Task Offloading for UAV-based Mobile Edge Computing via Deep Reinforcement Learning

Jun Li*, Qian Liu*, Pingyang Wu*, Feng Shu* and Shi Jin§
*School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing, 210094
§School of Information Science and Engineering, Southeast University, Nanjing, 211189
E-mail:{jun.li, qian.liu, pingyang.wu, feng.shu}@njust.edu.cn, jinshi@seu.edu.cn

***Abstract*—With rapid increase of data processing demands from users in mobile edge computing (MEC), the conventional mobile edge servers (MESs) are no longer capable of providing timely and effective services. Against this background, we focus on applying unmanned aerial vehicle (UAV) as an MES to provide computational task offloading services for users. In this paper, we aim at maximizing the migration throughput of user tasks with limited energy at the UAV. To be specific, we first formulate the maximization problem as a semi-Markov decision process (SMDP) without transition probability. Then we propose the deep reinforcement learning (DRL)-based scheme of maximizing user tasks migration throughput to solve the maximization problem. The scheme realizes a maximum autonomic migration throughput of users with limited UAV energy and improves quality of service (QoS) of MEC to some extent. Simulation results demonstrate that the proposed scheme is sufficient with favourable convergence.**

***Index Terms*—Unmanned aerial vehicle, semi-Markov decision process, deep reinforcement learning**

I. Introduction

Unmanned aerial vehicle (UAV) can be controlled remotely to complete some specific tasks in military and civilian domains. Especially, UAV can be employed as a mobile base station for emergency communication when conventional base station is hard to implement in remote or disaster districts. The communication system including UAV has more favourable advantages than traditional ones [1].

Recently, many scholars have begun to study the communication systems based on UAV. The work in [2] proposes an UAV communication system based on 4G technology to enhance network coverage. The study in [3] employs multiple UAV-mounted aerial base stations to maximize the minimum throughput over all users. Safeguarding data transmissions is also an important topic, via transmitting useful signals and noise together [4] or designing a new trajectory for UAV to complete anti-interference transmission [5]. However, the above mentioned works applying UAVs as mobile base stations solve the problem by static programming, which is no longer useful to solve problem with unknown environment.

As for the part of energy and path planning, the authors in [6] propose a new fundamental energy trade-off in ground-to-UAV wireless communication. The study in [7] considers the general constraints on the trajectory of UAV, including its locations and velocities, as well as acceleration. The Euclidean paths are also mapped to energy costs in the path planning problem in [8], which is solved with the 4-zone strategy using Dijkstras Algorithm. Markov decision process (MDP) has also been used to solve path planning problem. In automatic control domain, many studies of path planning for UAV are modeled as MDP for different purpose i.e. persistent surveillance, UAV navigation [9] and obstacle avoidance [10]. But the above mentioned works solve path planning problem with MDP without considering communications between users and UAVs.

At the same time, mobile edge servers (MESs) provide computational task offloading services for users is also a hot topic in mobile edge computing (MEC).The work in [11] proposes an online learning algorithm to decide how to process the computing tasks in order to minimize the delay cost. Online learning can also provide online sequential control decisions given full or partial information about the optimal controls in MDP.

To the best of author's knowledge, we first employ UAV as an MES to provide computational task offloading services for users. Our goal is to maximize the migration throughput of user tasks. To achieve this, we first formulate the problem as a semi-Markov decision process (SMDP) without transition probability, which is model-free and does not need any prior information about environment. Then, we convert the optimization problem into maximizing the average return of SMDP during processing period. Next, we propose deep reinforcement learning (DRL)-based scheme of maximizing user tasks migration throughput to maximize average return of SMDP. The proposed scheme makes use of reinforcement learning (RL) and neural network (NN) together to deal with the explosion of state space and improve the quality of service (QoS) of MEC to some extent.

