Deep Reinforcement Learning Based Computing Offloading and Resource Allocation Algorithm for Mobile Edge Networks

1st Jinwei Xu
Jiangsu Key Laboratory of Wireless Communications
Nanjing University of Posts and Telecommunications
Nanjing, China
1218012010@njupt.edu.cn

2nd Xu Liu
Jiangsu Key Laboratory of Wireless Communications
Nanjing University of Posts and Telecommunications
Nanjing, China
liuxu@njupt.edu.cn

3rd Xiaorong Zhu
Jiangsu Key Laboratory of Wireless Communications
Nanjing University of Posts and Telecommunications
Nanjing, China
xrzhu@njupt.edu.cn

2020 IEEE 6th International Conference on Computer and Communications (ICCC) | 978-1-7281-8635-1/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICCC51575.2020.9345089

***Abstract*—With the rapid development of Internet, continuous emergence of various innovative applications makes current mobile network face pressure of lower latency and computing capability. Mobile edge computing (MEC) has been proposed to be a promising solution to reduce the delay of interaction between applications and compensate the deficiencies of traditional cloud computing. In this paper, we propose a computing offloading and resource allocation algorithm to deal with problems in mobile edge networks (MEN), including offloading decision, transmission power and computation resources allocation. With the goal of minimizing the total cost of the system, an algorithm combining Deep Reinforcement Learning (DRL) and Genetic Algorithm (GA) is used to obtain an approximate optimal solution for the system. Simulation results prove the effectiveness of the algorithm.**

***Keywords*—mobile edge computing, computing offloading, deep reinforcement learning, resources allocation**

I. Introduction

With the evolution of mobile terminal devices, the development of innovative applications has an urgent request of lower latency and higher volume of computation resources. MEC sinks resources such as computation and storage to the edge of the network. The requests of terminals will be processed directly by the MEC server nearby, which significantly reduces the transmission delay. Therefore, MEC attracts great attention from scholars and experts.

Nowadays, researches on MEC are dedicated to solving problems such as offloading decision, wireless resource allocation and computation resource allocation. Authors in [1] and [2] jointly considered the problem of computing offloading and wireless resources management. Besides, [3] studied the problem of computing offloading among many users, and Game Theory was used to achieve the optimal computing offloading scheme efficiently. However, the authors in [1]-[3] focus on indivisible tasks and are not considered the fine-grained and divisible tasks, such as AR and image processing.

Resource allocation of MEC includes transmission power allocation, channel resources allocation and computation resource allocation. [4] took advantage of gradient descent to find the optimal offloading and transmission power allocation scheme. Authors in [5] designed the offloading mode and power allocation and adopted a hierarchical optimization method to obtain a suboptimal solution, respectively. [6] and [7] considered not only the problem of power allocation under partial offloading conditions, but also the sub-channel allocation during transmission. [8] and [9] studied the allocation of computation resources and pysical resources in mobile edge networks. However, the systems above were composed of a single base station and multiple users which seldom considered UE's selection of SCs and interference among UEs. Therefore, how to design effective and practical computing offloading strategies has become an crucial problem to solve.

Recently, DRL has been applied to achieve optimal computing offloading decisions and resource allocation schemes in MEN[10]-[12]. [10] assigns different execution priorities to the tasks of each user, and adopt Deep Q-learning Network (DQN) to generate the optimal offloading decision. Authors in [11] propose a new reinforcement learning (RL) training algorithm to decrease the computation complexity in large-scale networks. Compared with [11], [12] studeis task offloading and resource management in ultra dense networks (UDN), and DQN is used to solve high-dimensional problems, which greatly reduces the system's task processing delay and improve the utilization of system resource.

In this paper, we consider a computing offloading algorithm based on Deep Q-learning Network and Genetic Algorithm (DQN-GA). With the goal of minimizing the total cost of the system, this algorithm can effectively solve the problems of computing offloading decision, transmission power and computing resources allocation. Firstly, we use DQN to generate the offloading decision and channel allocation decision. Secondly, GA is adopted for solving the non-convex problem of this paper, and the results will be fed back for training DQN model. Through the process of alternating iterative training and solving, an approximately optimal offloading strategy, channel

allocation scheme, computation resource and transmission power allocation descision are obtained.

