Efficient UAV/Satellite-assisted IoT Task Offloading: A Multi-agent Reinforcement Learning Solution

Kangjia Yu[1], Qimei Cui[1], Ziyuan Zhang[1], Xueqing Huang[2], Xuefei Zhang[1], Xiaofeng Tao[1]
[1]National Engineering Lab for Mobile Network Technologies, Beijing University of Posts and Telecommunications, 100876
[2]New York Institute of Technology, Old Westbury, NY, 11568
{yukangjia, cuiqimei, zzy1022, zhangxuefei, taoxf}@bupt.edu.cn, xhuang25@nyit.edu

***Abstract*—In the future mobile edge networks, the Internet of things (IoT) applications will be latency-sensitive and computationally intensive. Given the resource limitation of IoT devices, mobile edge computing (MEC) servers are critical to support the efficient processing of IoT tasks. Since MEC servers attached to the ground base stations are generally deployed in fixed locations and vulnerable to physical damage, the unmanned aerial vehicle (UAV) and satellite-assisted MEC framework has been proposed to leverage the flexibility of UAVs and the broad coverage of satellites. However, efficient utilization of the UAV/satellite resources is challenging for the static ground IoT devices because of the dynamic in terms of aerial and space network topology and IoT task arrival rates. To adapt to the changing environment and utilize the interaction among multiple UAVs, we propose a multi-agent deep deterministic policy gradient (MADDPG) framework to jointly optimize the traveling routes of multi-UAVs and the offloading decision of IoT devices. To minimize the processing cost in terms of task processing latency and energy consumption of IoT devices, cooperative UAVs can help find the optimal task offloading location for each IoT device. Simulation results show the proposed algorithm based on MADDPG can averagely decrease 20% of the above processing cost compared with the benchmark approach.**

***Keywords*—UAVs, Satellite, IoT devices, MEC, Task offloading, MADDPG**

I. Introduction

With the widespread of IoT applications, mobile data traffic will continue to grow and the generated IoT tasks will be latency-sensitive and computationally intensive. With the advantage of distributed computing and low latency, mobile edge computing (MEC) has been proposed to enhance IoT devices with limited resources [1]. However, MEC servers deployed on the fixed ground station cannot flexibly change locations according to the needs of mobile users. In addition, the edge servers will be dysfunctional after partial or complete infrastructure damage caused by natural disasters.

The space-air-ground integrated network (SAGIN) assisted by satellite and unmanned aerial vehicles (UAVs) emerges as a promising architecture to provide a wide range of coverage and computing offloading services for remote IoT devices. UAVs and satellites can well relieve the traffic pressure brought by the surge and uneven distribution of mobile data and make up for the shortage of ground stationary networks. The deployment of UAVs and satellites brings several challenges, including trajectory design, power allocation, and energy efficiency. There exist plenty of mathematical optimization-based works that try to solve these problems for SAGIN. For instance, to minimize the energy consumption of the UAV-enhanced MEC system, a low complexity fuzzy c-means clustering algorithm was proposed to jointly schedule the user association, power control, computing capacity allocation, and location of UAVs [2]. Meanwhile, the number and location of UAVs have been optimized by the differential evolution algorithm with an elimination operator to reduce the dimensionality of the search space [3]. A spaceborne MEC network resource allocation strategy was designed to leverage low-orbit satellites as edge nodes and provision low-latency computing services for access terminals [4].

Since the location of UAVs and the computing resources demands of IoT devices change rapidly in the SAGIN-assisted MEC system environment, it is challenging to obtain global information and construct a mathematical optimization model. As a result, reinforcement learning (RL) has been adopted to adjust the location of MEC servers without requiring prior knowledge of the system [5]. For remote IoT devices deployed in the SAGIN edge/cloud framework, a deep reinforcement learning-based computing offloading method was used to minimize the weighted summation of server usage cost, energy consumption, and latency [6]. By considering the dynamic arrival of tasks to edge servers, a risk-sensitive reinforcement learning algorithm was designed to obtain a task scheduling policy that minimizes the task offloading and computational latency under the energy constraints of UAV [7]. However, the above works only consider the scenario of a single UAV and when training in a multiagent environment, the dynamic changes of other agents violate the Markov assumptions required for RL convergence.