The contributions of this paper are as follows:

- We formulate the optimization problem as an SMDP with combination state spaces of location states of user and UAV, UAV surplus energy state and channel gain state between user and UAV, which takes into account the various factors in the environment to solve joint multi-states Markov problem efficiently.
- The proposed DRL-based scheme of maximizing user tasks migration throughput has two NNs with same structure but different parameters. One of them updates its parameters every step but the other one updates its parameters after a fixed interval, which makes the proposed scheme efficient and converge quickly.

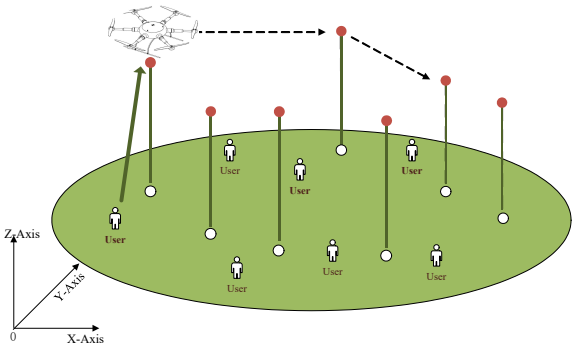

Fig. 1. System model of single UAV and multiple users: the red points represent the M fixed perceptual access points and the white points is the red points mapping on the ground.

II. System Modle

As depicted in Fig.1, the system model consists of single UAV and multiple users. UAV is able to serve all users but it can only serve one user in one time. N users are randomly divided in a region, with $l_i(t) = (x_i, y_i), i \in \{1, 2, ..., N\}$ representing the location of the ith user. Due to its limited processing capacity, the ith user has to communicate with UAV to upload its computation tasks. For simplicity, UAV can only hover over M fixed perceptual access points within the distribution range of user. $d_j(t) = (x_j, y_j), j \in \{1, 2, ..., M\}$ represents the location of the jth point, indicating the location of UAV at this point. Then UAV can form a direct connection with the ith user and complete the corresponding communication and computation tasks.

We consider a discrete-time model by dividing the operating period into time slots of unequal length, indexed by $t \in \{0, 1, 2..., T\}$. T is a constant and represents the total times of service. Each time slot t has two durations: UAV flies to the jth perceptual access point and hovers there to receive and complete the offloading tasks $\mu_i(t)$ from the ith user. We assume that the number of bits per task N_b is fixed and all tasks have same N_b, which means the same $\mu_i(t)$ consumes same CPU cycles of UAV. UAV is powered by a battery of fixed capacity B. It has three kinds of energy consumption during services: UAV flying consumption $e_{\mathrm{f}}(t)$, UAV hovering consumption $e_{\mathrm{h}}(t)$ and UAV calculating consumption $e_{\mathrm{c}}(t)$. And the surplus energy state of UAV at time slot t is $b(t) \in [0, B]$. Though UAV is able to calculate and process the offloading tasks from users, it should receive the offloading tasks $\mu_i(t)$ reasonably at each time slot according to its energy limitation $\sum_t [e_f(t) + e_h(t) + e_c(t)] < B$. Worst of all is that UAV only serves one fixed user, who is the closest to it or has largest throughput. Therefore, it also should satisfy the quality of service (QoS) of each user $\sum_t \mu_i(t) < Z$. Z is a constant and represents the lowest limit of QoS.

III. Problem Formulation

The process of maximizing migration throughput of user tasks, $\max \sum_{i,t} \mu_i(t)$, based on limited energy of UAV can be formulated as an SMDP. So that we can maximize average return in one episode of SMDP to maximize the average migration throughput of user tasks. In the following of this section, we define the state space, action space, reward function of SMDP and the objective function of this system model.

A. State Space

The system state space is $S_t = \{s_t | t = 0, 1, 2..., T\}$ in time slot t, where s_t is described by a 4-tuple $s_t = \{l_i(t), d_j(t), b(t), c_{i,j}(t)\}$. $l_i(t)$ is the location state of the ith user and $d_j(t)$ means the location state of UAV in time slot t. The surplus energy state of UAV is $b(t)$ and $c_{i,j}(t)$ represents the channel gain state between the ith user and the jth perceptual access point, which has large effects on transmission speed, as well as UAV hovering energy consumption $e_{\mathrm{h}}(t)$.