The rest of this paper is organized as follows. Section II exhibits the system model in this paper. Section III describes and solves the problems in details. Section IV focuses on the design and principle of DQN-GA. Section V shows the simulation structure of the algorithm and the comparison with other algorithms. Section VI gives a conclusion of this article.

II. System Model

A. Network Model

A mobile edge network scenario is composed of multiple users, base stations and MEC servers. The system model is shown in Fig 1. The number of user equipments (UE) is *N*, indicated by $i, i \leftarrow \{1, 2, 3, ..., N\}$, and each UE has a task to be processed. The number of small cell (SC) is M, indicated by $j, j \leftarrow \{1, 2, 3, ..., M\}$. Assuming that all the MEC servers are controlled by an SDN controller. In this case, each UE has a computation task $T_{\mathrm{i}} = (I_i, D_i, \tau_i^{\max})$ to process, where I_i denotes the total data of the task, D_i represents the CPU cycles consumed for processing one-bit data, $\tau_i^{\max}$ is the maximum latency required by each task. Each UE can choose to compute the task locally or offloading to the MEC server for execution.

In this paper, we apply partial offloading to the fine-grained and divisible tasks. For the UE *i*, proportion of the task to be offloaded to the MEC server at SC *j* is denoted as $a_i^j \leftarrow [0, 1]$ and the proportion to be computed locally is $1 - a_i^j$. Particularly, we assume that each UE can only offload its task to at most one MEC server for processing.

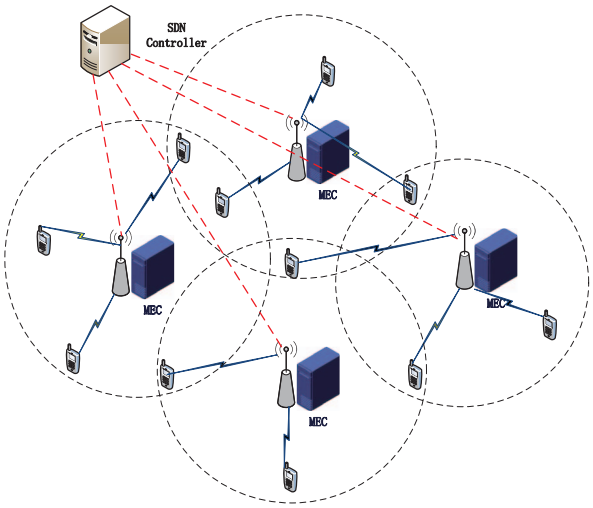

Fig. 1. Network Model

B. Communication model

In this paper, it is assumed that all UEs reuse the channels. The channels occupied by a SC are orthogonal to each other, and the bandwidth is distributed to each UE on average. Therefore, the interference among UEs from different SCs is taken into consideration.

Assume that number of channels is *K* and the channel allocation indicator is denoteds as $b_i^k \in \{0, 1\}, k \leftarrow K$. When $b_i^k = 0$, it means that UE *i* is assigned with the channel *k*; otherwise, it means the channel *k* is not provided to UE *i*. When UEs among different SCs occupy the same channel for data transmission, interference will occur. The interference is $N_{i,j}^k = \sum_{n \leftarrow N \backslash 0} \sum_{m \leftarrow M \backslash j} a_n^m \cdot b_n^k \cdot \gamma_{n,m}^k \cdot P_n \cdot \left| h_{n,m}^k \right|^2$, where P_n is denoted as the transmission power allocated to UE *n* and $P_{\max}$ is the maximum transmission power of the system[1]. Besides, $h_{n,m}^k$ represents the channel gain between SC *m* and UE *n* on channel *k* and $\gamma_{n,m}^k$ is denoted as the interference indicator function[3]:

$$\gamma_{n,m}^k = \begin{cases} 1, RF_{n,m}^k \geq RF_{th} \\ 0, else \end{cases}, \tag{1}$$

where $RF_{n,m}^k$ is the interference power received while RF_{th} represents the threshold of interference. When the interference power exceeds the threshold, we have $\gamma_{n,m}^k = 1$; otherwise, we have $\gamma_{n,m}^k = 0$. Therefore, when UE *i* under SC *j* on the channel *k* choose to offload task to MEC servers, the data rate of transmission can be calculated as:

$$R_{i,j}^k = W \log_2 \left(1 + \frac{P_i \left| h_{i,j}^k \right|^2}{N_0 + N_{i,j}^k} \right), \tag{2}$$

where N_0 is additive white Gaussian noise and *W* is the bandwidth assigned to UEs.