For the system scenario with remote IoT devices assisted by multiple UAVs and satellites, we aim to minimize the system processing cost in terms of task processing latency and energy consumption of IoT devices. To adapt to the ever-changing environment and utilize the interaction among multiple UAVs, a multi-agent deep deterministic policy gradient (MADDPG) framework is proposed to jointly optimize the UAV locations and the offloading decision of IoT devices. With the proposed framework, cooperative UAVs can help find the optimal location to which the IoT devices can offload their computing tasks. As compared with the benchmark approach, simulation results show that on average, the proposed MADDPG-based

algorithm can decrease the processing cost by 20%.

II. System Model

For the remote area shown in Fig. 1, we consider a mobile edge computing system assisted by M UAVs and a satellite. Each UAV in the set $\mathcal{M} = \{1, \cdots, M\}$ carries a MEC server and hovers at a height of H above the square with l_{max} side length. In addition, a low-orbit in space satellite carrying a cloud server can cover the entire area. A set of $\mathcal{N} = \{1, \cdots, N\}$ IoT devices are randomly deployed in the area. For the discrete-time system with $\mathcal{T} = \{1, \cdots, T\}$ time slots, each IoT device will generate a computing task at each time slot t, $t \in \mathcal{T}$.

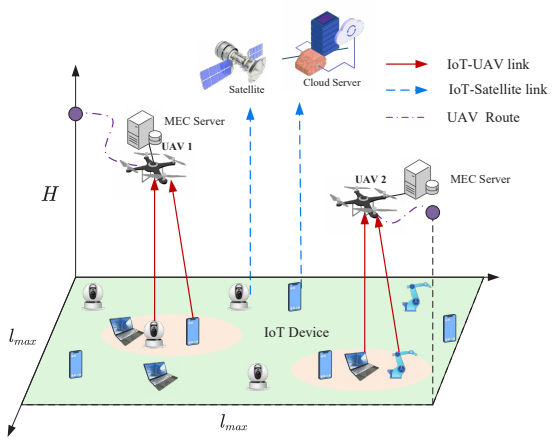

Fig. 1. UAV/Satellite-assisted computing tasks offloading framework.

The resource-limited IoT devices can offload their computing tasks to UAVs and satellite. For each IoT device, we define $\mathcal{M}' = \{0, \cdots, M+1\}$ as the set of potential locations where the computing tasks will be processed. If the location index $m \in \mathcal{M}'$ is equal to zero, there is no offloading, i.e., local processing. Meanwhile, $m = M+1$ indicates offloading to the satellite. Other values of m imply offloading to a UAV. Then, at the t-th time slot, the binary decision variable for IoT device n, $n \in \mathcal{N}$, can be defined as follows.

$$\alpha_{n,m,t} \in \{0,1\}, \forall n \in \mathcal{N}, m \in \mathcal{M}', t \in \mathcal{T} \tag{1}$$

where $\alpha_{n,m,t} = 1$, $m \in \mathcal{M}$, means the tasks generated by the n-th device will be processed at the m-th UAV. Moreover, we assume that each task can only be processed in one location.

$$\sum_{m \in \mathcal{M}'} \alpha_{n,m,t} = 1, \forall n \in \mathcal{N}, t \in \mathcal{T} \tag{2}$$

To find the optimal offloading decision such that the latency experienced by IoT devices and the energy consumed by IoT devices are minimized, the UAV traveling model, the IoT device-UAV/satellite communications model, and the data processing models are introduced next, which are followed by the formalized joint UAV traveling and IoT task offloading optimization problem.

