Air-Ground Edge Task Offloading Based on Multi-UAV Path Optimization and Resource Allocation

Chenguang He, Jing Li(✉), Shouming Wei, and Mengrui Guo

School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin, China
lijing1123lv@163.com

Abstract. In recent years, space-air-ground integrated network(SAGIN) has been recognized as a promising area in 6G research. In the air-ground component of SAGIN, airborne device such as unmanned aerial vehicles (UAVs) can provide task offloading and computing services to ground devices. Edge servers are installed on UAVs to provide data offloading services for ground devices. The high mobility of UAVs, compared to base stations and edge computing devices on the ground, makes it better able to provide timely and effective services to the devices. Considering the needs of delay-sensitive devices, this paper jointly allocates computational resources and designs UAV trajectories to achieve the goal of minimizing delay. In this paper, we use non-orthogonal multiple access (NOMA) technique in the uplink channel, which allows a UAV to serve multiple ground devices simultaneously. Both the UAV and the ground devices are moving, and the ground devices need to re-establish their connection to the UAV every once in a while. Based on the traditional Deep Reinforcement Learning (DRL) algorithm, this study proposes the Multi-Agent DRL (MADRL) algorithm to jointly determine the optimal 3D trajectory and computational resource allocation of UAVs.The MADRL algorithm achieves complete ground cooperation among multiple UAVs as agents in optimizing the latency by co-training the neural network, simplifying the network structure, and improving the training efficiency. Numerical results show that the proposed MADRL algorithm can converge under the system quality of service (QoS) constraints, and the convergence speed is faster than that of the traditional deep Q network (DQN) algorithm. The average total delay of the system can also be effectively reduced and converged in a multi-UAV scenario.

Keywords: Edge Computing · Deep Reinforcement Learning · Task Offloading

1 Introduction

As 5G technology continues to develop, the number of latency-intensive and computing-intensive applications continues to increase, and the communication system's demand for computing and bandwidth resources continues to increase. Therefore, mobile edge computing (MEC) is proposed and considered to be able to effectively meet latency

H.-H. Chen and W. Meng (Eds.): WiSATS 2024, LNICST 606, pp. 189–201, 2025.
https://doi.org/10.1007/978-3-031-86203-8_16

requirements and reduce the burden on users by providing additional resources. In MEC systems, users can offload their tasks to nearby edge servers, thereby reducing latency and energy consumption. However, common ground edge computing devices cannot meet the computing needs of users in complex scenarios, such as remote mountainous areas and disaster zones. The air-space-ground integrated system has been widely studied due to its advantages such as flexible deployment and remote connection to cloud servers. Considering the long distance between low orbit satellites and the ground, user mobility is negligible compared to low earth orbit (LEO). For UAVs deployed in the air, the movement of ground device has a great impact on the line of sight and channel strength between the air and the ground. Since UAVs and ground equipments are mobile, the distance between them is constantly changing with the change of position, resulting in different channel states. NOMA technology can be well adapted to communication systems with different channel states.

The trajectory design of the UAV in this paper is an NP-hard problem, and many conventional algorithms to solve this problem have the disadvantage of high complexity. In the UAV edge offloading scenario, the movement trajectory of the UAV is usually fixed to a circular trajectory, etc., but this situation is not consistent with the actual situation [3]. Moreover, since UAV flight is a continuous and long term process, how to improve the overall communication quality is a difficult problem. Markov decision processes (MDP) and DRL algorithm can be used to design UAV flight trajectory to maximize long term return. This paper combines the RL algorithm with the deep neural network and proposes a DRL algorithm to design the UAV's motion trajectory and computing resource allocation strategy.

