Deep Reinforcement Learning based Path Planning for UAV-assisted Edge Computing Networks

Yingsheng Peng, Yong Liu, and Han Zhang
School of Physics and Telecommunication Engineering, South China Normal University, Guangzhou, 510026, China
Email:2019021962@m.scnu.edu.cn,yliu@m.scnu.edu.cn, zhanghan@scnu.edu.cn

2021 IEEE Wireless Communications and Networking Conference (WCNC) | 978-1-7281-9505-6/20/$31.00 ©2021 IEEE | DOI: 10.1109/WCNC49053.2021.9417292

***Abstract*—Mobile edge computing (MEC) harvests the computation capability at the network edge to perform the computation intensive tasks for diverse IoT applications. Meanwhile, the unmanned aerial vehicle (UAV) has a great potential to flexibly enlarge the coverage, and enhance the network performance. Accordingly, it has been a promising paradigm to use the UAV to provide the edge computing service for massive IoT devices. This paper studies the path planning problem of a UAV-assisted edge computing network, where an UAV is deployed with an edge server to execute the computing tasks offloaded from multiple devices. We consider the mobility of devices, where a Gauss-Markov random movement model is adopted. By taking the energy consumed for the dynamic flying and executing the tasks at the UAV into account, we formulate a path planning problem that aims to maximize the amount of offloaded data bits by the devices while minimizing the energy consumption of the UAV. To deal with the dynamic change of the complex environment, we apply the deep reinforcement learning (DRL) method to develop an online path planning algorithm based on double deep Q-learning network (DDQN). Extensive simulation results validate the effectiveness of the proposed DRL-based path planning algorithm in terms of the convergence speed and the system reward.**

***Index Terms*—Deep Reinforcement Learning,Gauss-Markov stochastic model,Mobile edge computing,Unmanned Aerial Vehicle,Path optimization**

I. Introduction

Mobile edge computing (MEC) enables the computing power of the network edge to flexibly and quickly deploy innovative applications and services for massive IoT devices [1]. With the deployment of MEC, the devices can transfer their computation-intensive tasks to the nearby powerful edge servers for reducing latency and saving energy [1], [2]. Unlike fixed edge servers, recent works on the MEC have been devoted to mobile edge servers, which can provide more flexible, more economical and efficient computing services in harsh environments. Recently, some literature have proposed to use the UAV to improve the connectivity of ground-based IoT devices [3]. UAV-assisted wireless communication has shown its advantage in flexible deployment, fully controllable mobility and enhancing the network performance, such that it has attracted growing research interests. Accordingly, the UAV-assisted edge computing network is a natural choice and a promising paradigm, in which how to optimize the flying path of the UAV to meet with the communication and computing requirements of massive devices becomes an important and challenging issue.

Recently, some existing literatures has studied path planning problem in UAV-assisted mobile edge computing networks. In [4], the trajectory of the UAV and bit allocation were jointly optimized under the constraints of delay and energy consumption of the UAV. While, in those works, the devices were supposed to be fixed, and the mobility was not taken into consideration. In practice, the devices may change dynamically over time, and then the UAV need to adjust its trajectory accordingly to the time-varying position of mobile devices. Meanwhile, the aforementioned works mainly focused on the traditional optimization-based path planning algorithm, which may not always works well with the increase of the number of the UAV and devices since the explosive growth of optimizing variables makes the method not efficient [5]. In [6], by using deep neural network (DNN) for function approximation, deep reinforcement learning (DRL) has been proved to be effective in approximating the Q value. Afterward, the DRL has been employed to the design of online resource allocation and scheduling in wireless networks [7]–[9]. Specifically, in [7], the total system cost of execution delay and energy consumption of multi-user MEC network was minimized by optimizing offload decisions and computing resource allocation. An online offloading algorithm was proposed in [8] to maximize the weighted sum calculation rate for a wireless powered MEC network that enables the wireless energy harvesting capability. In [9], the computing offloading strategy of IoT devices based on deep reinforcement learning was investigated. While, to the best of our knowledge, few existing works have studied how to intelligently design the flight trajectory of the UAV in a mobile edge computing network to serve the massive devices, especially considering the dynamic mobility of devices and dynamic association between the UAV and devices.