C. Computation model

In this part, we will introduce local computing model and edge computing model separately.

1) Local computing model: when the UE decides to process its task locally, the latency and energy consumption of local computing are given by T_i^l and E_i^l:

$$T_i^l = \frac{I_i D_i \left(1 - a_i^j\right)}{F_i^l}, \tag{3}$$

$$E_i^l = \delta_i I_i D_i \left(1 - a_i^j\right), \tag{4}$$

where δ_i is a fixed constant representing the energy consumption per cycle of UE *i*. F_i^l is denoted as the computation capacity of UE *i*.

2) Edge computing model: when UE chooses to offload the task to MEC servers for processing, it needs to three steps: UE transmits the task to the MEC server, MEC servers process the offloaded tasks and MEC servers return results to UEs.

In the first step, the delay and energy consumption mainly come from task transmission. When UE decides to offload the task to the MEC server, a channel should be allocated to it for transmission to avoid useless offloading decision. The latency and energy consumption can be given as $T_{i,j}^{m,tran}$ and $E_{i,j}^{m,tran}$:

$$T_{i,j}^{m,tran} = \frac{I_i a_i^j}{R_{i,j}^k}, \tag{5}$$

$$E_{i,j}^{m,tran} = P_i T_{i,j}^{m,tran} = \frac{P_i I_i a_i^j}{R_{i,j}^k}. \quad (6)$$

In the second step, we define the computation resources allocated to the UE as F_i^m, which satisfies the following constraints: (1) $\sum_{i \leftarrow N} F_i^m \leq F_m, \forall i \leftarrow N$, (2)$F_i^m \leftarrow [0, F_{\max}]$, where $F_{\max}$ denotes the maximum computation resources of the system. Since all MEC servers are directly connected to the SCs, the energy consumpted by MEC servers are not considered[1]. Thus, the delay caused by task execution is $T_{i,j}^{m,exe}$:

$$T_{i,j}^{m,exe} = \frac{I_i D_i a_i^j}{F_i^m}. \quad (7)$$

In the third step, both the delay and energy consumption are ignored because the output data processed by MEC servers is rather smaller than the input. Therefore, the total of delay T_i^m and energy consumption E_i^m of UE *i* during task offloading can be given as follows:

$$T_i^m = T_{i,j}^{m,tran} + T_{i,j}^{m,exe}, \quad (8)$$

$$E_i^m = E_{i,j}^{m,tran}. \quad (9)$$

III. Problem Formulation

According to the analysis in Section II, during the task offloading of UE *i*, the total energy consumption E_i and task execution latency T_i of the system can be expressed as:

$$E_i = E_i^l + E_i^m = E_i^l + E_{i,j}^{m,tran}, \quad (10)$$

$$T_i = \max\left(T_i^l, T_i^m\right) = \max\left(T_i^l, \left(T_{i,j}^{m,tran} + T_{i,j}^{m,exe}\right)\right). \quad (11)$$

In this paper, the total cost of the system includes the weighted sum of the energy consumption and delay of all UEs, which is described as:

$$\begin{aligned} \mathrm{Cost}(A,B,P,F) &= \sum_{i \leftarrow N_u} \partial_t T_i + \partial_e E_i \\ &= \sum_{i \leftarrow N_u} \partial_t \max\left(T_i^l, T_i^m\right) + \partial_e \left(E_i^l + E_i^m\right) \\ &= \sum_{i \leftarrow N} \partial_t \cdot \max\left(T_i^l, \left(T_{i,j}^{m,tran} + T_{i,j}^{m,exe}\right)\right) \\ &\quad + \partial_e \left(E_i^l + E_{i,j}^{m,tran}\right), \end{aligned} \quad (12)$$

where ∂_t and ∂_e represent the weighted factors of delay and energy consumption ranged from 0 to 1, respectively. Besides, *A* represents the set of offloading decisions and *B* is defined as channel allocation decisions. *P* denotes transmission power allocations for UEs. *F* shows computation resources allocation of MEC servers for UEs.