2 System Model

As shown in Fig. 1, it is an integrated air, space and ground system, including a large number of mobile ground device and a small number of ground base stations, UAVs equipped with edge service areas, and LEO satellites. Considering that the user capacity of the base station is limited and conflicts may occur when multiple users access, some of the ground device chooses to connect to UAVs or low orbit satellites for data offloading. The main content of this work is the problem of data offloading from ground device to UAVs. In this scenario, the UAV is equipped with an edge server and uses NOMA technology. Ground devices are grouped according to their spatial location, with two devices in each group associated with a UAV. Ground devices in a group share the frequency band and will be subject to intra-group interference. At the UAV end, Successive Interference Cancellation (SIC) technology is used to eliminate interference to obtain information about each user. Multiple UAVs use the same frequency band and need to keep a certain distance from each other to avoid interval interference. UAVs, ground base stations, and LEO satellites all work in different frequency bands.

Denote the UAV as $u \in \mathbb{U} = \{1, 2, \ldots, U\}$, and the set of devices served by the UAV as $m \in \mathbb{M} = \{1, 2, \ldots, M\}$, after the device is grouped, each group is expressed as $s \in \mathbb{S} = \{1, 2, \ldots, S\}$. Each device can only be connected to one UAV, and a UAV can connect to two devices at the same time. The service period of UAV is $T = Kt_{\text{slot}}$, and each user is set to operate at the maximum speed $V_{\max}$ within the time slot t_{slot} moves

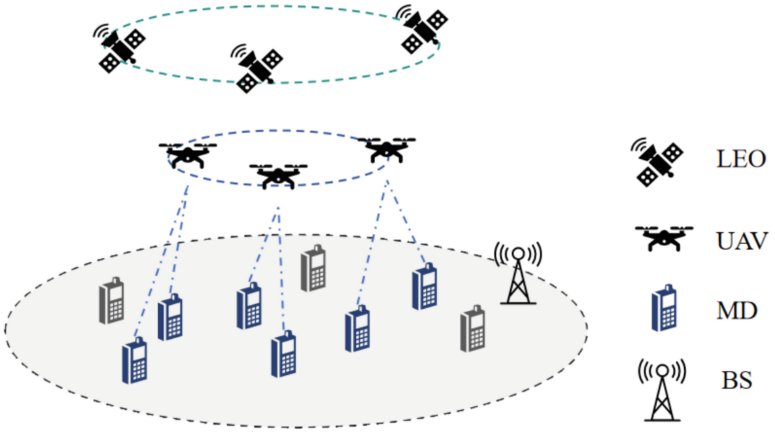

Fig. 1. System Model

in random directions on the horizontal plane. Considering that the devices are mobile, all devices need to be regrouped at the beginning of each time slot and the connection relationship with the UAV needs to be redetermined to achieve higher data rates.

2.1 Channel Model

The propagation channel from the UAV server to the ground device is modeled with probabilistic LoS and NLoS links. The probability of a LoS link between UAV u and ground user m is:

$$\mathrm{P_{LoS}}(t) = \frac{1}{1 + \delta_1 exp(-\delta_2(\theta_{u,m}(t) - \delta_1))} \tag{1}$$

$$\theta_{u,m}(t) = \frac{180}{\pi \arcsin(h_u / d_{u,m}(t))} \tag{2}$$

where δ_1 and δ_2 are environment-related constants, and $\theta_{u,m}$ indicates the elevation angle. $d_{u,m}(t) = \sqrt{h_u^2 + (x_u(t) - x_m(t))^2 + (y_u(t) - y_m(t))^2}$ represents the distance between the UAV and the ground device. Then the probability of NLoS is $1 - P_{\mathrm{LoS}}(t)$. The average path loss of LoS and NLoS links is written as:

$$\begin{aligned} L_{u,m}^{LoS}(t) &= 20\log(\frac{4\pi f_c d_{u,m}(t)}{c}) + \eta_{LoS}, \\ L_{u,m}^{NLoS}(t) &= 20\log(\frac{4\pi f_c d_{u,m}(t)}{c}) + \eta_{NLoS}, \end{aligned} \tag{3}$$