A. UAV Traveling Model

During each time slot t, UAVs move at a fixed distance of 5 m towards the flight direction, which is defined by the angle $\beta_{m,t} \in [0, 2\pi)$. At the beginning of the t-th time slot, we denote $(X_{m,t}, Y_{m,t}, H)$ as the three-dimensional coordinates of the m-th UAV. The Euclidean distance between two UAVs will be $R_{m,m^*,t}$.

$$R_{m,m^*,t} = \sqrt{(X_{m,t} - X_{m^*,t})^2 + (Y_{m,t} - Y_{m^*,t})^2}, \tag{3}$$

where the inter-UAV distance has to be greater than R_{min} to avoid a collision.

$$R_{m,m^*,t} \geq R_{min}, \forall m, m^* \in \mathcal{M}, m \neq m^*. \tag{4}$$

Suppose the coordinate of the n-th IoT device is $(x_n, y_n, 0)$, then at time t, the horizontal distance between device n and UAV m is defined as:

$$R_{n,m,t} = \sqrt{(X_{m,t} - x_n)^2 + (Y_{m,t} - y_n)^2}. \tag{5}$$

If the computing task generated by IoT device n is to be offloaded to UAV m for processing, the IoT device must be within the service area covered by the UAV.

$$\alpha_{n,m,t} R_{n,m,t} \leq R_{max}, \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{6}$$

where R_{max} is the coverage radius of each UAV.

B. Data Transmission Model

For the communication link between the ground IoT device and the UAV, only large-scale fading is considered. At time slot t, the path loss between IoT device n and UAV m is:

$$L^H_{n,m,t} = 20\log\left(\frac{4\pi f_c \sqrt{H^2 + R_{n,m,t}{}^2}}{c}\right) + P_{LoS}\eta_{LoS} + (1 - P_{LoS})\eta_{NLoS}, \tag{7}$$

where η_{LoS} and η_{NLoS} represent the additional losses caused by the line-of-sight (LoS) link and non-line-of-sight (NLoS) on top of the free-space path loss, respectively. f_c in the numerator is the carrier frequency while c in the denominator is the speed of light. P_{LoS} is the probability of LoS transmission of the link, which is defined as follows.

$$P_{LoS} = \frac{1}{1 + a\exp\left(-b\left(\arctan\left(\frac{H}{R_{n,m,t}}\right) - a\right)\right)}, \tag{8}$$

where $(a, b, \eta_{LoS}, \eta_{NLoS})$ are constants depending on the environment. In remote areas, they are generally set as $(4.88, 0.43, 0.1, 21)$ [8].

Since multiple tasks from different IoT devices might be offloaded to the same UAV, to avoid transmission interference, we assume orthogonal frequency division multiple access (OFDMA) is adopted to support the IoT device-UAV communications [9]. The resulting uplink data transmission rate between IoT device n and UAV m is given below.

$$r_{n,m,t} = B\log_2\left(1 + \frac{P_{nU} \times 10^{-L^H_{n,m,t}/10}}{\sigma_o^2}\right), \tag{9}$$

where B is the transmission bandwidth, P_{nU} represents the transmit power from the IoT device to the UAV, σ_o^2 represents the power of the background noise.

For the IoT device to satellite communications link, we assume a constant data transfer rate r_{nS} [10], which is usually less than the data transfer rate of the IoT-UAV link $r_{n,m,t}$.

C. Computing Task Processing Model

For the above system, at time slot t, suppose the n-th IoT device generates a task $I_{n,t} = (C_{n,t}, D_{n,t})$, where $C_{n,t}$ refers to the number of CPU cycles required to calculate each bit (Cycles/bit), and $D_{n,t}$ indicates the size of the task's input data. As compared with the input data size, the size of the output of the task is assumed to be negligible.

There are three processing models for task $I_{n,t}$: 1) local processing ($\alpha_{n,0,t} = 1$); 2) offload to UAV for processing; 3) offload to satellite for processing ($\alpha_{n,M+1,t} = 1$). The latency and energy consumption in the three models are detailed below.

1) Local Processing Model: The task $I_{n,t}$ is to be processed locally on the n-th IoT device, and the time required to complete the computing task is defined as follows.

$$T_{n,0,t} = \frac{C_{n,t}D_{n,t}}{f_L}, \tag{10}$$

where f_L [in cycle/s] is the operating clock frequency of each IoT device.

The local energy consumption required to compute this task is defined as:

$$E_{n,0,t} = \eta_1 (f_L)^{v-1} C_{n,t} D_{n,t}, \tag{11}$$

where $\eta_1 = 10^{-27}$ is the effective switched capacitor, and $v = 3$ is a constant.