In this paper, we study the path planning for a UAV-assisted edge computing network, in which an UAV is deployed with an edge server to execute the computing tasks offloaded from multiple IoT devices. Considering the dynamic mobility of devices in practical scenario, we adopt a Guass-Markov random movement model. On one hand, in order to explore the provided computing capability of the UAV, we aim to maximize the amount of data bits of tasks offloaded from devices to the UAV. On other hand, to prolong the lifetime of the UAV, the total energy consumption of the UAV for both the flying and task execution need to be minimized. Thus, we formulate a path planning problem to optimize the weighted amount of offloaded data bits and the energy consumption. To

978-1-7281-9505-6/21/$31.00 ©2021 IEEE

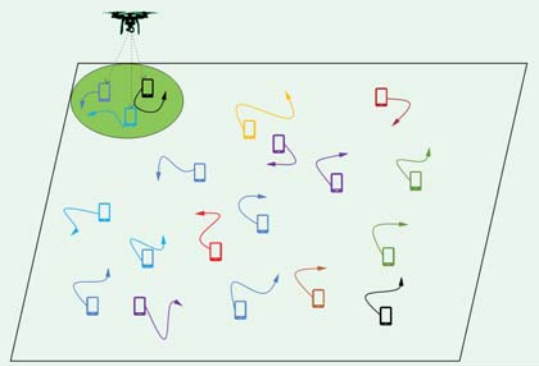

Fig. 1. System model

counter the dynamic and complex environment, we use the deep reinforcement learning (DRL) framework, and propose an online path planning algorithm based on double deep Q-learning network (DDQN), which takes the observation of the surrounding environment as the input, and predicts the reward of the action by training the depth neural network. Finally, the simulation results validate the performance improvement of the proposed DRL-based path planning algorithm in terms of the convergence speed and the system reward, namely the weighted amount of offloaded data bits and the energy consumption.

The rest of this article is organized as follows. The system model and formulated problem are introduced in Section II. In Section III, the deep reinforcement learning based path planning algorithm is proposed. The simulation results are presented in Section IV, followed by the conclusions.

II. System Model and Problem Formulation

This paper considers a time-slotted UAV-assisted edge computing network as illustrated in Fig. 1, where an UAV is deployed with an edge server, and can provide computing services for N IoT devices. Each device can randomly move from one position to another, and generate a certain amount of computing task in each time slot. To complete these tasks, devices with constrained computing capability usually offload their tasks to the UAV, which is employed with powerful computing capability.

Limited by the effective communication distance, the UAV can only serve the devices within its coverage area, thus the dynamic flying and hovering of the UAV can further enlarge the coverage area to serve all devices. To simplify the analysis, we consider that the UAV can only hover in one of M fixed access points. When the UAV is hovering, it can help to execute the tasks of devices in the current area, which is called the association between the UAV and devices.

A. Mobility Model of IoT Devices

Considering the mobility of devices, the Gauss-Markov random movement model [10] is adopted in this paper. Accordingly, the mobility speed $v_i(t)$ and direction $\phi_i(t)$ of the i-th device in the time slot t are updated as

$$v_i(t) = c_1 v_i(t-1) + (1-c_1)\,\bar{v} + \sqrt{1-c_1^2}\Phi_i, \quad (1)$$

$$\phi_i(t) = c_2 \phi_i(t-1) + (1-c_2)\,\bar{\phi}_i + \sqrt{1-c_2^2}\Psi_i. \quad (2)$$

In above expression, $0 \le c_1, c_2 \le 1$ are the parameters used to adjust the impact of the previous state on the current state, $\bar{v}$ is the common averaged speed of all devices, $\bar{\phi}_i$ is the average direction of i-th device, and Φ_i and Ψ_i are two random variables that follow the independent Gaussian distributions.

Given the mobility speed and direction, the position coordinates of the i-th device are given by [11]

$$x_i^{TD}(t) = x_i^{TD}(t-1) + v_i(t-1)\cos\left(\phi_i(t-1)\right)T_0, \quad (3)$$

$$y_i^{TD}(t) = y_i^{TD}(t-1) + v_i(t-1)\sin\left(\phi_i(t-1)\right)T_0, \quad (4)$$

where T_0 represents the duration of each time slot. Usually, T_0 is small enough, such that the positions of devices can be considered to remain unchanged in any time slot, but vary from one slot to another.