Therefore, the average path loss between UAV and ground device can be calculated by Eq. (4):

$$L_{u,m}(t) = P_{LoS}(t) \cdot L_{u,m}^{LoS}(t) + P_{NLoS}(t) \cdot L_{u,m}^{NLoS}(t) \tag{4}$$

In the case of small scale fading in the channel, the channel gain from the UAV to the device in t is:

$$g_{u,m}(t) = G_{u,m} \cdot 10^{-L_{u,m}(t)/10} \tag{5}$$

$G_{u,m}$ represents the fading coefficient between UAV and device.

2.2 Signal Model

UAV u provides data offloading services for ground device m_1, m_2 in group s. Let $\alpha_{u,m} = 1$ mean that UAV u and device m maintain Connected, $\alpha_{u,m} = 0$ means not connected, the transmit signal of device m is:

$$x_m(t) = \sum_{u=1}^{U} \alpha_{u,m}(t)\sqrt{P_{u,m}(t)}x_{u,m}(t) \tag{6}$$

where $\sqrt{P_{u,m}(\mathrm{t})}$ is the transmit power allocated to UAV u by device m, and $x_{u,m}(t)$ is device m transmit signal sent to UAV u. The received signal at UAV u is:

$$y_u(t) = g_{u,m_1}(t)x_{u,m_1}(t) + g_{u,m_2}(t)x_{u,m_2}(t) + I_e(t) + \sigma(t) \tag{7}$$

where $\sigma(t)$ is additive Gaussian white noise, $I_e(t)$ is the sum of signals of devices associated with other UAVs, which belongs to inter-group interference and can be expressed as:

$$I_e(t) = \sum_{k=1,k\neq m_1,m_2}^{M} g_{u,k}(t)\sqrt{P_k(t)}x_k(t) \tag{8}$$

where m_1, m_2 represents the ground device connected to UAV u. The UAV implements the decoding of devices m_1, m_2 through serial interference cancellation technology, and the order of decoding is determined according to the channel gain of different devices. Assume that the channel gain of device m_1 to UAV u is large, and the channel gain of device m_2 is small. Considering that the signal of device m_1 is strong and the signal of m_2 is weak, the signal of device m_1 is decoded first, and the signal of device m_2 is used as intra-group interference; then the signal of device m_2 is decoded, assuming B is the sub-channel bandwidth, and the data rate of devices m_1, m_2 can be obtained as:

$$R_{u,m_1} = B\log_2\left(1 + \frac{\left(\alpha_{u,m_1}(t)g_{u,m_1}(t)\sqrt{P_{u,m_1}(t)}\right)^2}{\left(\alpha_{u,m_2}(t)g_{u,m_2}(t)\sqrt{P_{u,m_1}(t)}\right)^2 + \sum\limits_{k=1,k\neq m_1,m_2}^{M}\left(g_{u,k}(t)\sqrt{P_k(t)}\right)^2 + \sigma(t)^2}\right) \tag{9}$$

$$R_{u,m_2} = B\log_2\left(1 + \frac{\left(\alpha_{u,m_2}(t)g_{u,m_2}(t)\sqrt{P_{u,m_2}(t)}\right)^2}{\sum\limits_{k=1,k\neq m_1,m_2}^{M}\left(g_{u,k}(t)\sqrt{P_k(t)}\right)^2 + \sigma(t)^2}\right) \tag{10}$$

