Learning-based Task-Offloading for Mobile Edge Computing with User Mobility

Jiayi Huang, Xiajun Huang, Zipeng Lu, Jun Xu*

School of Computer and Electronic Information/School of Artificial Intelligence
Nanjing Normal University, Nanjing, Jiangsu 210023, China
jyhuang@nnu.edu.cn, xjhuang@nnu.edu.cn, zplu@nnu.edu.cn, junxu@njnu.edu.cn

***Abstract*—Mobile edge computing (MEC), envisioned as a cloud computing extension, is becoming an essential architecture for data transmission. The system performance (such as the execution delay) can be improved by offloading tasks from mobile user (MU) to edge servers. However, there is a lack of research on task offloading in the case of random movement of MUs. In this paper, we study the task offloading problem considering the MUs' mobility. We model the task offloading problem as a Multi-armed Bandit (MAB) model. We then study the task offloading algorithms based on the typical Upper-Confidencial-Bound (UCB) and ε-greedy algorithms, which can make efficient task offloading decisions by adaptively learning the MU's mobile property. By investigating the impact of the mobile property on network performance, we further propose an EPE-UCB task offloading algorithm. Simulation results demonstrate the efficiency of our algorithm in achieving low cost.**

***Index Terms*—MEC, task offloading, MAB, UCB, user mobility.**

I. Introduction

Mobile edge computing (MEC) has evolved rapidly in recent years and can provide services for computation-intensive and delay-sensitive applications. Compared to traditional cloud computing, the edge servers are closer to mobile users (MUs) which will increase data rates and reduce latency [1], [2]. By offloading tasks from the computation-constrained and energy-constrained MUs to edge servers, it can reduce the execution delay and prolong the battery life of the MUs. Therefore, designing efficient task offloading strategies is important to improve network performance [3], [4].

However, designing efficient task offloading strategy is challenging due to the dynamics of channel fadings and computing resources [5]. In some scenarios, MUs are moving which implies the uncertain locations of MUs when making task offloading decisions. This is an important factor which should be considered when designing task offloading strategies [2].

This work was supported in part by College Students' innovation and entrepreneurship training program, in part by National Natural Science Foundation of China under Grant 61901306, in part by the Research Start-up Fund of Nanjing Normal University, in part by the Dual-innovation Doctor Program of Jiangsu Province under Grant JSSCBS20220423, and in part by the Nanjing Science and Technology Innovation Project for Oversea researchers. (*corresponding author: Jun Xu)

In this paper, we study the task offloading problem with moving users based on the Multi-armed Bandit (MAB) model. It is a reinforcement learning model which deals with uncertainty of arms [6]–[9]. Some researchers adopted the MAB model to investigate the task offloading problem with uncertainties. Li *et. al.* investigated the spectrum scheduling problem in wireless networks with uncertainty of channel availability [8]. He *et. al.* proposed an efficient online service placement and resource allocation scheme based on MAB [10]. Guo *et. al.* studied the online computation offloading problem with communication and computation dynamics using MAB framework [11]. Wu *et. al.* proposed task offloading algorithms for MEC with channel uncertainty [12]–[14]. Although these researches demonstrated the efficiency of the MAB-based algorithms, the user's mobility has not been considered.

In practice, MUs often move randomly. To design a more practical task offloading algorithm, we consider the mobility of MUs. Moreover, we aim to minimize the long-term cost when designing the task offloading algorithms. The cost consists of delay and payment. The delay is a main factor considered by many researchers, and the payment is caused by offloading tasks to edge servers.

Our main contributions are summarized as follows,

- We investigate the task offloading problem considering MUs' mobility. We model the task-offloading problem as an MAB problem.
- We study the task offloading problem based on the well-known UCB algorithm and ε-greedy algorithm. Moreover, we propose a novel EPE-UCB algorithm, which can attenuate the impact of the MUs' preceding locations on the selection of the optimal edge server to offload tasks.
- We evaluate the performance of our EPE-UCB, UCB-based, ε-greedy, and random task offloading algorithms. Results show the convergence of these algorithms and demonstrate the efficiency of our algorithm in achieving low cost.

The remainder of this paper is organized as follows. Section II presents the network model and formulates the task offloading problem. Section III proposes the UCB-based, ε-

greedy, and our EPE-UCB task offloading algorithms. Section IV evaluates the performance of the task offloading algorithms. Finally, Section V concludes this paper.