B. Edge computing at the UAV

In each time slot, the UAV need to collect and execute the computing tasks of the devices in its coverage area. Denote $\mu_i(t)$ as the number of offloaded tasks offloaded at the i-th device in the time slot t, and N_i is the number of data bits per task. Accordingly, in the time slot t, the amount of data bits of the tasks offloaded from the i-th device to the UAV is

$$\lambda_i(t) = \mu_i(t) N_i, \quad (5)$$

Given the position of devices in (3) and (4), we can obtain the horizontal distance between the i-th device and the UAV as

$$R_i(t) = \sqrt{\left[x^U(t) - x_i^{TD}(t)\right]^2 + \left[y^U(t) - y_i^{TD}(t)\right]^2}, \quad (6)$$

where $x^U(t), y^U(t)$ are the position coordinates of the UAV that are chosen from the set of the pre-determined M access points.

If the i-th mobile device wants to offload its task to the UAV, it must be located in the coverage area of the UAV and the horizontal distance satisfy [12]

$$R_i(t) \le H \tan\beta, \quad (7)$$

where H is the altitude of the UAV from the ground. It is assumed that UAV is equipped with a directional antenna of adjustable beamwidth. The azimuth and elevation half-power beamwidths of antenna are equal for UAV, which are both denoted by $\beta \in \left(0, \frac{\pi}{2}\right)$. We use a binary variable $a_i(t) = 1$ to indicate whether the i-th device is in the coverage area of the UAV or not, which is given by

$$a_i(t) = \begin{cases} 1, \mathrm{R}_i(t) \le H\tan\beta, i \in N \\ 0, \text{ else,} \end{cases} \quad (8)$$

where $a_i(t)$ also reflects the association relationship between the device and the UAV. To this end, we can obtain the set of mobile devices that are in the coverage area of the UAV in the slot t as $\mathcal{M}(t) = \{i \mid i \in \{R_i(t) \leq H \tan\beta\}\}$, where $M(t) = |\mathcal{M}(t)| = \sum_{i=1}^{N} a_i(t)$ is the total number of associated devices. Thus, the total amount of tasks offloaded to the UAV is

$$\lambda(t) = \sum_{i=1}^{N} a_i(t)\lambda_i(t). \tag{9}$$

C. Energy Consumption Model of the UAV

In fact, the energy management of the UAV plays the significant role to prolong the lifetime of the UAV and achieve a good network performance. The energy of the UAV is not only consumed for the dynamic flying, but also used to complete the computing tasks offloaded from devices. When the UAV is flying from the one access point to another in the time slot t, the flying energy consumption is given by [11]

$$e_f(t) = P_f \frac{L}{V}, \tag{10}$$

where L=$\sqrt{[x^U(t) - x^U(t-1)]^2 + [y^U(t) - y^U(t-1)]^2}$ is the flying distance, V and P_f represent the flying speed and power of the UAV, respectively. When the position selected by the UAV remains unchanged from the time slot t-1 to the time slot t, we assume that the energy consumed by the UAV during this period is $e_f(t) = P_0$. The computing energy consumption is dependent on the total amount of data bits offloaded to the UAV and the CPU frequency of edge server, which is given by [13]

$$e_c(t) = \gamma_c C f_c^2 \lambda(t), \tag{11}$$

where γ_c is the effective switched capacitor, C is the number of CPU cycles per bit, and f_c is the CPU frequency of edge server. Therefore, the total energy consumption of the UAV in the time slot t is

$$e(t) = e_c(t) + \eta e_f(t), \tag{12}$$

where η is a penalty coefficient of flight energy consumption, which is set to reduce the difference of magnitude. Therefore, the total remaining energy consumption of the UAV in the time slot t is

$$B(t+1) = B(t) - e(t), \tag{13}$$

where the initial energy is set to $B(0) = B_0$.