2.3 Computing Model

UAV u provides data computing services for devices m_1, m_2. The tasks generated by devices m_1, m_2 will be offloaded to the server of UAV u in time slot t. The delay of task execution includes three parts: the transmission delay of uploading the task, the delay of computing the task at the UAV server u and the delay of offloading the execution results. Generally, the downlink transmission rate is higher, so the downlink transmission delay is ignored. Assume $D_m(t)$ represents the task data size of device m, then the transmission delay of offloading to UAV u is expressed as:

$$\tau_{u,m}^{tran}(t) = D_m(t)/R_{u,m}(t) \tag{11}$$

Assume that the UAV server time slot t allocates the computing resources of the device m to $f_{u, m}(t)$, and the computing delay is:

$$\tau_{u,m}^{comp}(t) = D_m(t)/f_{u,m}(t) \tag{12}$$

Therefore, the total time delay can be obtained as:

$$\tau_{u,m}(t) = \tau_{u,m}^{tran}(t) + \tau_{u,m}^{comp}(t) \tag{13}$$

3 Problem Formulation

The goal of this work is to optimize system performance while minimizing the average delay. The optimization variables include UAV server-user association $\mathbf{A} = \left\{\alpha_{u,m}(t), u \in \mathbb{U}, m \in \mathbb{M}\right\}$, UAV computing resource allocation $\mathbf{F} = \left\{f_{u,m}(t), u \in \mathbb{U}, m \in \mathbb{M}\right\}$, and position $\mathbf{L} = \{(x_u(t), y_u(t), z_u(t)), u \in \mathbb{U}\}$. The optimization problem can be formulated as:

$$\min_{\mathbf{F},\mathbf{L},\mathbf{A}} T_{sum} = \frac{1}{T}\sum_{t=1}^{T}\sum_{u=1}^{U}\sum_{m=1}^{M}\tau_{u,m}(t) \tag{14a}$$

$$s.t.\ \alpha_{u,m}(t) \in \{0, 1\},\ \forall u \in \mathbb{U}, m \in \mathbb{M}, \tag{14b}$$

$$\sum_{u=1}^{U}\alpha_{u,m}(t) = 1, \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{14c}$$

$$L_i(t) \neq L_j(t), \forall i, j \in \mathbb{U} \tag{14d}$$

$$R_{u,m}(t) \geq R_{QoS}, \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{14e}$$

$$\tau_{u,m}(t) \leq \tau_{QoS}, \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{14f}$$

$$\sum_{u=1}^{U}\alpha_{u,m}(t)f_{u,m}(t) \leq f_u(t), \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{14g}$$

Among them (14b) and (14c) indicate that the ground device is only connected to one UAV at time slot t; (14d) indicates that several UAVs cannot collide with each other; (14e) indicates that the transmission rate of each device needs to satisfy QoS rate limit R_{OoS} ; (14f) means that the total delay corresponding to each device needs to meet the QoS delay limit τ_{OoS} ; (14g) means that UAV is The computing resources provided by ground device cannot exceed its own computing resource limit $f_u(t)$.

Considering the movement of UAVs and ground device, the target problem is highly dynamic, making it challenging to solve the problem with traditional convex optimization algorithms.

4 Proposed Solutions

4.1 MDP

The purpose of this work is to minimize the average total task processing delay of the devices, so the optimization problem in the previous section is reformulated as a markov decision process (MDP), defined as (S, A,P,R, γ). Among them, S is the state set, A is the action set, P is the state transition probability, R is the reward function, and γ is the discount factor.

1) **Agent**: Each UAV is an agent.
2) **State space** S: The environment is described through the state space. The agent makes decisions based on S and determines the next action A. In this work, the mission data volume of the ground device $D_m(t)$, the UAV position L_u, and the channel status information from the ground device to the UAV form the state space.
3) **Action space** A: According to the current state and decision making strategy, the k th agent takes an $a_k \in$ A action. In this work, the computing resource allocation of UAV flight actions A_u and UAV constitutes the action space.
4) **Reward function** R: The environment gives feedback R based on the action of the agent, and due to the action, the state transitions from R to S′. The reward function in this work is related to the average total delay of the system.