II. Network Model and Problem Formulation

A. Network Model

We study an MEC system with M MUs denoted as $\mathcal{M} = \{1, 2, \ldots, M\}$ and N edge servers denoted as $\mathcal{N} = \{1, 2, \ldots, N\}$. Each MU moves randomly. An example of the network model is shown in Fig. 1, where each access point (AP) is connected to an edge server by wired link, and each MU communicates with the corresponding AP or edge server by wireless link.

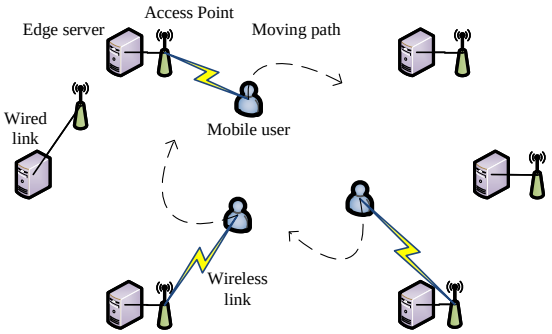


Fig. 1: An example of MEC systems with three MUs and multiple edge servers.

We consider a discrete system where time is divided into time slots $\mathbf{T} = \{0, 1, \ldots, T-1\}$. In every time slot, each MU should decide whether to offload its tasks to an edge server. If it decides to offload its tasks, it should then decide which edge server the tasks should be offloaded to.

Let a_t^i be the task offloading decision of i-th MU at time slot t. If $a_t^i = 0$, it means the task of MU i will be executed locally at time slot t. If $a_t^i > 0$ and $a_t^i \in \mathcal{N}$, it means the task will be offloaded to the a_t^i-th edge server at time slot t. For different task offloading decisions, the costs are different. The cost of a task offloading decision consists of delay and payment. We use $d_t(a_t^i)$ and $c_t(a_t^i)$ to denote the delay and the payment at time slot t for the task offloading decision a_t^i, respectively.

When $a_t^i \in \mathcal{N}$, the task of MU i will be offloaded to edge server a_t^i. The delay consists of the uplink transmission delay of the task, the computing delay on the selected edge server, and the downlink transmission delay of the result. Since the size of the result returned by the edge server is in general much smaller than the size of the task before computation, the downlink transmission delay can be ignored.

For the positions of MUs and the edge servers, we use $(x^i(t), y^i(t))$ to denote the coordinates of i-th MU, and (x_j^{es}, y_j^{es}) to denote the coordinate of the j-th edge server. Let $q_j^i(t)$ be the distance between the i-th MU and the j-th edge server, we then have that,

$$q_j^i(t) = \sqrt{\left(x^i(t) - x_j^{es}\right)^2 + \left(y^i(t) - y_j^{es}\right)^2}. \tag{1}$$

Note that, the coordinates of the edge servers are fixed, while the coordinates of MUs change from time slot to time slot. It implies that the distances from each MU to each edge server at different time slots are different. The distance between an MU and an edge server affects the channel gain, thus will affect the uplink transmission rate.

Let $G_j^i(t)$ denote the channel gain of the channel between the i-th MU and the j-th edge server. We have that,

$$G_j^i(t) = \kappa(q_j^i(t))^{-\alpha}, \tag{2}$$

where κ is a constant, and α is pathloss exponent.

During signal transmission, the Additive White Gaussian Noise (AWGN) will affect the quality of the signal. The AWGN noise power can be calculated using the following equation,

$$N = N_0 B, \tag{3}$$

where N_0 is the variance of the AWGN and B is the channel bandwidth.

Let p be the transmit power of each MU, and $SNR_j^i(t)$ be the Signal to Noise Ratio (SNR) from the i-th MU to the j-th edge server at time slot t. We then have that,

$$SNR_j^i(t) = \frac{pG_j^i(t)}{N}. \tag{4}$$

According to Shannon's formula, the uplink transmission rate $V_j^i(t)$ from the i-th MU to the j-th edge server is then given as follows,

$$V_j^i(t) = B\log_2(1 + SNR_j^i(t)). \tag{5}$$

Let L be the size of the task of MUs at each time slot, U_j be the computation frequency of edge server j, and U_0 be the computation frequency of the MUs' local processor. The edge server provides the same comptuing resources to the MUs choosing it. We then have that,

$$d_t(a_t^i) = \begin{cases} \frac{L}{U_0}, & a_t^i = 0; \\ \frac{L}{U_j} + \frac{L}{V_j^i(t)}, & a_t^i = j. \end{cases} \tag{6}$$

With the commercialization of 5G, edge servers require a certain cost to maintain and run, so users need to pay for the computing resources they take up. In mobile edge computing, the charging standards vary according to different service providers and specific services [15]. Let $c_t(a_t^i)$ be the payment of the task offloading decision a_t^i, we then have that,

$$c_t(a_t^i) = \begin{cases} 0, & a_t^i = 0; \\ Le_j, & a_t^i = j; \end{cases} \tag{7}$$

where e_j is the per bit offloading payment for edge server j.