D. Problem Formulation

To explore the provided computing capability of the UAV, we need to maximize the amount of data bits of tasks offloaded from devices to the UAV. Meanwhile, to prolong the lifetime, the total energy consumption have to be minimized. Thus, our goal is to optimize the flying path of the UAV and manage the energy resource to maximize the weighted amount of data bits of offloaded computing tasks and the energy consumption, and we can formulate the following optimization problem

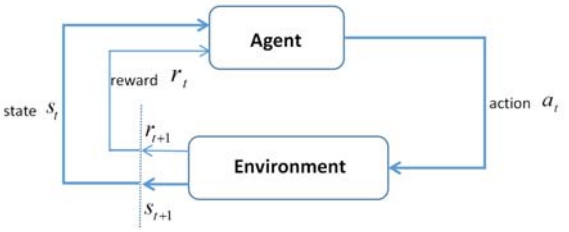

Fig. 2. An example model of reinforcement learning method.

$$\begin{aligned} \max_{x^U(t), y^U(t)} \quad & \frac{1}{T} \sum_{t=0}^{T-1} \sum_{i \in \mathcal{M}(t)} \sigma\lambda_i(t) - \varphi e(t) \\ \text{s.t.} \quad & (1) - (13), B(t) \geq 0, \forall t, \end{aligned} \tag{14}$$

where the weight coefficient σ, φ is to balance the order of magnitude. T is the total number of time slots.

III. Deep Reinforcement Learning based Path Planning Design

In this section, to achieve the intelligent path planning and energy management, we employ the deep reinforcement learning to address the formulated optimization problem and obtain the online scheduling scheme.

The UAV can be considered as an intelligent agent to explore unknown external environment. The problem can be modeled as a Markov decision process. As shown in Fig.2, in the time slot t, the agent is in the current environment state s_t, and then takes action a_t. After that, the environment feedback a reward r_{t+1} to the agent, and evolves to the next state s_{t+1}.This procedure will continue until the environment sends a end state. The details of state, action and reward function are presented in the following.

1) State: The state includes the amount of computing tasks of the i-th device $\lambda_i(t)$, the current position coordinates of devices $l_i^{TD}(t) = (x_i^{TD}(t), y_i^{TD}(t))$, and the position coordinate of the UAV at the end of last time slot, $l^U(t-1) = (x^U(t-1), y^U(t-1))$. Meanwhile, the remaining energy of the UAV in the time slot t $B(t)$ can also be used as a state variable. Thus, the state of the agent is described as $s_t = (l_i^{TD}(t), l^U(t-1), \lambda_i(t), B(t))$.

2) Action: The action is to control the flying and determine the hovering position of the UAV to serve IoT devices. In each time slot t, the UAV need to select a fixed access point from the candidate points according to the current state observations and continue the task offloading. We use $a_t = l^U(t) = (x^U(t), y^U(t))$ to denote the action of the agent in the time slot t.

3) Reward: Our goal is to maximize the number of data bits of offloaded tasks with guaranteeing the energy consumption. Thus, the purpose of each action is to maximize the offloaded

data bits under the premise of energy consumption, the system reward need to contain both factors, which is define as

$$r_{t+1} = \sigma\lambda(t) - \varphi e(t). \tag{15}$$

The purpose of reinforcement learning method is to find the optimal strategy π^* (a mapping from the state in its space S to the probability of selecting each action among the action space A), to maximizes the expected reward from any initial state, G_t,

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, 0 \leq \gamma \leq 1, \tag{16}$$

where γ is a discount factor that weighs the importance of immediate and future rewards.

In this paper, we use double deep Q learning and experience replay to train the agent to effectively learn path planning strategies. The action-state value function based on the strategy π, $q_\pi(s,a)$, is given by

$$q_\pi(s,a) = E_\pi \left[G_t \mid s_t = s, a_t = a \right], \tag{17}$$

where G_t is defined in (16). If all of the action-state value function $q_\pi(s,a)$ are obtained, the optimal strategy can be determined as well. The deep Q network (DQN) is a learning algorithm based on value function. In [14], DQN approximates the Q value by using two deep neural networks. One is the the training network, in which the input is the current state-action pair, and the output is the predicted value $Q_p^{DQN}(s_t, a_t; \theta)$. The other is the target network, where its input is the next state, and the output is the maximum Q value of the state-action pair in the next state. The target value of DQN is $Q_t^{\mathrm{DQN}} = r_t + \gamma \max_a Q(s_{t+1}, a; \theta^-)$.