2) UAV Processing Model: The task $I_{n,t}$ is first transmitted to the UAV and then processed by the MEC server on the UAV. The time required by completing the task includes input data transmission time and processing time.

$$T_{n,m,t} = \frac{D_{n,t}}{r_{n,m,t}} + \frac{C_{n,t}D_{n,t}}{f_U}, \tag{12}$$

where the f_U is the operating clock frequency of each UAV and the task retrieval time is not considered because of the small output data.

The local energy consumed by this task is mainly for input data transmission:

$$E_{n,m,t} = P_{nU} \frac{D_{n,t}}{r_{n,m,t}}, \tag{13}$$

3) Satellite Processing Model: Owing to the long distance between IoT devices and the low-orbit satellite, the propagation delay t_w is not negligible. The time required by offloading the task to the cloud includes data transmission time, processing duration time and propagation delay, which is defined as:

$$T_{n,M+1,t} = \frac{D_{n,t}}{r_{nS}} + \frac{C_{n,t}D_{n,t}}{f_S} + 2t_w, \tag{14}$$

where r_{nS} is the data transmission rate between IoT device and satellite, and f_S is the operating clock frequency of the satellite. The propagation delay t_w is doubled because both the data transmission to the satellite and result retrieval from the satellite are considered.

The local energy consumed by this task is also mainly for input data transmission:

$$E_{n,M+1,t} = P_{nS} \frac{D_{n,t}}{r_{nS}}, \tag{15}$$

where P_{nS} represents the transmission power from the n-th IoT device to satellite.

D. Problem Formulation of Joint UAV Traveling and IoT Task Offloading

The goal of this paper is to minimize the processing cost of all tasks, which is the weighted sum of (1) the latency of all tasks and (2) the energy consumption of all local IoT devices. Then, the corresponding joint optimization problem of UAV traveling and IoT task offloading can be formulated as follows:

$$\min_{\{\alpha_{n,m,t}, \beta_{m,t}\}} \sum_{t\in\mathcal{T}} \sum_{m\in\mathcal{M}'} \sum_{n\in\mathcal{N}} \alpha_{n,m,t} (\omega_{\mathrm{T}} T_{n,m,t} + \omega_{\mathrm{E}} E_{n,m,t}) \tag{16}$$

$$\begin{aligned} s.t.\ & (1), (2), (4), (6) \\ & C1: 0 \le X_{m,t} \le l_{max}, \forall m \in \mathcal{M}, t \in \mathcal{T} \\ & C2: 0 \le Y_{m,t} \le l_{max}, \forall m \in \mathcal{M}, t \in \mathcal{T}, \end{aligned}$$

where w_T and w_E are the weighting factors of latency and energy consumption respectively which are determined according to the requirement of specific scenario. The newly introduced constraints C1-C2 limit UAVs to the target area; Note that although $\beta_{m,t}$ does not directly appear in the objective function or the constraints, it will determine the location of each UAV at each time slot, and hence impact the data rate and final task processing cost.

Since the optimization problem in Eq. (16) involves a mix of discrete and continuous decision variables: $\alpha_{n,m,t}$ is binary while $\beta_{m,t}$ is continuous, it is difficult to solve this NP-hard problem with mathematical optimization methods. Moreover, each UAV has to derive the optimal action in terms of setting the values for $\beta_{m,t}$, without prior knowledge of the locations and actions of other UAVs. Therefore, this paper introduces a distributed reinforcement learning-based algorithm named the multi-agent deep deterministic policy gradient (MADDPG) scheme to solve this problem [11].

III. Multi-agent Reinforcement Learning based Task Offloading Algorithm

For the proposed UAV/satellite-assisted computing task offloading framework, to simply the system model and obtain the optimal decisions ($\beta_{m,t}$ and $\alpha_{n,m,t}$), the UAV traveling route design and IoT task offloading are decoupled. In particular, a) each UAV m needs to first decide the flight direction; b) once

the locations of all the UAVs are determined, each IoT device n can compare all the possible offloading locations and choose the one with the minimum processing cost.