Since the delay and payment are of different units, we normalize the delay and payment. Let φ be the normalization

rule, we use $d_t^{\varphi}(a_t^i)$ and $c_t^{\varphi}(a_t^i)$ to denote the delay and payment after normalization.

Let $C_t^i(a_t^i)$ be the total cost of the i-th MU with task offloading decision a_t^i at time slot t, we then have that,

$$C_t^i(a_t^i) = \omega d_t^{\varphi}(a_t^i) + (1-\omega)c_t^{\varphi}(a_t^i), \tag{8}$$

where $\omega \in [0,1]$ is the weight.

B. Problem Formulation

The ultimate goal of our task offloading strategies is to minimize the long-term average cost. The problem can be described as follows,

$$\min_{a_t^i \in \mathcal{N}} \lim_{T\to\infty} \frac{1}{T} \sum_{i=1}^{M} \sum_{t=0}^{T-1} C_t^i\left(a_t^i\right). \tag{9}$$

Our task offloading problem for multiple MUs can be decomposed into multiple single-MU task offloading problems. We formulate the task offloading problem for each MU as an MAB problem. Each decision is viewed as an arm. Pulling an arm implies selecting the corresponding edge server. By pulling an arm, it incurs a cost. Our goal is to minimize the long-term average cost.

III. Learning-based Task Offloading Algorithm

As for the classical MAB problem, the UCB and ε-greedy algorithms are two well-known algorithms [7]. These two algorithms can be used for our task offloading problem. We first address the key ideas of the UCB-based and ε-greedy based task offloading algorithms. We then propose our Eliminate Past Effect UCB (EPE-UCB) task offloading algorithm.

For the task offloading problem with N edge servers, there are $N+1$ actions. We use $\mathbf{A} = \{0, 1, \ldots, N\}$ to denote the actions. Action k belongs to $\mathbf{A}$. Let $N_k^i(t)$ be the number of times of MU i's action k being taken at time slot t. Let $Q_k^i(t)$ be the accumulative cost of MU i's action k at time slot t.

The average cost at time slot t of action k is given as follows,

$$Q_k^i(t) = \frac{1}{N_k^i(t)} \sum_{t=0}^{t-1} C_t^i(k). \tag{10}$$

To make the algorithm efficient, we obtain the average cost using the incremental method which is given as follows,

$$\begin{aligned} Q_k^i(t+1) =& \frac{(N_k^i(t)-1) \times Q_k^i(t) + C_{t+1}^i(k)}{N_k^i(t)} \\ =& Q_k^i(t) + \frac{1}{N_k^i(t)}(C_{t+1}^i(k) - Q_k^i(t)). \end{aligned} \tag{11}$$

For the ε-greedy algorithm, we set the exploration probability as ε. During the whole process, multiple MUs move randomly. In each time slot, a random number r is generated, and if r is less than ε, a randomized action is selected, which helps us to discover the unknown low-cost action, and thus improves our estimation accuracy. If r is greater than ε, it will choose the action with the minimum Q value.

In the UCB-based algorithm, the learning process is divided into two phases: probing phase and operational phase.

At the probing phase, each task-offloading action is tried once. Then during the operational phase, the action selected by the UCB-based task offloading algorithm is given as follows,

$$k = \arg\min_{k\in\mathbf{A}} \left\{ Q_k^i(t) + \sqrt{\frac{2\ln(N_k^i(t))}{N_k^i(t)}} \right\}, \tag{12}$$

where $Q_k^i(t) + \sqrt{\frac{2\ln(N_k^i(t))}{N_k^i(t)}}$ is upper confidence bound for action k.

The pseudocode of the ε-greedy based task offloading algorithm algorithm is given by Algorithm 1. The pseudocode of the UCB-based task offloading algorithm is given by Algorithm 2.