$$A = \left\{a_1^1 ... a_1^j, ..., a_N^1 ... a_N^j\right\} \quad (13)$$

$$B = \left\{b_1^1 ... b_1^K, ..., b_N^1 ... b_N^K\right\} \quad (14)$$

$$P = \{P_1, P_2, ... P_N\} \quad (15)$$

$$F = \{F_1^m, F_2^m ... F_N^m\} \quad (16)$$

The joint optimization of computation offloading, channel allocation and resources allocation are taken into consideration in this paper. Then, the corresponding optimization problem of minimizing the total cost of the system can be formulated as P1:

$$P1: Q^*(A,B,P,F) = \min_{A,B,P,F} \mathrm{Cost}(A,B,P,F) \quad (17a)$$

$$\text{s.t.} \quad C1: a_i^j \leftarrow [0,1], \forall i \leftarrow N, j \leftarrow M \quad (17b)$$

$$C2: b_i^k \leftarrow \{0,1\}, \forall i \leftarrow N, k \leftarrow K \quad (17c)$$

$$C3: P_i \leftarrow [0, P_{max}], \forall i \leftarrow N \quad (17d)$$

$$C4: 0 \leq F_i^m \leq F_m, \forall i \leftarrow N \quad (17e)$$

$$C5: 0 \leq F_i^m \leq F_m, \forall i \leftarrow N \quad (17f)$$

$$C6: \sum_{i \leftarrow N} \sum_{k \leftarrow K} b_i^k \leq K, \forall j \leftarrow M \quad (17g)$$

$$C7: T_i \leq \tau_i^{\max}, \forall i \leftarrow N \quad (17h)$$

$$C8: \sum_{i \leftarrow N} P_i \leq P_{\max}, \forall i \leftarrow N \quad (17i)$$

where C1 and C2 represent the constraints of the offloading decision and channel allocation decision respectively. C3-C4 enforces the uploading transmission power allocation constraint, and C5-C6 describe the constraint of computation resources allocation to each UE. C7 represents the latency constraint of each UE, and C8 enforces the constraint of channel allocation to each UE.

The optimization problem P1 is a mixed integer nonlinear problem(MINLP), which is a NP-hard problem. However, once the offloading decisions and channel allocations are given, P1 can be simplified to a problem without integer variables which is denoted as P2. P2 aims to solve transmission power and computing resources allocation and it can be described as follows:

$$P2: Q^*(A,B) = \min_{P,F} \mathrm{Cost}(A,B,P,F) \quad (18a)$$

$$\text{s.t.} \quad C1: 0 \leq F_i^m \leq F_{\max}, \forall i \leftarrow N \quad (18b)$$

$$C2: \sum_{i \leftarrow N_u} F_i^m \leq F_{\max}, \forall i \leftarrow N \quad (18c)$$

$$C3: P_i \leftarrow [0, P_{max}], \forall i \leftarrow N \quad (18d)$$

$$C4: \sum_{i \leftarrow N_u} P_i \leq P_{\max}, \forall i \leftarrow N \quad (18e)$$

Denoting $l1 = \frac{I_i D_i \left(1 - a_i^j\right)}{F_i^l}$, $l2 = \frac{I_i a_i^j}{R_{i,j}^k} + \frac{I_i D_i a_i^j}{F_i^m}$ and $l3 = W \log_2\left(P_i \left|h_{i,j}^k\right|^2\right) - \log_2\left(N_0 + N_{i,j}^k\right)$, P2 can be converted

to problem P3 according to $R_{i,j}^k$:

$$\begin{aligned} P3: Q^*(A,B) &= \min_{P,F} \mathrm{Cost}(A,B,P,F) \\ &= \min_{P,F} \sum_{i \leftarrow N} \partial_t \cdot \max\left(T_i^l, \left(T_{i,j}^{m,tran} + T_{i,j}^{m,exe}\right)\right) \\ &+ \partial_t \cdot \left(E_i^l + E_{i,j}^{m,tran}\right) \\ &= \min_{P,F} \sum_{i \leftarrow N} \partial_t \cdot \max\left(l1, l2\right) \\ &+ \partial_e \left(\delta_i I_i D_i \left(1 - a_i^j\right) + \frac{P_i I_i a_i^j}{R_{i,j}^k}\right) \\ &\overset{\frac{S}{N}>>1}{\approx} \min_{P,F} \sum_{i \leftarrow N} \partial_t \cdot \max\left(l1, \left(\frac{I_i a_i^j}{l3} + \frac{I_i D_i a_i^j}{F_i^m}\right)\right) \\ &+ \partial_e \cdot \left(\delta_i I_i D_i \left(1 - a_i^j\right) + \frac{P_i I_i a_i^j}{l3}\right) \end{aligned} \tag{19}$$

For convenience, we denote $X1 = I_i D_i \left(1 - a_i^j\right) \cdot \left(\frac{\partial_t}{F_i^l} + \partial_e \cdot \delta_i\right) + \frac{\partial_e \cdot P_i I_i a_i^j}{l3}$, $Y1 = \partial_e \cdot \delta_i I_i D_i \left(1 - a_i^j\right) + \frac{\partial_t \cdot I_i a_i^j + \partial_e \cdot P_i I_i a_i^j}{l3} + \partial_t \cdot \frac{I_i D_i a_i^j}{F_i^m}$. Hence, P3 can be finally summarized as:

$$Q^*(A,B) = \min_{P,F} \sum_{i \leftarrow N} \begin{cases} X1, & l1 > l2 \\ Y1, & l1 \leq l2 \end{cases} \tag{20}$$

.

Then, we adopt GA to obtain the optimal P_i and F_i^m of P3. At the same time, the couple of decisions in P1 corresponding to the solution of P3 will be fed back for training DQN. Through solving P1 and P3 alternately, the approximately optimal solutions of the entire system will be achieved.The specific solution steps of GA will be explained in detail in the Section IV.

IV. Deep Reinforcement Learning Based Computing Offloading and Resource Allocation Algorithm

A. DQN-based offloading decision and channel allocation

In this section, we describe the computation offloading process as an infinite discount continuous state Markov Decision Process (MDP). To solve the problem, DQN is used to generate offloading and channel allocation decisions. The crucial elements of DQN algorithm are given below.

System state: during each decision period *t*, the network status of each UE can be characterized by the channel gain at the moment, the state space of the DQN can be denoted as $S_t = \{h_t\}$, which will be fed into DNN to generate the offloading and channel allocation decisions.

Action apace: the action space contains two parts: offloading decisions and channel allocation decisions, denoted as $A = \{a_t, b_t\}$. DNN is used to generate of the action space, and the optimal action space is obtained by solving the optimization problem of P3.

Reward function: the optimization goal of DQN is to maximize the long-term benefits of the system, while the optimization goal of this paper is to minimize the total cost of the system. Hence, the reward value of DQN in this paper is defined as:

$$reward = \frac{C_{local} - C_{curr}}{C_{local}}, \tag{21}$$

where C_{local} represents the total cost when all the tasks are executed locally and C_{curr} represents the total cost of the system at the moment.

B. GA-based computation resource and transmission power allocation

In this part, we take $\{a_t, b_t\}$ as the input of P3 and use GA to achieve the solution of P_i and F_i^m. In order to estimate the fitness of individuals in the algorithm, we can define the fitness function as:

$$\begin{aligned} Fitness = &\sum_{i \leftarrow N} \partial_t \cdot \max\left(T_i^l, \left(T_{i,j}^{m,tran} + T_{i,j}^{m,exe}\right)\right) \\ &+ \partial_e \left(E_i^l + E_{i,j}^{m,tran}\right) + Pe \end{aligned} \tag{22}$$

where $Pe = \sum_{i \leftarrow N} \partial_j \left[\max\left(0, \max\left(T_i^l, T_i^m\right) - \tau_i^{\max}\right)\right]$ represents the penalty value and ∂_j represents the penalty factor, which tends to be positive infinity. Each chromosome of the individual is denoted as a solution to the optimization problem and it is described as:

$$C_i^t = \left[C_1, C_2, C_3 ... C_N\right], \tag{23}$$

where $C_i = [p_i^*, f_i^*]^{\mathrm{T}}$ represents the optimal transmission power and computing resource allocation scheme of user *i*.