The reward function needs to satisfy the location and resource constraints in the optimization problem while minimizing the average processing delay. If the QoS conditions of the device and UAV are violated, the corresponding agent will be punished by the environment. The reward function is set to:

$$R'_u(t) = \begin{cases} \frac{\tau_{QoS}}{T_{sum}(t)}, & \tau_{u,m,worst}(t) < \tau_{QoS} \\ \frac{\tau_{QoS}}{2^{\kappa} \cdot T_{sum}(t)}, & \tau_{u,m,worst}(t) \geq \tau_{QoS} \end{cases}, \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{15}$$

$$\text{if } R_{u,m}(t) < R_{QoS},\ R_u(t) = R'_u(t) - \lambda,\ \forall u \in \mathbb{U}, m \in \mathbb{M} \tag{16}$$

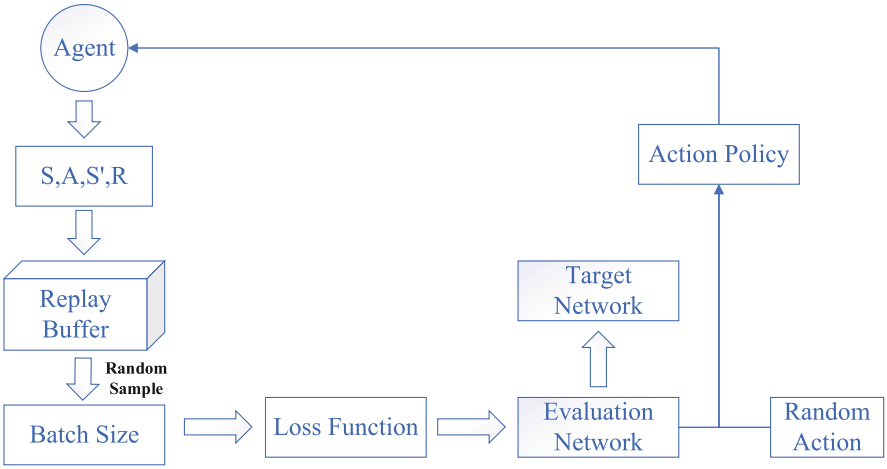

Fig. 2. Flow chart of the MADRL algorithm based on DQN

where κ is the penalty coefficient, and R(t) is the data rate. Since multiple UAVs are fully cooperative, in order to minimize the system delay, each agent's reward must be determined by the total delay rather than the delay of a single UAV. The penalty factor allows the system to guarantee that the QoS of all ground equipment is met.

4.2 UAV-Ground Device Association

Considering that the UAV device provides computing offloading services for the ground device device group, in order to minimize the delay, the connection between the UAV and its nearest device group should be ensured. First of all, in order to minimize the distance between the UAV and the device, the distance between the device in the group should be relatively close, while the ground device moves randomly, so after a period of time, the grouping of the device should be reconsidered.

This work uses the bisecting kmeans clustering algorithm to group ground device. Compared with the traditional kmeans algorithm, the bisecting kmeans algorithm is not sensitive to the initial center point of the cluster, has lower algorithm complexity and good clustering effect. This work uses the bisecting kmeans algorithm to minimize the sum of mean square errors (SSE) as the goal, and divides the ground device into U groups. The distance from the inner center to the UAV determines the connection relationship between the UAV and the device group. The specific process can be seen in Algorithm 1.

4.3 DRL-Based MADRL Algorithm

Since there are multiple agents in the system and information cannot be exchanged between each agent, it is very difficult to train a separate neural network and minimize the average delay. Based on the deep Q-network (DQN), this work designs the MADRL algorithm to jointly optimize the movement trajectory and computing resource allocation of the UAV. Its schematic diagram is shown in Fig. 2. Each UAV is an agent that shares the same state space and trains the same neural network with other agents without knowing the actions of other agents.

