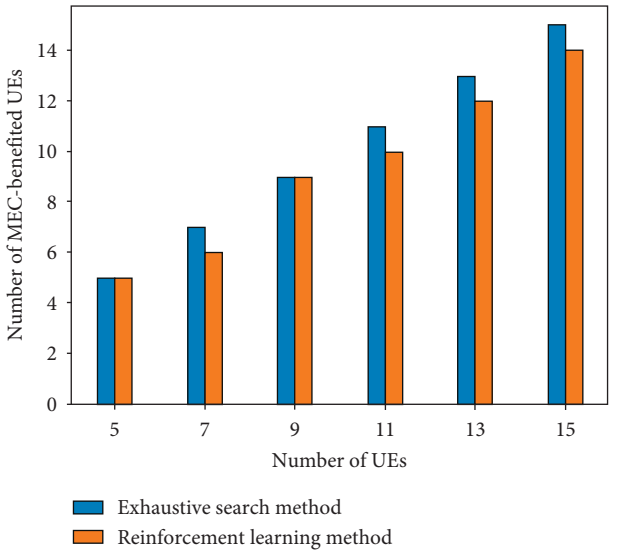

Figure 2: The effectiveness of the proposed algorithm.

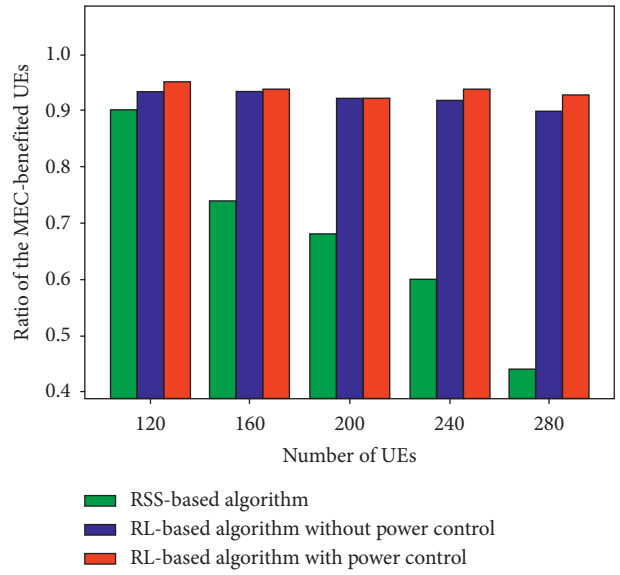

Figure 3: The number of the beneficial UEs by using different algorithms.

weighting factor u_n for the nth UE is randomly selected from the set $\{0, 0.5, 1\}$.

First, we testify the effectiveness of the proposed RL-based algorithm. The number of the RRHs in the MEC system is $K = 5$, and each UE transmits at the maximum power P^Y. We compare the number of the MEC-benefitted UEs obtained by using the RL-based algorithm to that obtained by using the exhaustive search (ES) algorithm. Note that the ES algorithm is global optimum but the computational complexity grows exponentially with the number of the UEs. The simulation is repeated 100 times and the averaged results are shown in Figure 2. We see that the RL-based algorithm can find near-optimal solutions to problem (8). Since power control is not applied in the simulation, the performance can be taken as the lower bound of the RL-based algorithm.

Next, we testify the ability of the RL-based algorithm to deal with a large scale network where 120~280 UEs can simultaneously issue their task offloading requests. For that purpose, we increase the number of the RRHs to $K = 9$. In Figure 3, we show the ratio of the beneficial UEs in the system by using different task offloading algorithms.

From Figure 3, we see that the performance of the RSS-based algorithm decreases sharply with the increasing N, while the RL-based algorithms (with and without power control) can maintain the beneficial ratio at a high level of 93%. In addition, we see that the RL-based algorithm with power control outperforms the counterpart without power control in all the network situations.

Finally, we show the effect of power control in reducing the computational cost of a UE. To this end, we compare in Figure 4 the average computation overheads obtained by a UE before and after applying the power control. The overhead of a UE is obtained by using equation (3) as it executes the task locally or by using equation (7) as it offloads the task to the edge cloud.