In order to avoid local optimal solutions, we adopt the traditional selection, crossover and mutation operations alternately to improve the diversity of individuals. The main steps of the algorithm are as follows:

Step 1: for the part of selection, during the selection of *K* parents, the most suitable individual is selected from several individuals to maintain the individual diversity. If the selected parent set does not contain the best individual, then the best individual will be used to replace the worst one.

Step 2: for the part of crossover, two parents will be randomly selected and exchand each row of data segments corresponding to the two parents with a probability of P_c. That is, exchanging each data segment of C_i with a probability of P_c, and thus two descendants are produced.

Step 3: for the part of mutation, each individual will be selected in turn to perform the chromosome mutation operation. For example, p_i and f_i will mutate among the range of the variables at the probability of P_m. As a result, the value of p_i ranges from 0 to $P_{\max}$, while f_i ranges from 0 to $F_{\max}$.

V. Simulation Results

In this part, the performance of the DQN-GA is evaluated through simulations. Multiple SCs and UEs are randomly distributed in a $200 \times 200m^2$ and the radius of each SC is 50 m. Assume that all SCs share 8 channels and the bandwidths of them are all set to 100 kHz. The computation capacity of each UE and MEC server is set [1, 2] and 10, respectively .

Algorithm 1 Deep Reinforcement Learning Based Computing Offloading and Resource Allocation Algorithm (DQN-GA)

1: Initialize an experience pool with a size of 1000 and the parameter of DNN with θ.
2: Define the maximum number of algorithm iteration N and set the training interval of DNN with Δ.
3: For t = 1, 2,..., N do:
4: Take advantage of DQN to generate a group of offloading decision and channel allocation decision $(a_t, b_t) = f_{\theta_t}(h_t)$.
5: Utilize KNN to quantify (a_t, b_t) into several feasible action decisions $\left(\hat{a_t}, \hat{b_t}\right)$.
6: For all $\left(\hat{a_t}, \hat{b_t}\right)$, solve the problem (P3) by GA to achieve $Q^*\left(\hat{a_t}, \hat{b_t}\right)$.
7: Select the action $\left(\hat{a_t^*}, \hat{b_t^*}\right)$ of $Q^*\left(\hat{a_t}, \hat{b_t}\right)$ as the optimal decision (a_t^*, b_t^*) at time t.
8: Add (h_t, a_t^*, b_t^*) to the experience pool for DNN training.
9: **if** $t \bmod \Delta = 0$ **then**
10: Select a small batch of data from the experience pool randomly.
11: Using (h_t, a_t^*, b_t^*) to train and update the parameters of DNN.
12: **end if**
13: end

Fig. 2 describes the convergence of the algorithm proposed in this paper. During the process of achieving offloading decisions and resource allocations with DQN-GA, the training loss of the algorithm continues to decrease until the algorithm convergences.

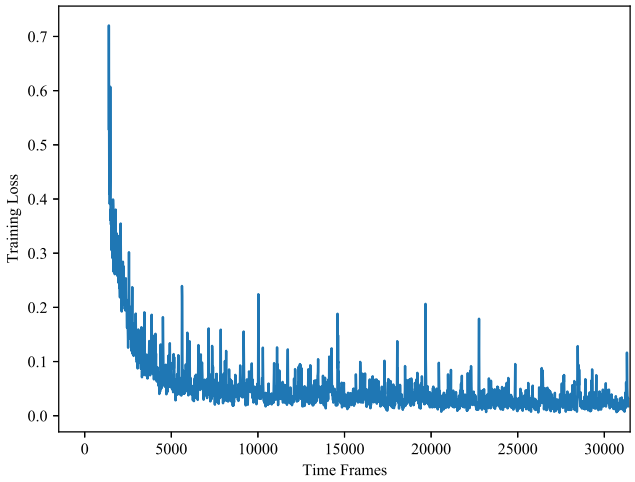

Fig. 2. The convergence of DQN-GA

Fig. 3 verifies the performance of the algorithm when the number of UEs continues to increase. The number of UEs varies from 10 to 40. We consider the comparison of four situations: Full-Local, Full-MEC, Q-learning and DQN-GA. It can be seen from Fig.3 that Full-Local makes the total cost of the system increase almost linearly, which is much higher than the others. For the case of Full-MEC, due to the limited computing resources of MEC servers, when the number of UEs increases, the computing resources allocated to UEs decreases and the growth rate of total cost continues to increase. Compared with Q-learning, the performance of DQN-GA is slightly better and both of them still perform well under the condition of large number of UEs.