The DQN algorithm is a value-based reinforcement learning method. Its purpose is to maximize the long term reward of the agent. The agent will take actions according to the dynamic environment, so the action value (Q value) is used to evaluate each action. In order to maximize the long term reward, the DQN algorithm will select the action with the largest Q value.

$$Q(S,A) \leftarrow Q(S,A) + \ell\left[R + \delta \max Q\left(S', A'\right) - Q(S,A)\right] \tag{17}$$

where $\ell(0 < \ell < 1)$ represents learning rate, and $\gamma(0 < \gamma < 1)$ is discount factor.

In each episode, the agent will identify the current state S, and after inputting the state information into the evaluation network, the output actions of the evaluation network and the random actions together form an action strategy, and then the agent selects and implements actions according to the action strategy. The selection method of action strategy adopts greedy strategy, as shown in the following formula.

$$A = \begin{cases} \text{random action}, & \varepsilon, \\ \text{argmax}_A Q(S,A,\omega), & 1-\varepsilon. \end{cases} \tag{18}$$

where ε represents the probability of the agent choosing a random action. Considering that the network is more reliable in the later stages of training, it gradually decreases as the training progress. Where ω represents the neural network parameters. Actions change the UAV trajectory and allocate computing resources on the UAV.

The state S moves to the next state S' after the action is implemented, and the agent saves S, A, S', R to the replay buffer. A mini batch data set is randomly sampled from the replay buffer, and the data set is used to train the evaluation network through the loss function. In addition, when a nonlinear function is used to approximate the Q value function, the update of the Q value is prone to oscillation, showing unstable learning behavior. For this reason, the target network $Q\left(S', A'\right)$.

Algorithm 1: Binary Search

for each episode **do**
  Initialize positions of UAVs and devices
  Initialize evaluation network and target network with random parameter
  for each step t = 1, 2, …, T, **do**
  if t = $T_{recluster}$, **do**
    Form all samples in devices location set L_M into initial cluster C_0
    repeat
      Initialization minimum error $\mathrm{SSE}_{\min} = \inf$
      for i = 1, 2, …, $\left|\mathrm{L}_M\right|$, **do**
        Divide C_0 into two clusters using kmeans algrithm
        Calculate the total error SSE_{sum} after dividing the cluster
        if $\mathrm{SSE}_{sum} < \mathrm{SSE}_{\min}$ then
          Update minimum error $\mathrm{SSE}_{\min} = \mathrm{SSE}_{sum}$, $\lambda = i$
          Two new clusters after division: $C_{\lambda_a} = C_{i_a}, C_{\lambda_b} = C_{i_b}$
      end for
      Update the divided cluster as a cluster: $C_{\lambda_{split}} = C_{\lambda_a}$
      Add another cluster into the cluster set $C = C_{\lambda_b} \cup C$
    untill get k clusters
    Let the number of device in each group is the same
    Determine the distance of the UAV and the center position each of group, UAV is connected to the nearest group
  for each UAV, **do**
    Generate state array
    Choose action according to action policy
    Take action *A*, observe *R* and *S'*, store (*S, A, R, S'*)
    Sample training data from replay buffer
    Calculate target Y
    Train network parameter
    Update target network parameters regularly
    $S \leftarrow S'$
  end for
  Devices move
end for
end for

The parameters of the target network will be updated every N' steps. The Q value is temporarily fixed during the training process, thereby making the learning process more stable. The loss function is as follows:

$$Loss(\omega) = (Y - Q(S, A, \omega))^2 \tag{19}$$

where Y is the objective function, and the detailed algorithm flow is listed in Algorithm 1.

5 Simulation Result Analysis

In the simulation model of this work, users are randomly distributed in the service area at the initial moment, and the UAV is deployed in the air at a height of 110m. The multi-layer perceptron network used in the MDRL algorithm includes 3 layers of neural networks. The ReLu function is used as the activation function and the adam optimizer is used to train the neural network. The remaining parameters are in Table 1.