1) Location States of User and UAV: User location state $l_i(t)$ and UAV location state $d_j(t)$ are described as follows:

$$\begin{aligned} l_i(t) &= (x_i, y_i),\ i \in \{1, 2, ..., N\}, \\ d_j(t) &= (x_j, y_j),\ j \in \{1, 2, ..., M\}. \end{aligned} \tag{1}$$

2) Surplus Energy State: The initial energy state $b(0)$ of UAV is equal to B. Then UAV surplus energy $b(t)$ decreases with UAV serving users. So the iterative formula of $b(t)$ is:

$$b(t+1) = b(t) - [e_{\mathrm{f}}(t) + e_{\mathrm{h}}(t) + e_{\mathrm{c}}(t)]. \tag{2}$$

3) Channel Gain State: Generally, the transmission channel between user and UAV adopts Power Law Model. Meanwhile, we do not consider the influence of rayleigh fading on the channel gain condition. So the channel gain is only determined by path loss. We define the channel gain state $c_{i,j}(t)$ between the ith user and the jth point in time slot t as:

$$\begin{aligned} c_{ij}(t) &= \rho^{-\zeta}, \\ \rho &= \sqrt{H^2 + [x_j(t) - x_i(t)]^2 + [y_j(t) - y_i(t)]^2}, \end{aligned} \tag{3}$$

where H is UAV fixed flying altitude and ρ is the distance between the ith user and the jth point. We choose the path loss exponent ζ as 4 according to [12].

B. Action Space

The UAV can only hover over M perceptual access points, so we let the M points as M actions. The action space A in time slot t is described as:

$$A_t = \{a_j | j = 1, 2, ..., M\}, \tag{4}$$

where a_j represents the UAV flying to the jth point in time slot t.

C. Reward Function

The UAV attains gains $U(\mu_i(t))$ when it receives and processes the offloading tasks $\mu_i(t)$ from ith user. But it also leads to energy consumption $W(t)$ because of flying, hovering and calculating. The system reward in time slot t can be attained at beginning of next time slot $t + 1$, so the system reward of current state S_t and action A_t is:

$$R_{t+1} = U(\mu_i(t)) - W(t). \tag{5}$$

1) System Gain: UAV is applied as an MES for users to complete communication and computation tasks. When users offload tasks, we must give some gains to motivate UAV to serve users. The relationship between offloading tasks $\mu_i(t)$ and gains $U(\mu_i(t))$ is described by a sigmoid form:

$$U(\mu_i(t)) = 1 - \exp\left[-\frac{(\mu_i(t))^\theta}{\mu_i(t)+\gamma}\right], \tag{6}$$

where θ and γ are parameters for adjusting the efficiency of $U(\mu_i(t))$ and they can determine the effect of the number of offloading tasks $\mu_i(t)$. We choose $\theta = 2$ and $\gamma = 10$ as the optimal parameter pairs according [13].

2) System Consumption:

- UAV Flying Energy Consumption $e_{\mathrm{f}}(t)$
  We assume the flight power P_{f} and flight speed V are both certain. UAV flying energy consumption $e_{\mathrm{f}}(t)$ is only determined by the flight distance from one perceptual access point to the other one during time slot t:

$$e_{\mathrm{f}}(t) = P_{\mathrm{f}}\frac{\sqrt{[x_j(t)-x_j(t-1)]^2+[y_j(t)-y_j(t-1)]^2}}{V}. \tag{7}$$

