Trajectory Optimization and Computing Offloading Strategy in UAV-Assisted MEC System

Youhui Gan, Yejun He*
Guangdong Engineering Research Center of Base Station Antennas and Propagation
Shenzhen Key Laboratory of Antennas and Propagation
College of Electronics and Information Engineering, Shenzhen University, 518060, China
Email:youhuigan@qq.com, heyejun@126.com*

2021 Computing, Communications and IoT Applications (ComComAp) | 978-1-6654-2798-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/COMCOMAP53641.2021.9652887

***Abstract*—Unmanned aerial vehicle (UAV) plays an important application scenario in mobile edge computing (MEC). In this paper, we jointly optimize the computing offloading strategy in MEC and the UAV trajectory to achieve the real-time communications between users and UAV, minimize the total energy consumption and delay, and improve users quality of service (QoS). As for computing offloading strategy, typically, UAV allocates channels to each user for data transmission. The energy consumption and delay generated by data transmission and computation are used as the evaluation criteria for offloading strategy. And the population diversity-binary particle swarm optimization (PDPSO) algorithm is used to obtain the optimal value. As for UAV trajectory optimization, the UAV communicates with the users in real time scenario. When the UAV arrives at a new position, it can receive the weighted sum of total energy consumption and delay based on computing offloading strategy. We aim to minimize the sum of energy consumption and delay at each position as the UAV completes a flight trajectory. Moreover, the deep deterministic policy gradient (DDPG) algorithm is used to obtain the optimal trajectory of the UAV. The simulation results show that our joint optimization algorithm is better than other joint algorithms.**

***Index Terms*—Mobile edge computing (MEC), unmanned aerial vehicle (UAV), computing offloading, population diversity-binary particle swarm optimization (PDPSO), UAV trajectory optimization, and deep deterministic policy gradient (DDPG)**

I. Introduction

With the increase of various Internet of Things (IoT) devices, such as wearable devices, portable devices and traffic monitoring devices, the demand for computing intensive applications is increasing. Due to the limitations of computing resources of these IoT devices, mobile edge computing (MEC) came into being. MEC provides more computing resources for IoT devices and these devices can send the tasks to the edge server for processing through computing offloading to reduce the energy consumption of IoT devices [1]. However, in some application scenarios, such as communication interruption caused by natural disasters, sudden increase in traffic density, the high agility and mobility of unmanned aerial vehicle (UAV) have become the optimal way to solve these kind of problems. Compared with the traditional ground base stations (BSs), UAV can be quickly deployed. Firstly, UAV can be used as a flight BSs to enhance wireless coverage in areas where communication is unavailable or some hot spots with large traffic. Secondly, UAV can act as a relay BSs in areas where communication is blocked due to natural disasters [2]. Therefore, we combine MEC with UAV to minimize energy consumption and delay, as well as optimization of the offloading strategy and UAV trajectory.

UAV has a wide range of applications. Zhang *et al.* completed the task offloading to UAV calculation by assigning time slots to each terminal equipment, and optimized the UAV trajectory with the goal of minimizing the total energy consumption [3]. Yu *et al.* proposed an optimal solution algorithm based on successive convex approximation, which minimized the weighted sum of equipment service delay and UAV energy consumption by optimizing UAV location, computing resource allocation, and task allocation [4]. Yan *et al.* studied the energy-saving management of resources in UAV assisted edge computing, and proposed a block successive upper-bound minimization (BSUM) algorithm [5].

Meanwhile, in terms of multi cell edge data offloading, Cheng *et al.* maximized the total rate of UAV service users by optimizing the trajectory of UAV and meeting the rate requirements of all users [6]. In terms of multi-UAV computing offloading and resource allocation, A. M Seid *et al.* proposed a collaborative computing offloading and resource allocation scheme for multi-UAV aerial to ground (A2G) network for emergency situations [7]. In terms of wireless power transfer, as the carrier of energy, UAV can not only complete communication calculation, but also transfer energy to equipment. Therefore, many researchers use the UAV to solve the fairness problem of energy received by the receiver. For example, Xu *et al.* proposed a continuous hovering UAV flight trajectory design [8], that is, the UAV hovers continuously for a period of time at a given set of hovering positions and flies at the maximum speed between these hovering positions.