Table 1. Parameters value

Parameters	Value
Transmit power	0.3 W
Carrier frequency	2 GHz
Number of UAVs	1, 2, 3, 4
Bandwidth of device B	100 kHz
Maximum speed of device	1 m/s
UAV speed	6 m/s
Training steps in each episode	60, 240
Frequency allocation unit	1 MHz
Device data size following the poisson distribution λ	0.1 MB
τ_{QoS}	5 s
R_{QoS}	3 kb/s
Learning rate	0.001,0.002,0.005
Batch size	64
Greedy coefficient ε	0.8 - 0

Figure 3 shows the UAV trajectory obtained by the MADRL algorithm and the movement of the user. In order to make the trajectory planning more obvious, the ground device is moving at orientation, and an EPISODE step T = 240. It can be seen from the overall movement trend that the UAV first gradually increases the height, and as the direction of the device moves closer to the device, the height is adjusted in the process. It can be seen that the algorithm can effectively adapt to mobile device scenarios and maximize the advantages of UAV.

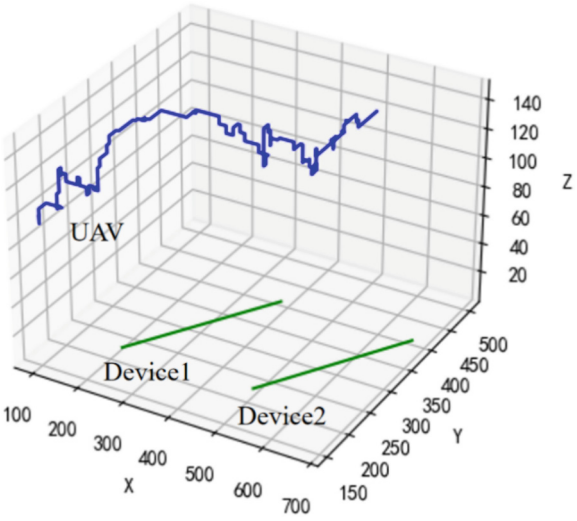

Fig. 3. Single UAV trajectory optimization.

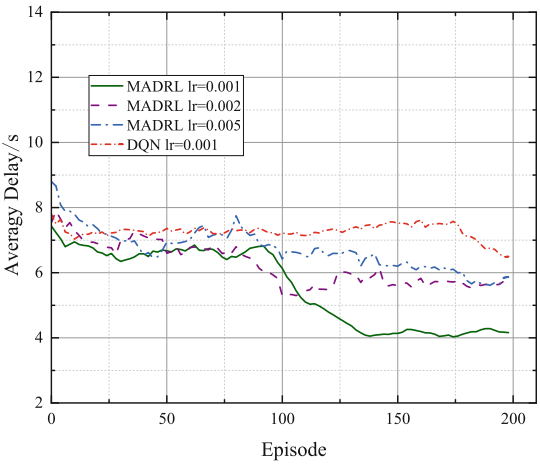

Fig. 4. Average delay vs learning rate

Figure 4 shows the average total device latency during training as a function of learning rate. Three UAVs are set up, the replay buffer size is set to 12000, and the step size is T = 60. By the 67th episode, the buffer is just full. Starting training from the 68th episode, it can be seen that when the learning rate is 0.001, the training effect is the best, and the average delay drops from 7 s to about 4 s. When the learning rate is 0.002 and 0.005, the training effect is poor, and the delay decrease is not obvious. The DQN algorithm with a learning rate of 0.001 only shows a downward trend after 170 episodes, and the training effect is poor. This is because the multi-agent system using the

DQN algorithm needs to train multiple neural networks independently, and redundancy in network parameters will occur.

Figure 5 compares the average total latency of multiple devices with different numbers of UAVs. It can be seen that when the number of users is 2, 4, 6, 8, the total delay increases as the number of users increases. And the average delay of a single device increases as the number of devices increases. This is because when the number of UAVs and ground device increases, the inter-cluster interference of the system increases, and when the number of agents increases, it becomes more difficult for the MADRL algorithm to obtain good convergence results. Therefore, in a single UAV scenario, the average processing delay of each ground device is almost unchanged and is the lowest, about 0.2 s. In the 4 UAV scenario, the average processing delay of each device is about 1.2 s.