- UAV Hovering Energy Consumption $e_{\mathrm{h}}(t)$
  When UAV flies to the jth point, it also should hover there until completing the offloading tasks from the ith user. UAV hovering energy consumption $e_{\mathrm{h}}(t)$ is determined by transmission speed $R_{ij}(t)$, which is indirectly affected by the channel gain state $c_{i,j}(t)$ between the ith user and the jth point [3]:

$$R_{ij}(t) = \log_2(1+\frac{P_{\mathrm{t}}c_{ij}(t)}{\sigma^2}), \tag{8}$$

  where P_{t} is transmission power, σ^2 is Gaussian white noise and both of them are constants. UAV hovering power is P_{h} and the number of bits per task is N_b, so UAV hovering energy consumption is:

$$e_{\mathrm{h}}(t) = P_{\mathrm{h}}\frac{\mu_i(t)N_b}{R_{ij}(t)}. \tag{9}$$

- UAV Calculating Energy Consumption $e_{\mathrm{c}}(t)$
  After receiving offloading tasks $\mu_i(t)$ from the ith user, it is necessary for UAV to calculate and process these tasks as soon as possible. Since we do not consider the different types of offloading tasks, UAV calculating energy consumption $e_{\mathrm{c}}(t)$ is determined by the number of offloading tasks $\mu_i(t)$ from the ith user at time slot t:

$$e_c(t) = \beta\mu_i(t)N_b, \tag{10}$$

  where β is a variable parameter.

Consequently, the total system consumption $W(t)$ of UAV includes flying, hovering and calculating energy consumption:

$$W(t) = \psi_1 e_{\mathrm{f}}(t) + \psi_2 e_{\mathrm{h}}(t) + \psi_3 e_{\mathrm{c}}(t), \tag{11}$$

where ψ_1, ψ_2 and ψ_3 are variable parameters to coordinate the proportion of three kinds energy consumption.

D. The Formulation of Objective Function

With training episodes increasing, the total reward in one episode of SMDP is bound to increase. But its raise does not represent an accurate improvement of system performance. So we demonstrate the average throughput in terms of the average return g in one episode. We define g as the sum of rewards in one episode divided the total number of services T:

$$g = \frac{\sum_{\mathrm{t}} R_{t+1}}{T},\ t \in \{0,1,...,T\}. \tag{12}$$

Then we convert the maximization migration throughput problem into finding an optimal policy π^* with maximal average return g in one episode, which can be formulated as follows:

$$\begin{aligned} \max\ & g, \\ \text{s.t.}\ & \sum_t \mu_i(t) < Z, \\ & \sum_t [e_{\mathrm{f}}(t)+e_{\mathrm{h}}(t)+e_{\mathrm{c}}(t)] < B, \\ & i \in \{1,2,...,N\}, \\ & t \in \{0,1,...,T\}. \end{aligned} \tag{13}$$

IV. DRL-Based Scheme of Maximizing User Tasks Migration Throughput

The idea that agent learns by interacting with environment is the key point of RL. In this chapter, we first introduce the traditional RL method about how to obtain optimal sate-action value in MDP. Then, we propose the DRL-based scheme of maximizing user tasks migration throughput who combines RL and NN together to solve our problem.

A. Traditional RL Method

Traditional RL method uses state-action value function Q to organize and structure the search for a good policy π. We can easily obtain the optimal policy π^* once we have found the optimal state-action value function $Q^*(S_t, A_t)$, which satisfies the Bellman optimality equations:

$$\begin{aligned} & Q^*(S_t,A_t) = \\ & \mathbb{E}(R_{t+1} + \omega \max_{a'} Q^*(S_{t+1},a')|S_t = s, A_t = a), \end{aligned} \tag{14}$$

where $\mathbb{E}(\cdot)$ is expectation function and $\omega \in [0,1]$ is decay degree. $S_t = s$ and $A_t = a$ represent current state and action respectively. The reward of current state-action pair is R_{t+1}. $Q*(S_{t+1}, a')$ is the state-action value of next state S_{t+1} and next action a'. The common update mechanism for state-action value in RL algorithms is:

$$\begin{aligned} Q(S_t,A_t) = (1-\alpha)Q(S_t,A_t) + \\ \alpha\left(R_{t+1} + \omega Q(S_{t+1},A_{t+1}) - Q(S_t,A_t)\right), \end{aligned} \tag{15}$$

where learning rate $\alpha \in [0,1]$ and decay degree $\omega \in [0,1]$ decide the dependent percentage of the value of current state-action pair on the next state-action pair.