Algorithm 1 ϵ-greedy based task offloading algorithm

Require: Number of edge servers N, the number of MUs M; Exploring probability ε;
Ensure: Task offloading policy.
Initialize $Q_k^i(t) = 0$ and $N_k^i(t) = 0$ for all k and t;
for $i = 1, 2, \ldots, M$ **do**
  for $t = 0, 1, \ldots, T-1$ **do**
    if $r < \varepsilon$ **then**
      k is a random value in $\mathbf{A}$;
    else
      $k = \arg\min_{\varpi\in\mathbf{A}}\{Q_k^i(t)\}$;
    end if
    $Q_k^i(t) = Q_k^i(t-1) + \frac{C_t^i(k) - Q_k^i(t-1)}{N_k^i(t-1)}$;
    $N_k^i(t) = N_k^i(t-1) + 1$;
  end for
end for

Algorithm 2 UCB-based task offloading algorithm

Require: The number of edge servers N, the number of MUs M;
Ensure: Task offloading policy.
Initialize $Q_k(t) = 0$ and $N_k(t) = 0$ for all k and t;
for $i = 1, 2, \ldots, M$ **do**
  for $t = 0, \ldots, 6$ **do**
    Try a task-offloading action in each round:
    $k = t$;
    Observe $C_t^i(k)$
    Update $N_k^i(t) = 1$, $Q_k^i(t)$;
  end for
  for $t = 7, 8, \ldots, T$ **do**
    Choose the offloading action:
    $k = \arg\min_{\varpi\in\mathbf{A}} \left\{ Q_k^i(t) + \sqrt{\frac{2\ln(N_k^i(t))}{N_k^i(t)}} \right\}$;
    Update:
    $Q_k^i(t) = Q_k^i(t-1) + \frac{1}{N_k^i(t-1)}(C_t^i(k) - Q_k^i(t-1))$;
    $N_k^i(t) = N_k^i(t-1) + 1$;
  end for
end for

However, during the MUs' movement, the experience generated by the preceding position is not fully applicable to

the current decision, i.e., the preceding experience gradually loses its validity during the user's movement, but the classical UCB-based and ε-greedy based task offloading algorithms still record it in Q. The update of the experience is still little studied. Motivated by this, we propose the EPE-UCB task offloading algorithm. Its pseudocode is given in Algorithm 3.

In the EPE-UCB algorithm, we introduce a value μ. At time slot t, the EPE-UCB algorithm can obtain the minimum cost $C^i_{min}(t)$ of the i-th MU, which is the cost of the optimal task offloading decision. It can be obtained by brute-force searching. Every 50 time slots, the EPE-UCB algorithm will test whether the value $\min_{i\in\mathbf{A}}\{Q^i_k(t)\} - C^i_{min}(t)$ is greater than μ. If it is greater than μ, we update $Q^i_k(t)$ and $N^i_k(t)$ by multiplying them by λ.

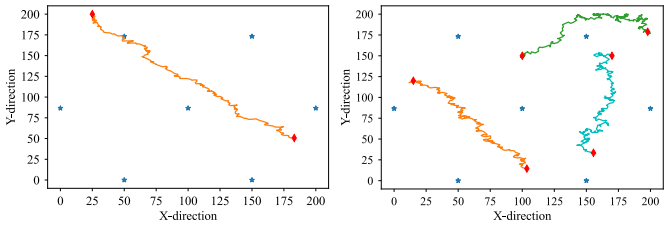
Algorithm 3 EPE-UCB task offloading algorithm

Require: The number of edge servers N, the number of MUs M, μ and λ
Ensure: Task offloading policy
for $i = 1, 2, \ldots, M$ **do**
  for $t = 0, 1, \ldots, T-1$ **do**
    if $t == 0$ **then**
      k is a random value in $\mathbf{A}$;
    else
      $k = \arg\min_{\varpi\in\mathbf{A}}\left(Q^i_k(t) + \sqrt{\frac{2\ln(N^i_k(t))}{N^i_k(t)}}\right)$;
      $Q^i_k(t) = Q^i_k(t-1) + \frac{1}{N^i_k(t-1)}(C^i_t(k) - Q^i_k(t-1))$;
      $N^i_k(t) = N^i_k(t-1) + 1$;
    end if
    if $t\%50 == 0$ **then**
      k is a random value in $\mathbf{A}$;
      Find the minimum cost $C^i_{min}(t)$;
      if $\min_{k\in\mathbf{A}}\{Q^i_k(t)\} - C^i_{min}(t) > \mu$ **then**
        $Q^i_k(t) = \lambda Q^i_k(t)$;
        $N^i_k(t) = \lambda N^i_k(t)$.
      end if
    end if
  end for
end for