The DQN structure directly selects $\max_a Q(s_{t+1}, a; \theta^-)$ in the target network, and the parameters are not updated over time, which may cause the Q value to be estimated too high, which leads to the final strategy is not optimal, but only sub-optimal [14]. In order to solve the Q value overestimation problem, double deep Q-learning network (DDQN) applies different action-value function selection and evaluation actions, respectively. The difference with DQN is that the target value in DDQN is defined as $Q_t^{DDQN} = r_t + \gamma Q(s_{t+1}, \arg\max_a Q(s_{t+1}, a, \theta); \theta^-)$. Based on the DDQN, in each time slot, both the training and test phases are necessary to achieve the intelligent path planning scheme, which are described as below:

Training phase: In this paper, the path planning scheme is implemented based on the DDQN algorithm. In the procedure of UAV learning and exploration, the information, including the current environment, action, feedback reward, and the state of the next time slot, is integrated as a sample for the training, *i.e.,* $d(t) = (s_t, a_t, s_{t+1}, r_{t+1})$. All collected training samples are stored in the replay memory D. In each episode, a small batch of empirical K is evenly extracted from the memory to update θ with variants of the stochastic gradient descent method to minimize the sum of squares error. Accordingly, the loss function is

$$J(\theta) = \frac{1}{2K} \sum_{k=1}^{K} \left[Q_t^{DDQN}(k) - Q_p^{DDQN}(k) \right]^2, \tag{18}$$

where $Q_t^{DDQN}(k)$ and $Q_p^{DDDN}(k)$ represent the target and predicted value of the k-th sample in K small batch samples, respectively.

The training process uses ε-greedy exploration strategy to balance the proportion of exploration and development. In each slot, the UAV selects the action of largest estimate with probability 1-ε, and randomly chooses others with total probability ε.

Test phase: In this phase, the agent first loads the network parameters it learned during the training phase. Then, the agent starts with an empty data buffer and interacts with a randomly initialized environment. After that, when it obtains its local observation of the environment as the current state, it will choose the action based on the output of the target network.

Above all, the whole procedure of the intelligent path planning can summarized in Algorithm 1.

Algorithm 1 Double deep Q-learning based Path Planning Algorithm

Input:
The state s_t=$\{l_i^{TD}(t), l^U(t-1), \lambda_i(t), B(t)\}$
Output:
Optimal policy π^*

```
Initialize the replay memory D; deep Q-network weights θ; weights θ⁻ by θ; B(0) = B_0, δ, γ and ε;
for j = 1, 2, ....., L_max do
    Randomly initialize position of the UAV and mobile devices; Let t = 0,Generate an initial state s_0 ;
    while B(t) > 0 do
        Take action a_t with ε-greedy policy at s_t:
        Case I: Randomly select an action with the probability of ε
        Case II: Select action: arg max Q(s_{t+1}, a)
        Obtain the reward r_{t+1} and state s_{t+1};
        Store the transition (s_t, a_t, r_{t+1}, s_{t+1}) in the memory D ;
        if t mod δ =0 then
            Randomly extract K mini-batch samples from the memory D;
            Train the network by loss function J(θ);
            Update θ⁻ by θ⁻ = θ in every G steps;
        end if
        Let B(t+1) = B(t) − e(t) and t = t + 1;
    end while
end for
```

IV. Performance Evaluation

In this section, the extensive numerical simulations are presented to verify the proposed deep reinforcement learning based path planning scheme for UAV-assist edge networks.

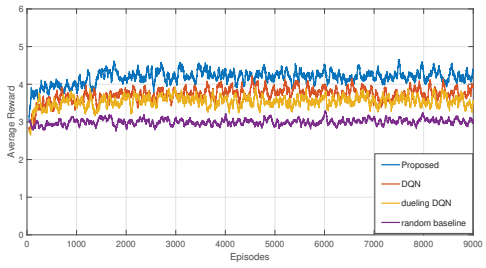

Fig. 3. Average rewards of different algorithms with N=60 $\bar{v}$=2m/s H=30m

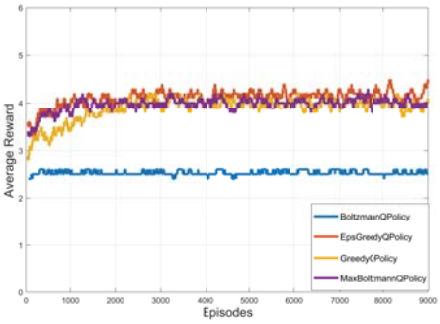

Fig. 4. Average reward for different selection strategies of N=60 $\bar{v}$=2m/s H=30m.