B. DRL-Based Scheme of Maximizing User Tasks Migration Throughput

The traditional RL method still uses table to store the value of state-action pairs, which has a serious limitation and suffers from the exponential computation complexity due to the explosion of huge state and action spaces. In this part we focus on DRL-based algorithm, which draws support from NN to deal with the curse of huge state and action spaces.

We use a 3-layers NN in our proposed scheme, which inputs state and outputs state-action value. What's more, we use two NNs with same structure but different parameters, ϕ^1 and ϕ^2. One is the evaluation network, whose input is the current state s and output is the prediction of the value $Q_{\text{predict}}(s, a; \phi^1)$. The other one is the target network, whose input is the next state s' and output is the prediction of the value $Q_{\text{target}}(s', a'; \phi^2)$. We use a function approximator to estimate the state-action value. The evaluation network updates its parameter ϕ^1 after every step but the target network updates ϕ^2 after a fixed intervals, maybe 200 steps. The action selection policy is also a little different from normal ε-greedy policy. There is a decrement δ needing to be subtracted from ε in each episode, resulting in different proportion of exploration and exploitation. The pseudo-code is shown in Algorithm 1.

Algorithm 1 DRL-based Scheme of Maximizing User Tasks Migration Throughput

Initial weights ϕ of NN and action-value $Q = 0$;
Initial ε of action selection policy and its decrement δ;
Initial total return $G = 0$ and average return $g = 0$;
Initial $episode = 0$;
repeat
  Initial state s and $t = 0$;
  repeat
    Store the transition(s, a, r, s') in $\mathcal{D}$;
    Sample mini-batch randomly from $\mathcal{D}$;
    Update weights ϕ according to (16),(17);
    Output state-action value $Q(s, a; \phi)$;
    If $\tau > \varepsilon$: Choose a from s randomly;
    else $\tau \leq \varepsilon$: Choose $a = \arg\max_a Q(s, a; \phi)$;
    Take action a, observe r, s';
    Set $G = G + r$;
    Set $t = t + 1$;
  until (UAV battery B is exhausted)
  Set $T = t$ and $g = G/T$;
  Set ε = ε - δ;
  Set $episode = episode + 1$
until $(episode = N_e)$

The number of training episodes is N_e. During each time slot t in every episode, for any given initial state s, UAV chooses action a under ε-greedy policy and prior Q values. Next, the reward r of state-action pair is attained and the transmission (s, a, r, s') is stored in memory $\mathcal{D}$ with size K. Then we calculate state-action value $Q(s, a; \phi)$ to update Q values. At the same time, we randomly extract mini-bath samples from $\mathcal{D}$ to update the weights of NN by the loss function $J(\phi_t)$:

$$J(\phi_t) = \mathbb{E}_{s,a}((Q(s, a; \phi_{t-1}) - Q(s, a; \phi_t))^2), \tag{16}$$

where ϕ_t represents the tth step weight of NN. The method we use is calculating the gradient descent of loss function to update ϕ_t:

$$\begin{aligned}&\nabla_{\phi_t} J(\phi_t) = \\ &\mathbb{E}_{s,a}((Q(s, a; \phi_{t-1}) - Q(s, a; \phi_t))\, \nabla_{\phi_t} Q(s, a; \phi_t)),\end{aligned} \tag{17}$$

where ∇ is gradient function.