A. UAV Traveling Path Design

To solve the core problem of the proposed framework, i.e., trajectory design of UAVs, the MADDPG algorithm is adopted with its ability to solve complex optimization problems via the cooperation of UAVs. As compared with the traditional reinforcement learning algorithms, where each UAV agent makes decisions according to the changing environment, MADDPG also considers the influence of each agent's action on others.

At time slot t, each UAV agent will observe the environment, take an action, and receive a reward. The details of observation, action, and reward are defined as follows.

1) Observation $s_{m,t}$: The observation includes the coordinates of the UAV m, the location and the generated tasks of IoT devices. In addition, the pair-wise distance among UAVs is included as part of the observation to avoid collisions.

2) Action $a_{m,t}$: The action is defined as $a_{m,t} = \beta_{m,t}$, which represents the flight direction of UAV m at the t-th time slot.

3) Reward r_t: Considering time delay and energy consumption comprehensively, the reward function is defined as:

$$r_t = \frac{1}{\frac{1}{N}\sum\limits_{m\in\mathcal{M}'}\sum\limits_{n\in\mathcal{N}}\alpha_{n,m,t}(\omega_{\mathrm{T}}T_{n,m,t}+\omega_{\mathrm{E}}E_{n,m,t})} - p, \quad (17)$$

where p is the penalty that will be imposed if UAVs fly out of the target area or collide with other UAVs. The reward is inversely proportional to the total processing cost.

Note that since MADDPG guides every agent to choose an action that is beneficial to the whole system rather than itself, every UAV m shares the same system reward r_t in Eq. (17). Meanwhile, in a multi-agent learning system, the state transition of the environment depends on the joint behavior of the agents. Consequently, the overall system state is defined as $s_t = \{s_{m,t}, \forall m \in \mathcal{M}\}$ and an overall system behavior is $a_t = \{a_{m,t}, \forall m \in \mathcal{M}\}$.

To maximize the cumulative discounted reward $\sum_{t\in\mathcal{T}}\gamma^{t-1}r_t$, with γ being the discount factor, MADDPG will construct four networks for each agent: a critic network, an actor network, and their corresponding target networks in Fig. 2 [12].

1) Critic network $Q^m(s_t, a_t)$: the inputs of the critic network are a mini-batch of samples from the replay buffer, and the output is the estimated Q value of the current state and the corresponding action. The loss function of the critic network is calculated by

$$L(\theta^{Q^m}) = \mathbb{E}\left[\left(Q^m\left(s_t, a_t|\theta^{Q^m}\right) - r_t - \gamma Q^{m'}\left(s_{t+1}, a_{t+1}|\theta^{Q^{m'}}\right)^2\right)\right], \quad (18)$$

where θ^{Q^m} and $\theta^{Q^{m'}}$ represents the parameters of the Q^m and the target network $Q^{m'}$.

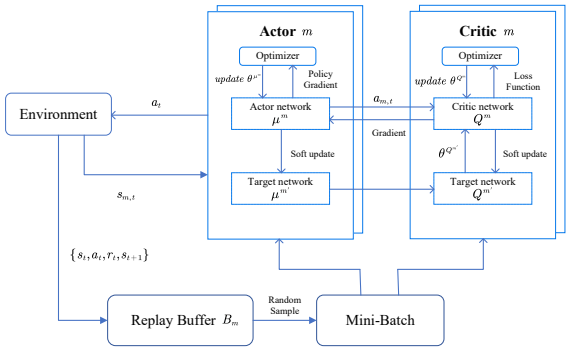

Fig. 2. The network structure of MADDPG.

2) Actor network $\mu^m(s_{m,t})$: At each time slot t, the input of actor network is the observation $s_{m,t}$, and the output is the action $a_{m,t}$ chosen by the deterministic policy μ^m. The gradient of the expected reward for UAV m is given below.

$$\nabla_{\theta^{\mu^m}} J = \mathbb{E}\left[\nabla_{\theta^{\mu^m}}\mu^m\left(s_{m,t}|\theta^{\mu^m}\right)\nabla_{a_{m,t}}Q^m\left(s_t, a_t|\theta^{Q^m}\right)\right], \quad (19)$$

where the θ^{μ^m} represents the parameters of policy μ^m.