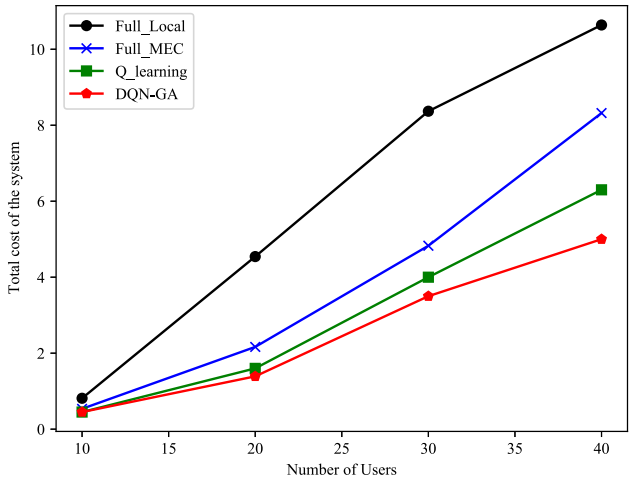

Fig. 3. Total cost of the system in different numbers of UEs

Fig. 4 shows the performance of the algorithm when the computing resources of MEC servers continue to increase. In Fig. 4, the total cost of the system of Full-Local remains unchanged, since the system cost of Full-Local mode has nothing to do with the size of MEC computing resources. At the same time, with the increase of MEC computing resources, the application of Full-MEC, Q-learning, and DQN-GA continuously reduced the total energy consumption of the system. At last, the total cost in the last three cases is almost the same.

Fig. 5 indicates the performance of the algorithm when the size of tasks continue to increase. It can be seen from Fig. 5 that the total cost of the system in the Full-Local solution mode grows rapidly because of limited computation capacity of UEs. Compared with Full-Local mode, the solution of Full-MEC, Q-Learning and DQN-GA can still maintain the total cost at a lower level as the sizes of tasks increase. Finally, it can be inferred that the DQN-GA designed in this paper performs better than others in decreasing system cost.

VI. CONCLUSION

In this paper, we mainly study the computing offloading and resource allocation schemes in mobile edge networks.

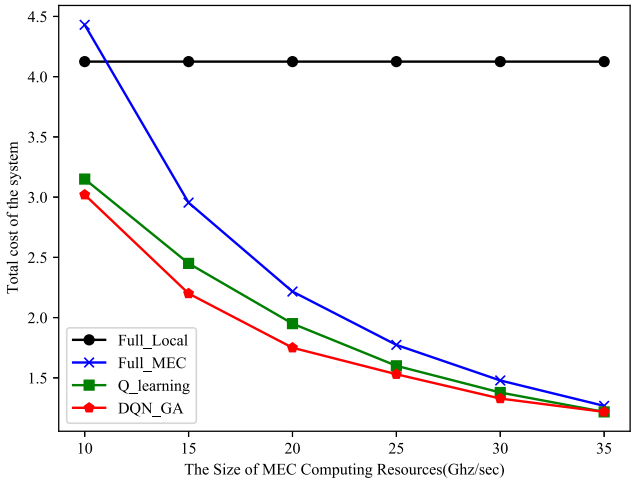

Fig. 4. Total cost of the system in different computation resources of MEC servers