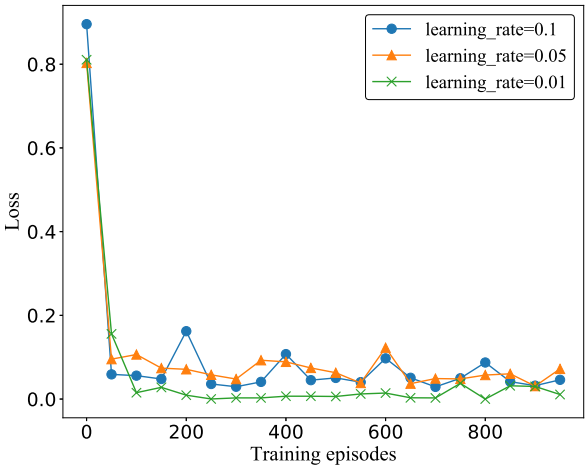

Fig. 2. Loss function $J(\phi)$ of NN about training episodes with different learning rates: $\alpha = 0.1$, $\alpha = 0.05$ and $\alpha = 0.01$.

V. Simulations and Results

We consider a model with $N = 15$ users and $M = 25$ fixed perceptual access points. The number of offloading tasks $\mu_i(t)$ is selected from the set of 15 random numbers between 0 to 10. The number of bits per task N_e is 256 bit. UAV is assumed to fly at a fixed altitude $H = 100$ meter. The Gaussian white noise is $\sigma^2 = -110$ dBm. Most other parameters are from [3]. Specially, we normalize all the values of gains and costs between 0 to 1.

A. Simulations of Loss Function and Training episodes

Fig.2 illustrates the relationship between loss function $J(\phi)$ and training episodes with different learning rates. At the beginning of training process, all state-action values equal to zero because of no prior information. The loss function $J(\phi)$ is determined by reward $R_{t+1} = r$ and learning rate α, which is very large first. Then we have transition (s, a, r, s') to update parameters of NN. $J(\phi)$ decreases rapidly with training and converges to zero after about 100 episodes. The line with $\alpha = 0.01$ has best convergence, so we choose it in our later simulations. But the other two lines all have several little peaks during about 200, 400 and 600 episodes because greater learning rate leads to greater randomness.

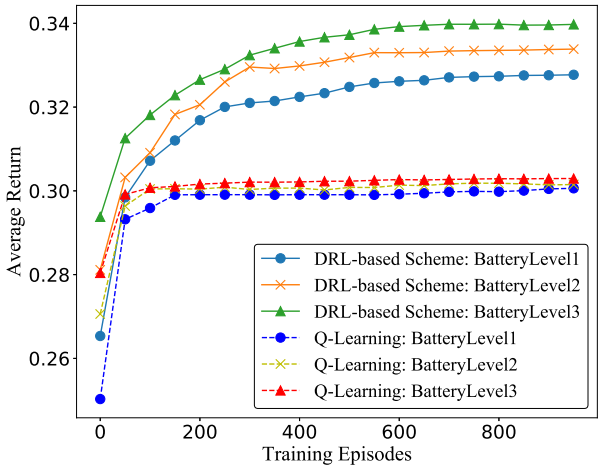

Fig. 3. The relationship between average return g and training episodes with three different battery levels.

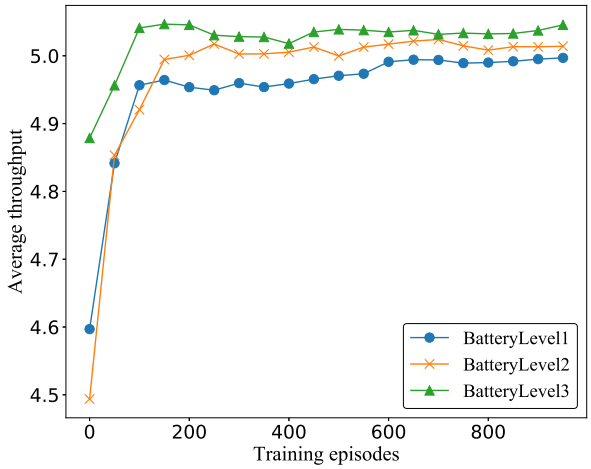

Fig. 4. The relationship between average throughput and training episodes, with the number of bits per task $N_b = 256$ bit.