In order to reduce the computational complexity, we set a small number of fixed access points $M = 9$, that is, the number of action space is 9. Unless otherwise specified, we set the initial energy of the UAV $B_0 = 3 * 10^6$ Joule, the flying speed $V = 20m/s$, the flying power $P_f = 60$ watt,$P_0 = 80$ watt, $C = 1000$,$f_c = 2GHz$,$N_i = 1$Mb,$\gamma_c = 10^{-25}$F,$\beta = \frac{\pi}{6}$ and $\mu_i(t)$ is uniformly distributed within $[0, 10]$. To implement the DDQN algorithm, a four-layer fully connected neural network with two hidden layers is used, where the number of neurons in the two hidden layers is set to 400 and 300, respectively. The neural network adopts Relu, in which $f(x) = \max(0, x)$ as the activation function of all hidden layers, and the optimizer is used to update the network parameters at a learning rate of 0.0001. In addition, in order to better explore the environment setting $\varepsilon = 0.1$, the experience playback buffer size is set to $500,000$. It is supposed that the UAV start to serve the mobile devices with a certain initial energy, and then explore better service path strategies in a complex and changing environment, increasing the reward of the system until the energy consumption is over. This period is called as a complete episode. In the training phase, the number of episodes is $L_{\max} = 9000$, and the maximum number of steps in each episodes is $T_{max} = 200$.

Firstly, the achieved averaged reward(includes the offloaded data bits and energy consumption) of the intelligent path planning scheme under the proposed DDQN-based algorithm,DQN,Dueling DQN and the random baseline algorithm is presented in Fig. 3. In Section III, the path planning scheme is based on the DDQN algorithm. We also show the simulation

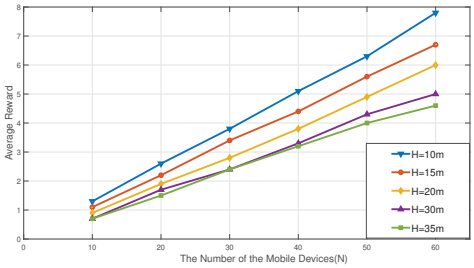

Fig. 5. Impact of UAV height H and the number of users N on the average reward with $\bar{v}$=2m/s.

results of the modified schemes based on the DQN and Dueling DQN algorithms. Among these algorithms, Dueling DQN is a variant of DQN, which decomposes the Q value into state value and advantage functions to obtain more useful information, and the random baseline algorithm is to randomly select a fixed access point for the UAV in each time slot to perform the corresponding task offload. We can observe that after 4000 episodes, all the path planning algorithms can reach the state of convergence, and the performance of proposed DDQN-based path planning algorithm is best, and that of the random baseline algorithm is worst. This result validates the superiority of the proposed path planning algorithm. Given the Q values, the strategy how to select the action can significantly affect the performance. In Fig. 4, we compare the performance of four action selection strategies. EpsGreedyQPolicy is a greedy strategy of $\varepsilon = 0.1$. GreeyQPolicy is to greedily choose the maximum Q value all the time. BoltzmannQPolicy is a Boltzmann Q strategy, which establishes a probability law according to Q value and returns a random selection action according to this law. MaxBoltzmannQPolicy is a combination of greedy strategy and Boltzmann Q strategy. As is shown, the performance of EpsGreedyQPolicy and GreeyQPolicy strategies are very close. While, the EpsGreedyQPolicy performs slightly better than GreeyQPolicy. This is because the EpsGreedyQPolicy is more appropriate to maintain a certain proportion of exploration in an unknown environment. Among the four strategies, the Boltzmann Q strategy is the worst.

In the UAV-assisted edge computing networks, the height of the UAV can greatly affect the coverage area, and the number of mobile device may also result a different performance.We secondly present the effect of both factors on the achieved performance in Fig. 5.The result is the average from 10 numerical simulations. As is shown, a larger UAV height leads to achieve a smaller average reward. This is mainly because a higher height of the UAV usually leads to a greater communication distance and smaller coverage area, such that the amount of data bits executed at the UAV becomes smaller. Meanwhile, we find that the average reward increases with respect to the number of mobile devices. Because when increasing the number of mobile devices,the UAV needs to consume more energy to complete the computing tasks of devices, but the weight of energy consumption is less than the weight of data

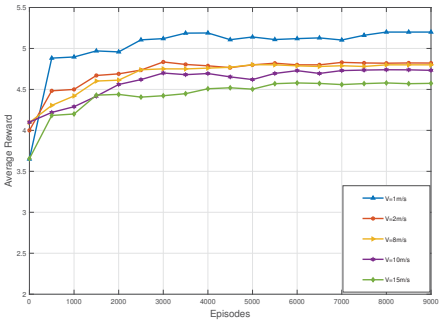

Fig. 6. Impact of different user speeds on the average reward when N=60 H=30m.