Deep reinforcement learning (DRL) algorithm is widely used in MEC. There are many scholars using different algorithms in DRL to solve various problems in MEC. Wang *et al.* used DRL algorithm to reasonably allocate the edge computing network resources and achieve balance in different edge environments [9]. Tang *et al.* introduced deep Q-network (DQN) and double-DQN networks to solve the problem of task offloading in MEC [10], Laha Ale *et al.* proposed an end-to-end DRL algorithm to select the best edge server from multiple edge servers for offloading [11].

In this paper, we optimize the UAV trajectory and realize the real-time communication between UAV and users under the computing offloading strategy. Among that, UAV not only serves as a transit where users not communicate directly with edge BSs, but also has the function of offloading and computing some tasks of users. Therefore, the communication between users and UAV is the same as between UAV and edge BSs. Generally, we ignore the latter. In summary, the main contributions of this paper are as follows.

1) In terms of computing offloading strategy, we divide our work into partial offloading and all local computing. In partial offloading, we obtain the final offloading strategy through the optimal offloading proportion. And the proportion is obtained by minimizing the maximum value between the local processed time and the UAV processed time.

2) In terms of UAV trajectory optimization, the UAV communicates with the users after each continuous action, and then determines the next action according to the weighted sum of energy consumption and time delay based on the offloading strategy. Because the action of UAV is continuous, we can see it as real-time communication between UAV and users.

3) We jointly optimize the offloading strategy and UAV's trajectory. Under the premise of optimal offloading strategy, we obtain the optimal trajectory of UAV, which not only minimizes the value of users energy consumption and delay in each step, but also minimizes the value of energy consumption and delay caused by UAV communicating with users in continuous flight.

The rest of this paper is organized as follows. The system model and problem formulation are introduced in Section II. The process of population diversity-binary particle swarm optimization (PDPSO), and deep deterministic policy gradient (DDPG) algorithm are introduced in Section III. Section IV gives the simulation results. Section V summarizes the content of the full text.

II. System Model

The system model of UAV joint with the MEC is shown in Fig. 1. We define users, UAV and edge BSs in a three-dimensional space, and express their positions in the form of coordinates. There is a set of users $U(t) = \{U_1(t), U_2(t), ..., U_k(t)\}$ in t-th time slot, where the k is the number of users. We define the position of users in t-th time slot as $Q_{user}(t) = \{user_x(t), user_y(t), 0\}$. And there is an UAV flying from right to left for data transmission with k users. We define the real-time position of UAV in t-th time slot as $Q_{uav}(t) = \{X(t), Y(t), H\}$, where the height of UAV is fixed at H. As we can see from Fig. 1, because the users cannot communicate directly with the edge BSs, the position of the edge BSs is outside the communication range between the UAV and the users. Each user randomly generates data for calculation. The tasks are expressed as $L(t) = \{L_1(t), L_2(t), ..., L_k(t)\}$. Next, we introduce the whole system model from two aspects: communication model and computing model. Then, we bring out the problems and formulate them.

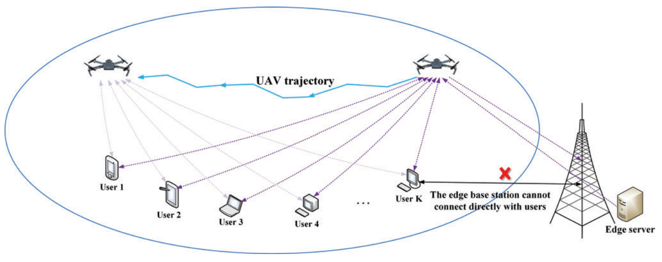


Fig. 1. The system model of UAV joint with the MEC.

A. Communication Model

The communication model is mainly the communication link between k users and UAV. Here, we consider the task uplink and ignore the downlink. We can calculate the channel gain $h_{up}(t)$ between each user and the UAV according to the locations of the users and the UAV, that is:

$$h_{up}(t) = \beta_0 d_{uu}^{-2}(t) = \frac{\beta_0}{H^2 + \|\mathbf{Q_{uav}}(t) - \mathbf{Q_{user}}(t)\|^2} \tag{1}$$

where β_0 represents the channel gain of 1m reference distance. $d_{uu} = \|\mathbf{Q_{uav}}(t) - \mathbf{Q_{user}}(t)\|$, where d_{uu} indicates the Euclidean distance between the UAV and the users corresponding to the t-th time slot. Here, both users and UAV need to move within a certain range:

$$Q_{uav}, Q_{user} \leq \{X_{size}, Y_{size}\} \tag{2}$$

With the channel gain between the users and the UAV, we can get the data transmission rate $r_{up}(t)$ as follows:

$$r_{up}(t) = B_0 \log_2 \left(1 + \frac{P_{user} h_{up}(t)}{\delta_0^2}\right) \tag{3}$$

where B_0 represents the bandwidth allocated to each user, P_{user} represents the transmission power of the user, and δ_0^2 represents the noise power at the UAV.