IV. Performance Evaluation

In the simulations, we evaluate our algorithm for the scenario with three moving MUs and seven edge servers. The edge servers are in a $200 * 200$ (meters^2) range distributed in a cellular pattern. We set the total number of time slots to 200, i.e., $T = 200$. Each MU generates task of size 0.32 Gbits per time slot. The local computation frequency for each MU is 1.5 GHz. The computation frequency for each edge server is 50 GHz. We set that $p = 0.1$ Watts, $N_0 = -174$ dBm/Hz and $B = 10$ MHz. In our experiments, we set μ as 0.00001 and λ as 0.5.

Fig. 2 shows a simulation scenario with seven edge servers and one or three MUs. The edge servers are fixed while MUs move randomly within a certain range.



(a) One MU (b) Three MUs

Fig. 2: Edge severs ditribution and movements of MUs.

The task offloading decision of the first MU at each time slot is shown in Fig. 3. Besides, the optimal task offloading decision at each time slot is given. The task offloading decision taken by UCB-based task offloading algorithm is not always the optimal one since the first MU moves between multiple edge servers. The task offloading decision taken will be the optimal one when the MU is close to a particular edge server. For example, from time slot 40 to 50, the optimal task offloading decision varies as the MU moving away from one edge server and gradually approaching to another edge server. However, the UCB-based task offloading algorithm still makes the same task offloading decision due to the historical experience. Our EPE-UCB based task offloading algorithm will make the optimal task offloading decision even when the MU moves away from one edge server to another.

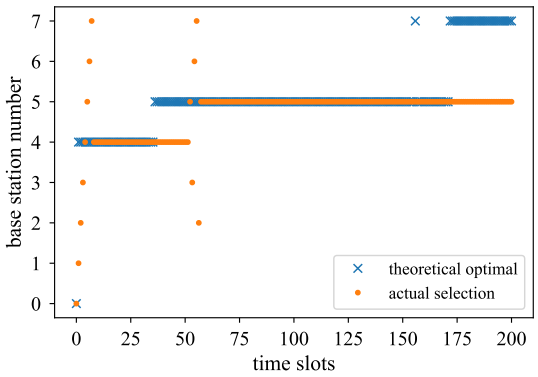


Fig. 3: Comparison of task offloading decision of the UCB-based algorithm and theoretical optimal decision.

Fig. 4 shows the convergence of the costs under the UCB-based, ε-greedy based, random and EPE-UCB task offloading algorithms. The results show that all the four algorithms converge. Moreover, the EPE-UCB task offloading algorithm achieves the lowest cost than the other three algorithms. The results demonstrate the advantage of our EPE-UCB task offloading algorithm in achieving low cost.

In practice, some MUs have high latency requirements, while others need to save cost. Fig. 5 shows that with the changing of the weight between proportions of delay and payment, our EPE-UCB algorithm can achieve the lowest cost compared with the other three algorithms.

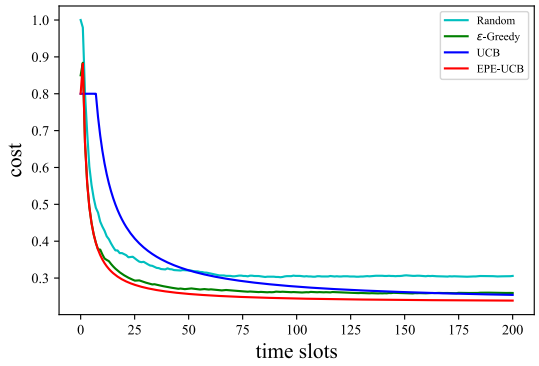


Fig. 4: Costs under four different algorithms

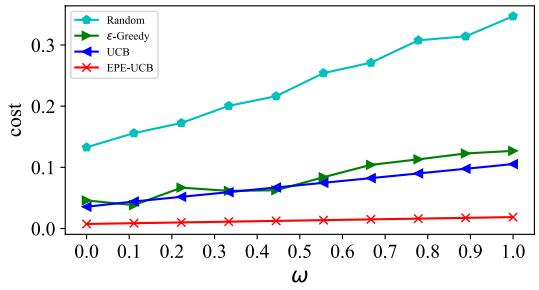


Fig. 5: Cost for different weight between delay and payment

Fig. 6 shows the sum of cost for different numbers of MUs at time slot 200. In the case of different numbers of users, our EPE-UCB algorithm can still achieve the lowest cost compared with the other three algorithms.