3) The target network $a_{m,t+1} = \mu^m(s_{m,t+1})$ and $Q^m(s_{t+1}, a_{t+1})$: the parameters of the target networks are updated by

$$\begin{cases} \theta^{\mu^{m'}} \leftarrow \tau\theta^{\mu^m} + (1-\tau)\theta^{\mu^{m'}}, \\ \theta^{Q^{m'}} \leftarrow \tau\theta^{Q^m} + (1-\tau)\theta^{Q^{m'}}, \end{cases} \quad (20)$$

where τ is the learning rate.

4) The replay buffer B_m stores the agent's training experience. At each time slot t, the tuples $\{s_t, a_t, r_t, s_{t+1}\}$ are saved in the B_m so that in training mode, the critic network can be updated by sampling a random mini-batch.

B. IoT Task Offloading

For each IoT device n, the optimal offloading decision at time slot t is given below.

$$\alpha_{n,m,t} = \begin{cases} 1, m = \arg\min\limits_{m\in\mathcal{M}'}\{\omega_{\mathrm{T}}T_{n,m,t} + \omega_{\mathrm{E}}E_{n,m,t}\} \\ 0, \text{else}. \end{cases} \quad (21)$$

The details of the proposed MADDPG-based UAV traveling route and IoT task offloading optimization are illustrated in Algorithm 1.

IV. Simulation Results And Analysis

In this section, simulations are conducted to demonstrate the effectiveness of the proposed MADDPG algorithm in terms of reducing the processing latency and energy consumption.

A. Simulation Settings

We set $M = 2$ UAVs flying above the 100×100 m^2 target area to serve N IoT devices, which belongs a set of $\{15, 20, ..., 40\}$. The initial coordinates of the UAVs are set at two opposite corners. In terms of neural network structure for both actor and critic networks, two fully connected hidden

Algorithm 1 The MADDPG-based UAV traveling route and IoT task offloading optimization

for UAV $m \in \mathcal{M}$ **do**
  Initialize actor network $\mu^m(\cdot)$, critic network $Q^m(\cdot)$, the Initial parameters are θ^{μ^m} and θ^{Q^m};
  Initialize target actor network $\mu^{m'}(\cdot)$, target critic network $Q^{m'}(\cdot)$, the initial parameters are $\theta^{\mu^{m'}}$ and $\theta^{Q^{m'}}$;
  Initialize experience replay buffer B_m;
end for
for Episode = 1 to 5000 **do**
  for UAV $m \in \mathcal{M}$ **do**
    Initialize observation $s_{m,t}$;
  end for
  for timeslot $t \in \mathcal{T}$ **do**
    Get state s_t;
    for UAV $m \in \mathcal{M}$ **do**
      Perform action $a_{m,t} = \mu^m\left(s_{m,t}|\theta^{\mu^m}\right)$;
    end for
    Update a_t;
    for IoT devices $n \in \mathcal{N}$ **do**
      Calculate $T_{n,m,t}$ and $E_{n,m,t}$;
      Obtain the offloading decision $\alpha_{n,m,t}$ according to Eq. (21)
    end for
    Get reward r_t;
    for UAV $m \in \mathcal{M}$ **do**
      Get $s_{m,t+1}$;
    end for
    Get s_{t+1};
    for UAV $m \in \mathcal{M}$ **do**
      Store $\{s_t, a_t, r_t, s_{t+1}\}$ into experience replay buffer B_m;
      if Training mode **then**
        Draw K samples from B_m;
        Update critic network, actor network and target networks according to Eqs. (18)-(20), respectively.
      end if
    end for
  end for
end for

layers with 256×256 neurons are used. The actor network and critic network are trained with a learning rate of $\tau = 0.001$. The Adam optimizer is used to update the actor and critic network. Other important simulation parameters are listed in Table I.

To verify the performance of the proposed MADDPG on UAV path optimization, the following two schemes are adopted for performance comparison, where both schemes adopt the same IoT task offloading policy as in Eq. (21).