Therefore, we have the communication delay and communication energy consumption under the communication model:

$$E_{up}(t) = P_{user} T_{up}(t) = P_{user} \frac{L_{up}(t)}{r_{up}(t)} \tag{4}$$

where L_{up} represents the amount of user data partially offloaded to the UAV for calculation. $L_{up}(t) = \alpha(t) L_{load}(t)$, where $L_{load}(t)$ represents the total amount of data that the user in the t-th time slot chooses to partially offload when the offloading strategy is 1. And α represents the offloading proportion, which is calculated according to the minimum delay, and is described in the following formula. Here, we use binary numbers to represent the offloading strategy $\varphi(t)$, as shown below:

$$\varphi(t) = \begin{cases} 1, & \text{partial offload} \\ 0, & \text{total local} \end{cases} \tag{5}$$

B. Computation Model

The computation model is divided into total local computation and partial offloading computation, in which the partial offloading includes αL_{load} data volume offloaded to UAV computation and $(1-\alpha)L_{load}$ data volume left for local computation. Therefore, we get the computation delay and energy consumption of total local and partial offloading as follows:

1) *Total local*:

$$T_{local}(t) = \frac{L_{local}(t)C_{user}}{f_{user}} = \frac{(1-\varphi(t))L(t)C_{user}}{f_{user}} \tag{6}$$

$$\begin{aligned} E_{local}(t) &= K_{user}(f_{user})^3 T_{local}(t) \\ &= K_{user}L_{local}(t)C_{user}(f_{user})^2 \end{aligned} \tag{7}$$

where K_{user} represents the CPU capacitance coefficient of users equipment, C_{user} represents the number of CPU cycles required by the users equipment to process 1 bit data, and f_{user} represents the users local computing resources.

2) *Partial offload*:

Part I : $(1-\alpha)L_{load}$ data local computing

$$T_{user}(t) = \frac{(1-\alpha)L_{load}C_{user}}{f_{user}} = \frac{(1-\alpha(t))\varphi(t)L(t)C_{user}}{f_{user}} \tag{8}$$

$$E_{user}(t) = K_{user}(f_{user})^3 T_{user}(t) \tag{9}$$

Part II : αL_{load} data UAV computing

$$T_{uav}(t) = \frac{\alpha L_{load}C_{uav}}{f_{uav}} = \frac{\alpha(t)\varphi(t)L(t)C_{uav}}{f_{uav}} \tag{10}$$

$$E_{uav}(t) = K_{uav}(f_{uav})^3 T_{uav}(t) \tag{11}$$

Similarly, K_{uav}, C_{uav} and f_{uav} respectively represent the CPU capacitance coefficient of UAV, the number of CPU cycles required for UAV to process 1 bit data, and UAV computing resources.

Therefore, we have the delay and energy consumption in the case of task offloading as follows:

$$T_{load}(t) = max(T_{user}(t), T_{up}(t) + T_{uav}(t)) \tag{12}$$

$$E_{load}(t) = E_{user}(t) + E_{up}(t) + E_{uav}(t) \tag{13}$$

C. Problem Formulation

We have obtained the delay and energy consumption of each part. Then, for the t-th time slot occupied by the n-th user and on the basis of determining the offloading strategy φ, the total delay and total energy consumption are as follows:

$$\begin{aligned} T(n,t) &= (1-\varphi(n,t))T_{local}(n,t) + \varphi(n,t)T_{load}(n,t) \\ &= (1-\varphi(n,t))T_{local}(n,t) + \varphi(n,t) \\ &\quad (max(T_{user}(n,t), T_{up}(n,t) + T_{uav}(n,t))) \end{aligned} \tag{14}$$

$$E(n,t) = (1-\varphi(n,t))E_{local}(n,t) + \varphi(n,t)E_{load}(n,t) \tag{15}$$

Therefore, we define the problem 1 to be solved as finding the minimum value of the weighted sum of total energy consumption and delay of data transmission and computation between all users and UAV:

$$\begin{aligned} \text{P1:} \quad & \min_{E_{load}(t), T_{load}(t)} \omega \sum_{n=1}^{k} E(n,t) + (1-\omega)\sum_{n=1}^{k} T(n,t) \\ \text{s.t.} \quad & \text{C1} : Q_{uav}, Q_{user}^{n} \le \{X_{size}, Y_{size}\} \quad \forall n \in k \\ & \text{C2} : \varphi(n,t) \in \{0,1\} \quad \forall n \in k, sum(t) \ge k \\ & \text{C3} : min\,[max(T_{user}, T_{up} + T_{uav})] \end{aligned} \tag{16}$$

For the third constraint, we can use mathematical methods to calculate the optimal offloading proportion α that minimizes the total offloading time. It is known that when a single user offloads a task in a single time, the task amount is unchanged, so the task amount offload to the local and UAV is inversely proportional, and the corresponding T_{user} and $T_{up} + T_{uav}$ are also inversely proportional. According to the characteristics of the mathematical increase and decrease function, $max(T_{user}, T_{up} + T_{uav})$ is the smallest only when $T_{user} = T_{up} + T_{uav}$. Thus, we get the optimal offloading proportion $\alpha(t)$:

$$min\,[max(T_{user}, T_{up} + T_{uav})] \tag{17}$$

$$\Longrightarrow T_{user} = T_{up} + T_{uav} \tag{18}$$

$$\Longrightarrow \frac{(1-\alpha)L_{load}C_{user}}{f_{user}} = \frac{\alpha L_{load}}{r_{up}} + \frac{\alpha L_{load}C_{uav}}{f_{uav}} \tag{19}$$

$$\Longrightarrow \alpha = \frac{C_{user}f_{uav}r_{up}}{(C_{uav}f_{user} + C_{user}f_{uav})r_{up} + f_{user}f_{uav}} \tag{20}$$

After obtaining the solution of the P1 problem, we complete the trajectory optimization of the UAV. We define the problem as that the UAV flies from the starting point to the end point, and interacts with the users all the time. Finally, the optimization goal is to minimize the weighted sum of the total energy consumption and delay generated by the real-time interaction between the UAV and the users.

$$\begin{aligned} \text{P2:} \quad & \min_{min(P1)} \sum_{step=1}^{\zeta} \Bigg[\omega \sum_{n=1}^{k} E(n,t,step) + \\ & \qquad (1-\omega)\sum_{n=1}^{k} T(n,t,step) \Bigg] \\ \text{s.t.} \quad & \text{C1} : Q_{uav}, Q_{user}^{n} \le \{X_{size}, Y_{size}\} \quad \forall n \in k \\ & \text{C2} : Q_{uav} : Q_{uav}^{start} \to Q_{uav}^{end} \end{aligned} \tag{21}$$

where ζ represents the total flight steps of the UAV. The UAV interacts with the users at each flight step to obtain a new computation offloading strategy to determine the flight direction of the next step. Finally, the total energy consumption and delay generated at each step is taken as the optimization goal for each flight process, where the UAV path with the minimum value is considered as the optimal trajectory.

III. Algorithm Design

In this section, we introduce the implementation process of PDPSO algorithm and DDPG algorithm, and how to combine these two algorithms to complete the trajectory optimization and computing offloading of UAV.

A. *PDPSO Algorithm*

The classical PSO algorithm is not described here. Based on the classical PSO algorithm, PDPSO algorithm obtains better optimization results by affecting the inertia weight ϖ of particle velocity update. We give the expression of particle position x_i and velocity v_i update as follows:

$$\begin{aligned} v_i(t+1) = \varpi(t)v_i(t) + c_1 r_1 \left[pbest - x_i(t)\right] + \\ c_2 r_2 \left[gbest - x_i(t)\right] \\ x_i(t+1) = x_i(t) + v_i(t+1) \end{aligned} \tag{22}$$

where c_1 and c_2 are the learning factor, r_1 and r_2 are the random numbers between [0,1], $pbest$ represents the optimal solution of each particle in iteration t, and $gbest$ represents the optimal solution of the whole population in iteration t.