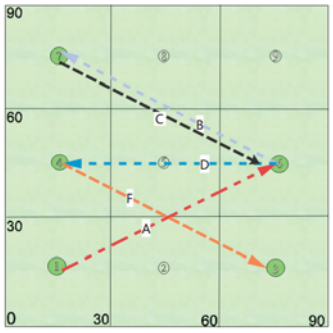

Fig. 7. An example serving path of the UAV.

bits offload, which makes the overall reward rise.

In Fig. 6, the effect of the mobility speed of IoT devices is presented, where the height of the UAV is set to 30m, and the number of devices is 60. In the cases of $\bar{v}$=1, 2, 8, 12 and $15m/s$, the proposed algorithm can converge, and a smaller average speed $\bar{v}$ leads to a larger reward.Because when the mobility speed of devices is small, the path planning problem gradually comes down to an optimization problem serving static devices, and the optimal strategy can be obtained rapidly and easily.

Finally, we show an example of path trajectory of the UAV in Fig. 7, where a area of 90m*90m is considered and $\bar{v}$=1m/s,H=$15\sqrt{3}$m,N=30. In the figure, nine access points labeled with serial numbers 1 to 9 are fixed. After the training phase of 6000 episodes, we can observe that in a complete episodes, the UAV serves the mobile devices through the route of 1→6→7→6→4→3.

V. Conclusion

In this paper, we have studied the intelligent path planning problem for a UAV-assisted edge computing network, where the mobility of IoT devices was taken into account. Specially, we have proposed a deep reinforcement learning based path planning algorithm that aimed to not only maximize the amount of data bits offloaded and executed at the UAV, but also minimize the energy consumption of the UAV for both the flying and computing operations. Extensive simulation results have validated the effectiveness of the proposed algorithm. In the future work, we plan to develop the multi-agent path planning and resource allocation scheme for multiple UAV-assisted edge computing networks, where the UAVs can collaborate to execute the computing tasks of mobile devices.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (NSFC) under grant 61901180, the Natural Science Foundation of Guangdong Province under grant No. 2019A1515011940, the Science and Technology Program of Guangzhou under grant No.202002030353, 2019050001, the Science and Technology Planning Project of Guangdong Province under grant No. 2017B030308009, 2017KZ010101, and also supported by Special Project for Youth Top-notch Scholars of Guangdong Province under grant No. 2016TQ03X100.

References

[1] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

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

[3] Y. Du, K. Yang, K. Wang, G. Zhang, Y. Zhao, and D. Chen, "Joint resources and workflow scheduling in uav-enabled wirelessly-powered mec for iot systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 10 187–10 200, 2019.

[4] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2018.

[5] R. S. Sutton and A. G. Barto et al., *Reinforcement learning: An introduction*. MIT press, 1998.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, and G. Ostrovski et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[7] J. Li, H. Gao, T. Lv, and Y. Lu, "Deep reinforcement learning based computation offloading and resource allocation for mec," in *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 2018, pp. 1–6.

[8] L. Huang, S. Bi, and Y. J. A. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," *IEEE Transactions on Mobile Computing*, vol. 19, no. 11, pp. 2581–2593, 2020.

[9] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for iot devices with energy harvesting," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1930–1941, 2019.

[10] S. Batabyal and P. Bhaumik, "Mobility models, traces and impact of mobility on opportunistic routing algorithms: A survey," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1679–1707, 2015.

[11] Q. Liu, L. Shi, L. Sun, J. Li, M. Ding, and F. Shu, "Path planning for uav-mounted mobile edge computing with deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5723–5728, 2020.

[12] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in uav-enabled mobile edge computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4576–4589, 2019.

[13] Q. Wu, Y. Zeng, and R. Zhang, "Joint trajectory and communication design for multi-uav enabled wireless networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 2109–2121, 2018.

[14] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *30th AAAI Conference on Artificial Intelligence*, pp. 2094–2100, 2016.