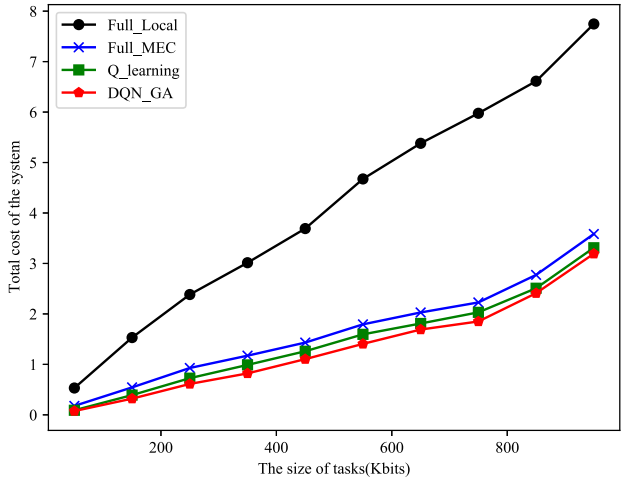

Fig. 5. Total cost of the system in different size of tasks

We comprehensively consider the problems of offloading decisions, interference, computation resources and transmission power allocation. Firstly, DQN is used to generate the offloading and channel allocation decisions. Secondly, based on the decisions achieved, GA is used to obtain the optimal computation resources and power allocation at each epoch. Resluts of GA and the corresponding offloading decisions will be fed back for DQN training. Simulation results show that compared with the existing algorithms, DQN-GA can not only effectively reduce the total cost of the system under the same conditions, but also perform well as the number of UEs or the data size of tasks continues to increase.

ACKNOWLEDGMENT

This work was sponsored by National Natural Science Foundation of China(61871237), NUPTSF(Grant No.NY217028), Program to Cultivate Middle-aged and Young Science Leaders of Universities of Jiangsu Province and Key R&D plan of Jiangsu Province(BE2019017), China Postdoctoral Science Foundation(2012M521106).

REFERENCES

[1] F. Guo, H. Zhang, H. Ji, X. Li and V. C. M. Leung, "Energy Efficient Computation Offloading for Multi-Access MEC Enabled Small Cell Networks," 2018 IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, MO, 2018, pp. 1-6.

[2] C. Wang, F. R. Yu, Q. Chen and L. Tang, "Joint computation and radio resource management for cellular networks with mobile edge computing," 2017 IEEE International Conference on Communications (ICC), Paris, 2017, pp. 1-6.

[3] X. Chen, L. Jiao, W. Li, and X. Fu, "Energy Efficient multi-user computation offloading for mobile-edge cloud computing," IEEE/ACM Transactions on Networking, 2016, pp. 2795–2808

[4] H. Sun, F. Zhou and R. Q. Hu, "Joint Offloading and Computation Energy Efficiency Maximization in a Mobile Edge Computing System," in IEEE Transactions on Vehicular Technology, 2019, pp. 3052-3056.

[5] Z. Kuang, L. Li, J. Gao, L. Zhao and A. Liu, "Partial Offloading Scheduling and Power Allocation for Mobile Edge Computing Systems," in IEEE Internet of Things Journal, 2019, pp. 6774-6785.

[6] M. Li, S. Yang, Z. Zhang, J. Ren and G. Yu, "Joint subcarrier and power allocation for OFDMA based mobile edge computing system," 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), Montreal, QC, 2017, pp. 1-6.

[7] S. Mu, Z. Zhong, D. Zhao and M. Ni, "Latency Constrained Partial Offloading and Subcarrier Allocations in Small Cell Networks," ICC 2019 - 2019 IEEE International Conference on Communications (ICC), Shanghai, China, 2019, pp. 1-7.

[8] L.Tong, Y. Li and W. Gao, "A hierarchical edge cloud architecture for mobile computing," IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, 2016, pp. 1-9.

[9] C. Wang, F. R. Yu, C. Liang, Q. Chen, L. Tang. "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," IEEE Transactions on Vehicular Technology, vol. 66, no. 8, pp. 7432–7445, Aug 2017.

[10] P. Yao, X. Chen, Y. Chen and Z. Li, "Deep Reinforcement Learning Based Offloading Scheme for Mobile Edge Computing," 2019 IEEE International Conference on Smart Internet of Things (SmartIoT), Tianjin, China, 2019, pp. 417-421.

[11] L. Huang, S. Bi and Y. J. Zhang, "Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks," in IEEE Transactions on Mobile Computing.

[12] J. Lv, J. Xiong, H. Guo and J. Liu, "Joint Computation Offloading and Resource Configuration in Ultra-Dense Edge Computing Networks: A Deep Reinforcement Learning Solution," 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), Honolulu, HI, USA, 2019, pp. 1-5.