We take the weighted sum of energy consumption and delay as the fitness function. After the algorithm converges, the particle position x_i is concentrated near the optimal value and the speed v_i is close to 0. Therefore, we introduce sigmoid function to determine the offloading strategy:

$$Sigmoid\left(v_i(t)\right) = \frac{1}{1 + e^{-v_i(t)}} \tag{23}$$

$$\varphi_i(t) = \begin{cases} 1, \rho < Sigmoid\left(v_i(t)\right) \\ 0, \ \text{otherwise} \end{cases} \tag{24}$$

where ρ is a random number between [0,1]. $\varphi = 1$ means partial offloading of the i-th user task. And $\varphi = 0$ means that the task of the i-th user is computed all locally.

Thus, we focus on how to get the inertia weight function ϖ. Here, we introduce population diversity D to optimize the inertia weight ϖ of each iteration.

$$D(t+1) = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}\left(\overline{d_i(t+1)} - d_i(t+1)\right)^2} \tag{25}$$

where m represents the number of particles and also is the number of users, $\overline{d_i(t)}$ represents the average Euclidean distance between the i-th particle and other particles, and $d_i(t)$ represents the minimum Euclidean distance between the i-th particle and other particles.

We calculate the inertia weight function $\varpi(t)$ according to the population density $D(t)$:

$$\varpi_i(t+1) = \begin{cases} \varpi_i(t)(e^{\frac{1}{D(t+1)+1}-1} + 1), D(t+1) \geq D(t) \\ \varpi_i(t)(e^{\frac{1}{D(t+1)+1}-1}), D(t+1) < D(t) \end{cases} \tag{26}$$

B. *DDPG Algorithm*

As a classical algorithm in DRL, DDPG adopts continuous action to better meet the trajectory requirements of UAV. Next, several important concepts in DDPG are introduced:

States: It mainly includes the locations of all users and UAV, and the amount of data of all users.

$$\begin{aligned} State = \{Q_{user}(n,t), Q_{uav}, L(n,t) | \ \forall n,t \in k, \\ \forall Q \in \{X_{size}, Y_{size}\}\} \end{aligned} \tag{27}$$

Action: UAV flies at a constant speed, so the flight energy consumption of UAV is only related to time. Therefore, we specify the flight heading angle θ of the UAV as the action choice.

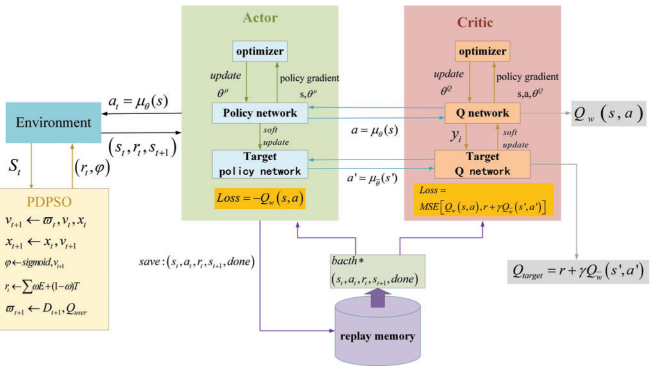
$$Action = \{\theta | \ \theta \in [0, 2\pi]\} \tag{28}$$

Reward: Here, every step of the UAV gets an action, and flies to a new location for data transmission with the users. The $reward'$ at this time is the weighted sum of the total energy consumption and delay after the PDPSO algorithm obtains the optimal offloading strategy and allocates the offloading task. We take the sum of $reward'$ corresponding to the total steps of UAV flight as $reward$.

$$reward = \sum_{step=1}^{\zeta} reward'(step) \tag{29}$$

$$reward' = -\min_{\varphi}\left[\omega\sum_{n=1}^{k} E + (1-\omega)\sum_{n=1}^{k} T\right] \tag{30}$$

Here, we assume that the users and UAV move within a certain range $\{X_{size}, Y_{size}\}$. If the UAV flies out of the specified range, we give a bad reward.