B. Performance of Average Return and Average Throughput

To better understand the performance of the proposed scheme, we use Q-learning as the benchmark algorithm in Fig.3. A big gap between the solid line and the dashed line means the proposed one has better performance. And Fig.4 shows the relationship between the average throughput and training episodes. The average return g of DRL-based scheme increases first with training and becomes stable between 0.33 and 0.34. The average throughput has same trend with average return, which is lowest firstly and then goes to steady about 5.0. We also consider three different battery levels : BatteryLevel1 is lower than BatteryLevel2 and BatteryLevel2 is lower than BatteryLevel3. They all show an upward trend and the convergence values of them just have a little difference in both Fig.3 and Fig.4. It shows us that the average return and throughput are almost same even with different battery levels, which is same as theoretical analysis. The more battery UAV has, the more times it serves users. But no matter how many times UAV serves users in one episode, the average return and average throughput always have the same upward trend and converges to almost same value.

VI. Conclusions

In this paper, we focus on applying UAV as an MES to provide computational task offloading services and maximizing migration throughput of user tasks. We formulate the maximization problem as an SMDP and solve it with DRL-based scheme of maximizing user tasks migration throughput. So we convert the maximization problem into maximizing the average return to maximize the average throughput of user tasks. The simulation results demonstrate that the proposed scheme is sufficient to guide UAV attain more rewards than previous one until convergence, which means the average migration throughput of user tasks cincreases until converging to maximal values.

References

[1] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, May. 2016.

[2] C. Ting, X. Yun, Z. Xiangmo, G. Tao, and X. Zhigang, "4G UAV communication system and hovering height optimization for public safety," in *IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Oct. 2017, pp. 1–6.

[3] 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, Mar. 2018.

[4] C. Liu, T. Q. S. Quek, and J. Lee, "Secure UAV communication in the presence of active eavesdropper," in *9th International Conference on Wireless Communications and Signal Processing*, Oct. 2017, pp. 1–6.

[5] G. Zhang, Q. Wu, M. Cui, and R. Zhang, "Securing UAV communications via trajectory optimization," in *IEEE Global Communications Conference*, Dec. 2017, pp. 1–6.

[6] D. Yang, Q. Wu, Y. Zeng, and R. Zhang, "Energy trade-off in ground-to-UAV communication via trajectory design," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2018.

[7] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3747–3760, June. 2017.

[8] J. T. Economou, G. Kladis, A. Tsourdos, and B. A. White, "UAV optimum energy assignment using Dijkstra's algorithm," in *European Control Conference*, Jul. 2007, pp. 287–292.

[9] W. H. Al-Sabban, L. F. Gonzalez, and R. N. Smith, "Wind-energy based path planning for unmanned aerial vehicles using Markov decision processes," in *IEEE International Conference on Robotics and Automation*, May. 2013, pp. 784–789.

[10] Z. Yijing, Z. Zheng, Z. Xiaoyi, and L. Yang, "Q-learning algorithm based UAV path learning and obstacle avoidence approach," in *36th Chinese Control Conference*, Jul. 2017, pp. 3397–3402.

[11] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 3, pp. 361–373, Sep. 2017.

[12] Y. Chen, M. Ding, J. Li, Z. Lin, G. Mao, and L. Hanzo, "Probabilistic small-cell caching: Performance analysis and optimization," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 5, pp. 4341–4354, May. 2017.

[13] H. Liu, S. Liu, and K. Zheng, "A reinforcement learning-based resource allocation scheme for cloud robotics," *IEEE Access*, vol. 6, pp. 17 215–17 222, 2018.