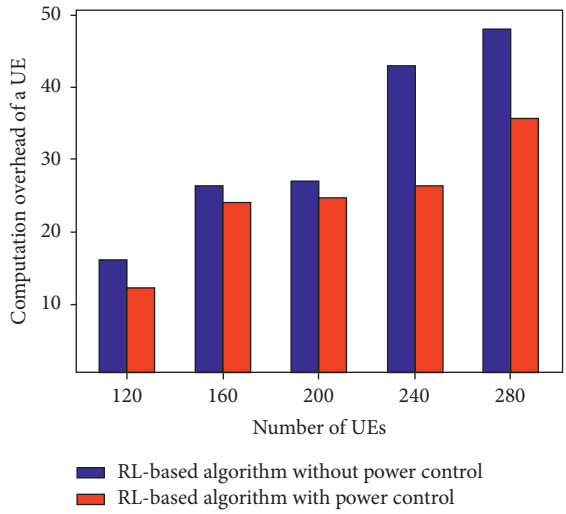

Figure 4: The average overhead of each UE by using different algorithms.

From Figure 4, we see that the RL-based algorithm with power control can bring lower computation overhead for a UE than the counterpart without power control. It implies that the RL-based algorithm with power control can well coordinate the multiuser interference and, therefore, can greatly reduce the computation overhead of a UE.

7. Conclusion

This paper proposes a RL-based MDP to solve the computation task offloading and power control problem in the MEC systems with multi-UE and multi-RRH settings. In comparison to the ES algorithm, the proposed RL-based algorithm can achieve a near-optimal system performance. While dealing with a large scale network, the proposed RL-based algorithm can achieve good performance no matter if it is from the standpoint of system or an individual.

Data Availability

The data used to support the findings of this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61971421).

References

[1] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1397–1411, 2017.

[2] W. Cheng, W. Zhang, H. Jing, S. Gao, and H. Zhang, "Orbital angular momentum for wireless communications," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 100–107, 2019.

[3] A. Alnoman, G. H. S. Carvalho, A. Anpalagan, and I. Woungang, "Energy efficiency on fully cloudified mobile networks: survey, challenges, and open issues," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 2, pp. 1271–1291, 2018.

[4] W. Cheng, X. Zhang, and H. Zhang, "Full-duplex spectrum-sensing and MAC-protocol for multichannel nontime-slotted cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 5, pp. 820–831, 2015.

[5] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proceedings of.2016 IEEE International Symposium on Information Theory*, pp. 1451–1455, Barcelona, Spain, July 2016.

[6] J. Zhang, W. Xia, F. Yan, and L. Shen, "Joint computation offloading and resource allocation optimization in heterogeneous networks with mobile edge computing," *IEEE Access*, vol. 6, pp. 19324–19337, 2018.

[7] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[8] N. Li, J.-F. Martinez-Ortega, V. H. Diaz et al., "Distributed power control for interference-aware multi-user mobile edge computing: a game theory approach," *IEEE Access*, vol. 6, pp. 36105–36114, 2018.

[9] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proceedings of 2010 USENIX Conference on Hot Topics in Cloud Computing*, Berkeley, CA, USA, October 2010.

[10] W. Cheng, H. Zhang, L. Liang, H. Jing, and Z. Li, "Orbital-angular-momentum embedded massive MIMO: achieving multiplicative spectrum-efficiency for mmWave communications," *IEEE Access*, vol. 6, pp. 2732–2745, 2018.

[11] K.-H. Loh, B. Golden, and E. Wasil, "Solving the maximum cardinality bin packing problem with a weight annealing-based algorithm," in *Operations Research and Cyber-Infrastructure*, Springer, New York, NY, USA, 2009.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, USA, 2nd edition, 2017.

[13] T. Soyata, R. Muraleedharan, C. Funai et al., "Cloud-vision: real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proceedings of 2012 IEEE Symposium on Computers and Communications (ISCC)*, pp. 59–66, Cappadocia, Turkey, July 2012.