Actor
Critic
optimizer
update θ^μ
policy gradient s,θ^μ
optimizer
update θ^Q
policy gradient s,a,θ^Q
Environment
$a_t = \mu_\theta(s)$
(s_t, r_t, s_{t+1})
S_t
(r_t, φ)
Policy network
soft update
Target policy network
$Loss = -Q_w(s,a)$
Q network
$Q_w(s,a)$
$a = \mu_\theta(s)$
y_i
soft update
Target Q network
$a' = \mu_{\bar\theta}(s')$
$Loss = MSE[Q_w(s,a), r+\gamma Q_{\bar w}(s',a')]$
PDPSO
$v_{t+1} \leftarrow \varpi_t, v_t, x_t$
$x_{t+1} \leftarrow x_t, v_{t+1}$
$\varphi \leftarrow sigmoid, v_{t+1}$
$r_t \leftarrow \sum \omega E + (1-\omega)T$
$\varpi_{t+1} \leftarrow D_{t+1}, Q_{user}$
$save: (s_t, a_t, r_t, s_{t+1}, done)$
bacth* $(s_t, a_t, r_t, s_{t+1}, done)$
replay memory
$Q_{target} = r + \gamma Q_{\bar w}(s', a')$

Fig. 2. Optimization process of joint PDPSO and DDPG algorithm.

C. *Problem Approach by Joint PDPSO and DDPG*

As shown in Fig. 2, we combine PDPSO and DDPG to solve the proposed problems, mainly including 1) the environment interacts with PDPSO network to obtain the best $reward$ and offloading strategy φ, 2) the environment interacts with the four networks in DDPG to obtain the best action. Among that, the method of computing and updating the actor network after batch size data sampling is as follows:

$$\begin{aligned} &\nabla_{\theta^\mu} J(\mu) \approx \\ &\frac{1}{N}\sum_i \left[\nabla_{\theta^\mu}\mu(s;\theta^\mu)|_{s=s_i} \nabla_a Q(s,a;\theta^Q)|_{s=s_i, a=\mu(s_i;\theta^\mu)}\right] \end{aligned} \tag{31}$$

The total pseudo code of the algorithm is shown in Algorithm 1.

Algorithm 1: Computing Offloading and UAV trajectory by Joint PDPSO and DDPG

```
Initialize actor policy network $\mu(s;\theta^{\mu})$, target policy
  network $\mu'$, critic policy network $Q(s,a;\theta^{Q})$, target
  policy network $Q'$ with weights $\theta^{\mu}$, $\theta^{\mu'}$, $\theta^{Q}$, $\theta^{Q'}$;
Initialize replay memory as $rpm$;
for episode $ep = 1, M$ do
    Randomly generate a random process $\eta$ for action
      exploration and limit the output to [0, 1];
    Randomly initialize the state $s_1$;
    for step $t = 1, T$ do
        Select action $a_t = \mu(s_t;\theta^{\mu}) + \eta_t$ according to
          the current policy and exploration noise;
        Enter state $s_t$ and action $a_t$ into the enivorment;
        Initialize particle position $x_1$, velocity $v_1$,
          inertia weight $\varpi$, $pbest$ and $gebst$;
        for iteration $i = 1, I$ do
            for each particle $n = 1, k$ do
                Update velocity $v_{i+1}$ and position $x_{i+1}$;
                Computing offloading strategy:
                  $$\varphi_i = \begin{cases} 1, & \rho < Sigmoid\left(v_{i+1}\right) \\ 0, & otherwise \end{cases}$$
            end
            Get $r_{t,i} = \omega \sum_{n=1}^{k} E(n,t,i) +$
              $(1-\omega)\sum_{n=1}^{k} T(n,t,i)$ ;
            Update $pbest$, $gbest$, diversity $D$ and $\varpi$;
        end
        Obtain reward $r_t$ and new state $s_{t+1}$;
        Store $(s_t, a_t, r_t, s_{t+1}, done)$ in $rpm$;
        Sample a batch size of $(s_j, a_j, r_j, s_{j+1}, done)$
          from $rpm$;
        Set $y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1} \mid \theta^{\mu'}\right) \mid \theta^{Q'}\right)$;
        Update critic by minimizing the loss:
          $L = \frac{1}{N}\sum_i \left(y_i - Q\left(s_i, a_i \mid \theta^{Q}\right)^2\right)$;
        Update the actor policy using the sampled
          gradient: $\nabla_{\theta^{\mu}} J(\mu)$;
        Update the target networks:
            $\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'}$
            $\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'}$
    end
    $reward = \sum_{t=1}^{T} r_t$
end
```

IV. Simulation Results

In this section, we solve the trajectory of UAV and the computing offloading strategy in MEC according to the proposed joint algorithm. The simulation results are described below. We use Python for simulation design, and the model simulation parameters are shown in Table I.