1) *Random route*: Each UAV randomly selects the flight direction within $[0, 2\pi)$, and the UAVs are limited to flying within the target area.

TABLE I
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
N	$\{15, 20, ..., 40\}$	σ_o^2	10^{-12} Watt
M	2	f_L	200 MHz
l_{max}	100 m	t_w	20 ms
R_{max}	20 m	B_m	10^6
R_{min}	1 m	K	1024
H	50 m	τ	0.001
B	1MHz	p	2
P_{nU}	0.1 Watt	T	50
P_{nS}	0.5 Watt	η_1	10^{-27}
$C_{n,t}$	[1500,2000] Cycles/bit	γ	0.95
$D_{n,t}$	[10,15] Kb	f_S	5 GHz
f_U	1 GHz	r_{nS}	5 Mb/s

2) *Circle route*: The two UAVs each circle a half square field and fly in a circle.

B. Simulation Results Analysis

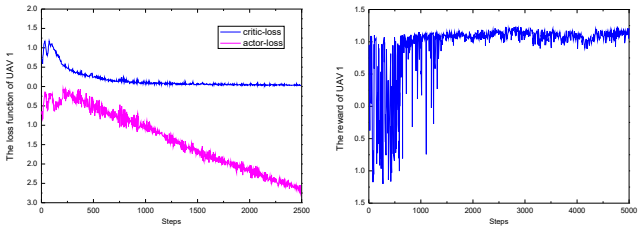

Fig. 3. The loss and reward of UAV 1 ($N = 15$).

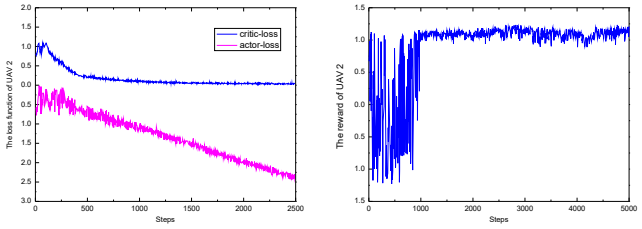

Fig. 4. The loss and reward of UAV 2 ($N = 15$).

The training curves of MADDPG for two UAVs are depicted in Figs. 3 and 4. In particular, as training continues, the losses of both the actor and critic network of two UAVs gradually decrease and finally reach convergence. The cumulative reward obtained by MADDPG remains at a low level when the training begins, and even appears negative and fluctuates greatly. The reward increases from the 1000-th step, and after about 1500 steps, the curve reaches around 1 and converges steadily.

Fig. 5 shows the travel routes of two UAVs under the guidance of the proposed MADDPG algorithm, where the blue dots represent the locations of 15 IoT devices, the stars and triangles represent the travel routes of UAV 1 and UAV 2, respectively. As can be seen from this figure, the UAVs move within the target area and find the optimal location to reduce latency and energy consumption. Additionally, we can see that each UAV covers a specific area cooperatively to maximize the defined reward.

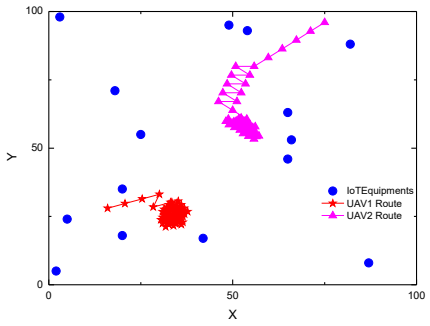

Fig. 5. The route of UAVs.

Figs. 6 and 7 show the trends of total energy consumption and total latency with the number of IoT devices. It can be seen that with the increase in the number of IoT devices, both the total energy and total latency increase linearly. The proposed MADDPG has the best performance, followed by the random route, and the circle route has the worst performance.

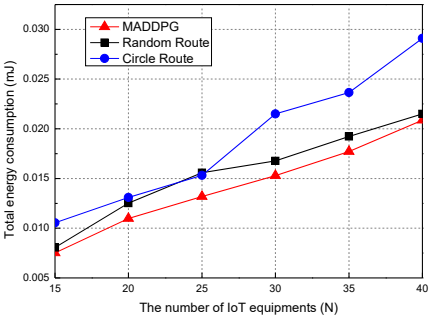

Fig. 6. The comparison of total energy consumption.