The simulation results demonstrate that our algorithm is more applicable to the task offloading problem in mobile scenarios than the UCB-based and ε-greedy based task offloading algorithms.

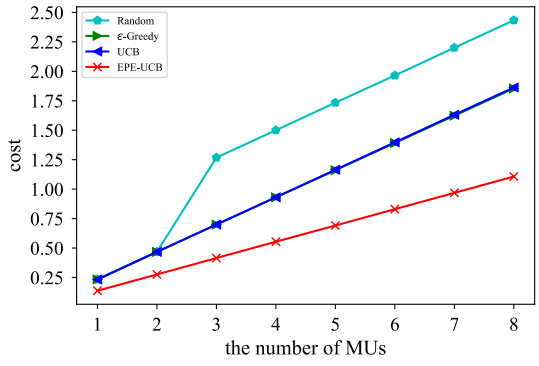


Fig. 6: Cost for different numbers of MUs

V. Conclusion

We studied the task offloading problem of MEC system with multiple moving MUs. We formulated the task offloading problem using MAB model. Based on this, we investigated two task offloading algorithms based on the well-known UCB and ε-greedy algorithms. To better solve the problem, we proposed a novel EPE-UCB task offloading algorithm, which is more applicable to MAB problems in mobile scenarios. It can achieve lower cost and faster convergence. Future work can consider competition for resources by multiple MUs.

References

[1] K. B. Letaief, Y. Shi, J. Lu, and J. Lu, "Edge artificial intelligence for 6g: Vision, enabling technologies, and applications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 5–36, 2021.

[2] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE communications surveys & tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[3] Y. Li, T. Wang, Y. Wu, and W. Jia, "Optimal dynamic spectrum allocation-assisted latency minimization for multiuser mobile edge computing," *Digital Communications and Networks*, vol. 8, no. 3, pp. 247–256, 2022.

[4] M. Mehrabi, D. You, V. Latzko, H. Salah, M. Reisslein, and F. H. Fitzek, "Device-enhanced mec: Multi-access edge computing (mec) aided by end device computation and caching: A survey," *IEEE Access*, vol. 7, pp. 166 079–166 108, 2019.

[5] Y. Mao, J. Zhang, S. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5994–6009, 2017.

[6] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, pp. 235–256, 2002.

[7] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction, second edition," 2017.

[8] F. Li, D. Yu, H. Yang, J. Yu, H. Karl, and X. Cheng, "Multi-armed-bandit-based spectrum scheduling algorithms in wireless networks: A survey," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 24–30, 2020.

[9] K. Liu and Q. Zhao, "Distributed learning in multi-armed bandit with multiple players," *IEEE transactions on signal processing*, vol. 58, no. 11, pp. 5667–5681, 2010.

[10] W. He, D. He, Y. Huang, Y. Zhang, Y. Xu, G. Yun-feng, and W. Zhang, "Bandit learning-based service placement and resource allocation for mobile edge computing," in *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 2020, pp. 1–6.

[11] K. Guo, R. Gao, W. Xia, and T. Q. S. Quek, "Online learning based computation offloading in mec systems with communication and computation dynamics," *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 1147–1162, 2021.

[12] B. Wu, T. Chen, and X. Wang, "An mab approach for mec-centric task-offloading control in multi-rat hetnets," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[13] B. Wu, T. Chen, W. Ni, and X. Wang, "Multi-agent multi-armed bandit learning for online management of edge-assisted computing," *IEEE Transactions on Communications*, vol. 69, no. 12, pp. 8188–8199, 2021.

[14] B. Wu, T. Chen, K. Yang, and X. Wang, "Edge-centric bandit learning for task-offloading allocations in multi-rat heterogeneous networks," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3702–3714, 2021.

[15] H. Li, X. Zhong, Y. Ji, X. Wang, and S. Zhang, "Batch based multi-arm bandits with knapsacks for tasks allocation in mec," in *2022 IEEE/CIC International Conference on Communications in China (ICCC)*, 2022, pp. 962–967.