Fig. 3 shows the comparison of joint optimization algorithms. We use three different joint optimization algorithms, which are 'DDPG+PDPSO', 'DQN+PDPSO', and 'SP+PDPSO'. The DDPG algorithm parameters are as follows: soft update coefficient $\tau = 0.001$, reward discount factor $\gamma = 0.9$, A-C network learning rate $lr = 0.001$, and action noise variance $\sigma = 0.05$. The DQN algorithm parameters are roughly the same with DDPG. And the SP algorithm uses the shortest path (SP) algorithm to optimize the UAV trajectory. Because the state space of users locations is too large, we take the average value of 10 training times as an episode value. Similarly, the final convergence value also has some fluctuations. As we can see in Fig. 3, the final convergence of 'DDPG+PDPSO' is better than the other two joint optimization algorithms.

TABLE I
Simulation Parameters

Parameters	Value
The number of users k	10
The amount of data generated by each user L	[1,10]Mbits
The height of UAV H	100m
Range of users and UAV movement $\{X_{size}, Y_{size}\}$	[100m,100m]
The channel gain of 1m reference distance β_0	-50dB
The bandwidth allocated to each user B_0	10MHz
The transmission power of the users P_{user}	0.5W
The noise power at the UAV δ_0^2	-70dBm/Hz
The CPU capacitance coefficient of users K_{user}	10^{-27}
The CPU capacitance coefficient of UAV K_{uav}	10^{-28}
Required CPU cycles per bit computation at local C_{user}	800 cycles/bit
Required CPU cycles per bit computation at UAV C_{uav}	1000 cycles/bit
The users computing resources f_{user}	1GHz
The UAV computing resources f_{uav}	3GHz
The weight of energy consumption and delay ω	0.75

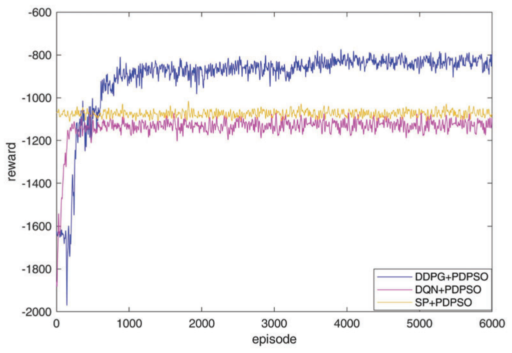


Fig. 3. Comparison of joint optimization algorithms.

Fig. 4 shows the optimal trajectory of UAV for different users distribution in the case of final convergence, in which the green line represents the UAV trajectory, the red dot represents one user and its data offloading strategy is partial offloading, and the blue dot represents one user and its data offloading strategy is all local. Here, 10 users are randomly distributed, and the UAV flies from the starting point coordinate $(0,0)$ to the end point coordinate $(100,100)$. As shown in Fig. 4(a) and Fig. 4(b), when the location of the users is scattered or concentrated near the diagonal, the UAV selects the shortest path to fly in order to minimize energy consumption and time delay. As shown in Fig. 4(c) and Fig. 4(d), when the overall

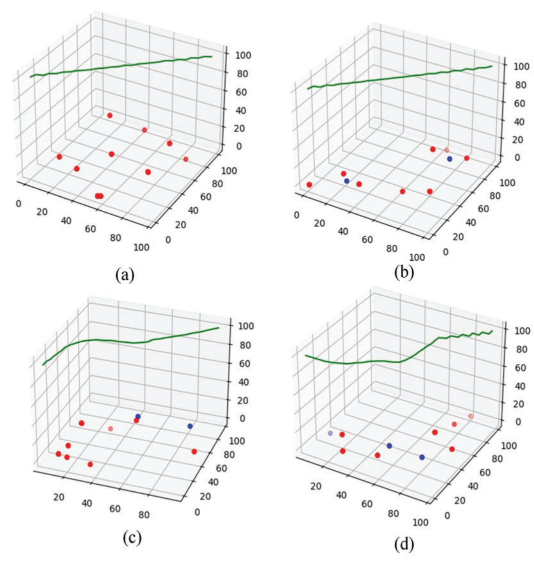


Fig. 4. The optimal trajectory of UAV for different users distribution.