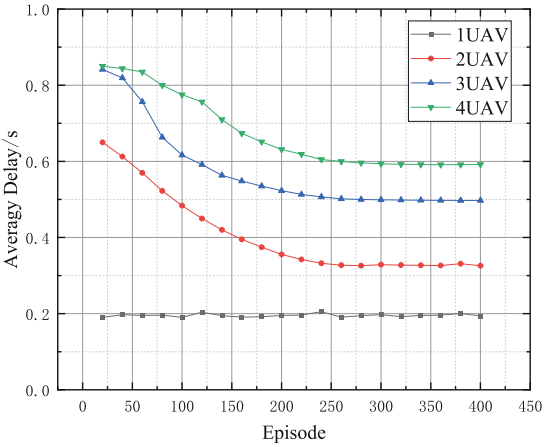

Fig. 5. Average total delay under different UAV numbers

6 Conclusion

This work proposes the MADRL algorithm to jointly determine the optimal three dimensional trajectory and computing resource allocation of UAVs. The MADRL algorithm implements multi-agent collaboration and minimizes the time delay and simplifies the network structure by jointly training the neural network. Numerical results show that the proposed MADRL algorithm can converge under system QoS constraints, and the convergence speed is faster than the traditional DQN algorithm. In multi-UAV scenarios, the average total delay of the system can also be effectively reduced and converged.

Acknowledgement. This work is supported by the Key R&D Program of Heilongjiang Province under Grant JD22A001.

References

1. Zhou, F., Wu, Y., Hu, R.Q., Wang, Y., Wong, K.K.: Energy-efficient NOMA enabled heterogeneous cloud radio access networks. IEEE Netw. **32**(2), 152–160 (2018)
2. Song, Q., Zheng, F.-C., Jin, S.: Multiple UAVs enabled data offloading for cellular hotspots. In: Proceedings of the IEEE Wireless Communication Network Conference (WCNC), April 2019, pp. 1–6 (2019)
3. Lyu, J., Zeng, Y., Zhang, R.: Spectrum sharing and cyclical multiple access in UAV-aided cellular offloading. In: Proceedings of the IEEE Global Communication Conference (GLOBECOM), December 2017, pp. 1–6 (2017)
4. Zhang, H., Du, J., Jiang, C., Fakhreddine, A., Alhammadi, A., Wang, J.: MATD3-based joint user association and resource allocation in UAV networks. In: GLOBECOM 2023 - 2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, pp. 6892–6897 (2023)
5. Hu, J., Zhang, H., Song, L., Schober, R., Poor, H.V.: Cooperative internet of UAVs: distributed trajectory design by multi-agent deep reinforcement learning. IEEE Trans. Commun. **68**(11), 6807–6821 (2020)
6. Yang, Z., Bi, S., Zhang, Y.-J.A.: Dynamic offloading and trajectory control for UAV-enabled mobile edge computing system with energy harvesting devices. IEEE Trans. Wireless Commun. **21**(12), 10 515–10 528 (2022)
7. Zhao, N., Ye, Z., Pei, Y., Liang, Y.-C., Niyato, D.: Multi-agent deep reinforcement learning for task offloading in UAV-assisted mobile edge computing. IEEE Trans. Wireless Commun. **21**(9), 6949–6960 (2022)
8. Vishnoi, V., Budhiraja, I., Gupta, S., Kumar, N.: A deep reinforcement learning scheme for sum rate and fairness maximization among d2d pairs underlaying cellular network with NOMA. IEEE Trans. Veh. Technol. **72**, 1–17 (2023)
9. Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning, arXiv preprint arXiv:1509.02971 (2015)