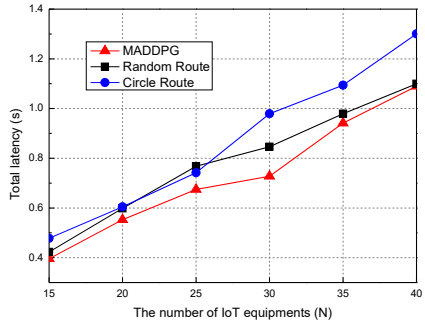

Fig. 7. The comparison of total latency.

As compared with the circle route, the energy consumption and latency of the proposed algorithm are reduced by an average of 20%. This is because after training, MADDPG assists the UAVs to cooperatively serve IoT devices, and more IoT devices can offload their tasks to the UAV, which will yield less energy consumption for all IoT devices.

V. Conclusion

In this paper, we propose a UAV/satellite-assisted mobile edge computing framework, in which multiple UAVs fly over the target area with different trajectories to provide support for the IoT devices on the ground. To adapt to the dynamic environment and utilize the interaction between multiple UAVs, MADDPG is adopted to jointly optimize the traveling routes of UAVs and offloading decisions of IoT tasks. The simulation results verify the convergence of the proposed algorithm and the effectiveness of MADDPG in the multi-UAV path planning process. As compared with the random route and circle route, the proposed MADDPG algorithm can decrease 20% of latency and energy consumption for IoT devices. The scheme proposed in this paper can help IoT devices process latency-sensitive and computationally intensive tasks efficiently. It also lays the foundation for the design and building of future IoT networks.

Acknowledgment

This work was supported in part by the the National Key Research and Development Program of China under Grant 2020YFB1806800; and in part by the Ministry of Education and China Mobile Joint Fund under Grant MCM20200202.

References

[1] Q. Cui, J. Zhang, X. Zhang, K.-C. Chen, X. Tao, and P. Zhang, "Online anticipatory proactive network association in mobile edge computing for iot," *IEEE Transactions on Wireless Communications*, vol. 19, no. 7, pp. 4519–4534, 2020.

[2] 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.

[3] Y. Wang, Z.-Y. Ru, K. Wang, and P.-Q. Huang, "Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3984–3997, 2020.

[4] F. Wang, D. Jiang, S. Qi, C. Qiao, and L. Shi, "A dynamic resource scheduling scheme in edge computing satellite networks," *Mob. Netw. Appl.*, vol. 26, no. 2, p. 597–608, apr 2021.

[5] Q. Cui, Z. Gong, W. Ni, Y. Hou, X. Chen, X. Tao, and P. Zhang, "Stochastic online learning for mobile edge computing: Learning from changes," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 63–69, 2019.

[6] N. Cheng, F. Lyu, W. Quan, C. Zhou, H. He, W. Shi, and X. Shen, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1117–1129, 2019.

[7] C. Zhou, W. Wu, H. He, P. Yang, F. Lyu, N. Cheng, and X. Shen, "Deep reinforcement learning for delay-oriented IoT task scheduling in SAGIN," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 911–925, 2021.

[8] N. Cheng, F. Lyu, W. Quan, C. Zhou, H. He, W. Shi, and X. Shen, "Space/aerial-assisted computing offloading for iot applications: A learning-based approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1117–1129, 2019.

[9] X. Tao, X. Xu, and Q. Cui, "An overview of cooperative communications," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 65–71, 2012.

[10] S. Mao, S. He, and J. Wu, "Joint UAV position optimization and resource scheduling in space-air-ground integrated networks with mixed cloud-edge computing," *IEEE Systems Journal*, vol. 15, no. 3, pp. 3992–4002, 2021.

[11] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *CoRR*, vol. abs/1706.02275, 2017. [Online]. Available: http://arxiv.org/abs/1706.02275

[12] G. Bo, X. Yang, Z. Lin, W. Hu, M. Alazab, and K. Rupak, "Multiagent actor-critic network-based incentive mechanism for mobile crowdsensing in industrial systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6182–6191, 2020.