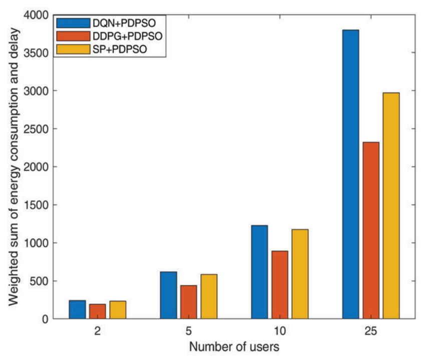


Fig. 5. Total energy consumption and delay versus the number of users.

location of the users is biased towards a certain direction, the UAV flies according to the users direction to minimize energy consumption and time delay.

Fig. 5 shows the changing of total energy consumption and delay with the increase of the number of users. As we can see, 'DDPG+PDPSO' algorithm is relatively optimal regardless of the number of users. When the number of users is large, 'DQN+PDPSO' algorithm performs poorly, while 'SP+PDPSO' algorithm increases linearly with the increase of the number of users.

V. Conclusion

In this paper, we proposed a joint optimization algorithm to solve the offloading strategy in MEC and UAV's trajectory. We used PDPSO algorithm to solve the offloading strategy, and used mathematical method to obtain the optimal offloading proportion. At the same time, we used DDPG algorithm to optimize the trajectory of UAV. Finally, we jointly optimized the computing offloading strategy and UAV trajectory. Under the condition of the optimal offloading strategy, we obtained the optimal UAV trajectory and realized the real-time data transmission and computation between users and UAV.

There are still some challenges in UAV-assisted MEC system. In the future work, we will further study multiple UAV-assisted MEC system, including interference between multi-UAV, and offloading selection between multi-UAV and multi-user.

Acknowledgment

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. 62071306, and in part by Shenzhen Science and Technology Program under Grants JCYJ20200109113601723 and JSG-G20210420091805014.

References

[1] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey and C. S. Hong, "Energy-Efficient Resource Management in UAV-Assisted Mobile Edge Computing," *IEEE Communications Letters*, vol. 25, no. 1, pp. 249-253, Jan. 2021.

[2] H. Wang, G. Ding, F. Gao, J. Chen, J. Wang and L. Wang, "Power Control in UAV-Supported Ultra Dense Networks: Communications, Caching, and Energy Transfer," *IEEE Communications Magazine*, vol. 56, no. 6, pp. 28-34, June 2018.

[3] T. Zhang, Y. Xu, J. Loo, D. Yang and L. Xiao, "Joint Computation and Communication Design for UAV-Assisted Mobile Edge Computing in IoT," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5505-5516, Aug. 2020.

[4] Z. Yu, Y. Gong, S. Gong and Y. Guo, "Joint Task Offloading and Resource Allocation in UAV-Enabled Mobile Edge Computing," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3147-3159, April 2020.

[5] Y. K. Tun, Y. M. Park, N. H. Tran, W. Saad, S. R. Pandey and C. S. Hong, "Energy-Efficient Resource Management in UAV-Assisted Mobile Edge Computing," *IEEE Communications Letters*, vol. 25, no. 1, pp. 249-253, Jan. 2021.

[6] F. Cheng *et al*., "UAV Trajectory Optimization for Data Offloading at the Edge of Multiple Cells," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6732-6736, July 2018.

[7] A. M. Seid, G. O. Boateng, S. Anokye, T. Kwantwi, G. Sun and G. Liu, "Collaborative Computation Offloading and Resource Allocation in Multi-UAV-Assisted IoT Networks: A Deep Reinforcement Learning Approach," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12203-12218, 1 Aug.1, 2021.

[8] J. Xu, Y. Zeng and R. Zhang, "UAV-Enabled Wireless Power Transfer: Trajectory Design and Energy Optimization," *IEEE Transactions on Wireless Communications*, vol. 17, no. 8, pp. 5092-5106, Aug. 2018.

[9] J. Wang, L. Zhao, J. Liu and N. Kato, "Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 3, pp. 1529-1541, 1 July-Sept. 2021.

[10] M. Tang and V. W. S. Wong, "Deep Reinforcement Learning for Task Offloading in Mobile Edge Computing Systems," *IEEE Transactions on Mobile Computing*, early access, 2020.

[11] L. Ale, N. Zhang, X. Fang, X. Chen, S. Wu and L. Li, "Delay-Aware and Energy-Efficient Computation Offloading in Mobile-Edge Computing Using Deep Reinforcement Learning," *IEEE Transactions on Cognitive Communications and Networking*, vol. 7, no. 3, pp. 881-892, Sept. 2021.