Joint Trajectory Planning, Application Placement and Energy Renewal for UAV-Assisted MEC: A Triple-Learner Based Approach

Jialiuyuan Li, Changyan Yi, *Member, IEEE*, Jiayuan Chen, Kun Zhu, *Member, IEEE*,
and Jun Cai, *Senior Member, IEEE*

***Abstract*—In this paper, an energy efficient scheduling problem for multiple unmanned aerial vehicle (UAV) assisted mobile edge computing (MEC) is studied. In the considered model, UAVs act as mobile edge servers to provide computing services to end-users with task offloading requests. Unlike existing works, we allow UAVs to determine not only their trajectories but also the decisions of whether returning to the depot for replenishing energies and updating application placements (due to their limited batteries and storage capacities). With the aim of maximizing the long-term energy efficiency of all UAVs, i.e., the total amount of offloaded tasks computed by all UAVs over their total energy consumption, a joint optimization of UAVs' trajectory planning, energy renewal and application placement is formulated. Taking into account the underlying cooperation and competition among intelligent UAVs, we reformulate such optimization problem as three coupled multi-agent stochastic games. Since the prior environment information is unavailable to UAVs, we propose a novel triple learner based reinforcement learning (TLRL) approach, integrating a trajectory learner, an energy learner and an application learner, for reaching equilibriums. Moreover, we analyze the convergence and the complexity of the proposed solution. Simulations are conducted to evaluate the performance of the proposed TLRL approach, and demonstrate its superiority over counterparts.**

***Index Terms*—Mobile edge computing, UAV, long-term optimization, stochastic game, reinforcement learning**

I. Introduction

Recently, the multi-unmanned aerial vehicle (UAV) assisted mobile edge computing (MEC) [1]–[4] has attracted a myriad of attentions due to its high-flexibility in providing MEC services for end-users (e.g., IoT devices). Particularly, UAVs with computing resources can dynamically adjust their positions to get close to end-users or fly to areas that cannot be covered by fixed MEC infrastructures [5]–[7]. Thus, compared to the traditional MEC, the multi-UAV assisted MEC can provide better quality of service (QoS) for end-users [8], [9].

Although the multi-UAV assisted MEC is envisioned as a light-weight but highly efficient paradigm for alleviating computation burdens on end-users, it also suffers from several inherent restrictions. For instance, computing tasks offloaded from different end-users are required to be processed by specific service applications, while the limited storage capacities of UAVs impede their abilities to store all applications. Additionally, the limited energy capacities of UAVs also hinders the implementation of this paradigm in providing the long-term MEC services. Besides, the amount of IoT devices served by each UAV may be relatively small due to the limitations of each UAV's coverage. Recent research efforts in this area include trajectory optimization [10], [11], service caching [12], UAV deployment [13], [14], etc. Nevertheless, there are still some critical issues, especially how UAVs' installed applications should be updated (under severely restricted wireless backhauls) and how UAVs' energy replenishment [15] should be jointly scheduled, which are imperative but have not yet been well investigated.

i) The limited energy capacities and coverage restrictions of UAVs make it challenging to provide the long-term MEC services for massive IoT devices simultaneously. This prompts us to appropriately schedule the trajectories of multiple UAVs (with different application placements) to collaboratively provide MEC services for IoT devices with different positions and task requests.
ii) Each IoT device's task is required to be supported by a specific application. However, due to the limited storage capacities, UAVs cannot store all required applications for every task. Additionally, UAVs may consume extensive energy when they wirelessly download new huge size applications from remote servers. Moreover, such long-range wireless connectivity may not always be stable. This motivates us to devise a more effective and efficient approach to dynamically update the application placement for multi-UAV assisted MEC.
iii) In multi-UAV assisted MEC, it is required to consider not only the trajectory planning and application placement, but also the energy renewal issue, which is widely recognized in the literature [16]–[21]. Therefore, a joint optimization of these three kinds of decisions considering three optimization problems is indispensable for high energy-efficient multi-UAV assisted MEC.

In this paper, we study a joint optimization of trajectory planning, energy renewal, and application placement for multi-UAV assisted MEC to maximize the long-term energy effi-

J. Li, C. Yi, J. Chen, and K. Zhu are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, 211106, China. (E-mail: {jialiuyuan.li, changyan.yi, jiayuan.chen, zhukun}@nuaa.edu.cn)

J. Cai is with the Department of Electrical and Computer Engineering, Concordia University, Montréal, QC, H3G 1M8, Canada. (E-mail: jun.cai@concordia.ca).

ciency of all UAVs, i.e., the total amount of offloaded tasks computed by all UAVs over their total energy consumption, when providing MEC services. Specifically, in the considered system, each UAV working over a target region has to decide its actions after finishing the last one, i.e., a flight direction for serving IoT devices in other areas or returning back to the depot for replenishing its energy and simultaneously updating its application placement (through wired connections), with the aim of maximizing the long-term energy efficiency of all UAVs. Taking these facts into account in resolving the optimization problem for the multi-UAV assisted MEC is challenging due to the following reasons. Since UAVs are intelligent, we can allow each of them to make its own decisions. However, with the system objective of improving the total energy efficiency of all UAVs, each UAV has to adjust its trajectory planning, energy renewal and application placement strategies with inherent cooperations with other UAVs. Meanwhile, allowing UAVs to make decisions by themselves may also lead to competitions: i) each UAV may selfishly move to a grid with intensive computation requirements, resulting in potential collisions among UAVs; ii) each UAV may prefer to serve more computation offloading requests for its own interests while do not return to the depot for replenishing energy until its battery is exhausted regardless of the others; and iii) each UAV may tend to place applications that are most popular while ignoring the QoS requirements of IoT devices and harming the system performance. Additionally, we consider that the future environment information (e.g., positions and task requirements of IoT devices) is unavailable to UAVs, so that an online optimization is required. To this end, we reformulate the joint optimization problem as three complicated multi-agent stochastic games, i.e., trajectory planning stochastic game (TPSG), application planning stochastic game (APSG) and energy renewal stochastic game (ERSG), for not only comprehensively describing all strategic interactions among UAVs but also facilitating the solution with a refined problem structure. While these three multi-agent stochastic games are coupled tightly, which are still difficult to be solved directly.

In this paper, we design a novel triple-learner based reinforcement learning (TLRL) approach for the multi-UAV assisted MEC, aiming to produce the long-term optimal energy efficient decisions for all UAVs. To characterize all aforementioned features, the stochastic games of trajectory planning, application placement and energy renewal are formulated respectively. By analyzing the characteristics and properties of the problem, a novel TLRL approach is proposed to obtain the corresponding equilibriums of these games. Consequently, the optimal trajectory planning, application placement strategy and energy renewal schedule for multiple UAVs can be derived accordingly.

For clarity, the main contributions of this paper are summarized in the following.

- A joint optimization of trajectory planning, energy renewal and application placement for multi-UAV assisted MEC is formulated, where the objective is to maximize the energy efficiency of all UAVs in the long term.
- Observing the underlying cooperation and competition among UAVs, the optimization problem is reformulated as three coupled multi-agent stochastic games, i.e., TPSG, ERSG and APSG.
- We propose a novel approach, called TLRL, to obtain the equilibriums of the three coupled stochastic games efficiently. Moreover, the convergence of TLRL approach is proved and the complexity of TLRL approach is also analyzed.
- Extensive simulations are conducted to show the superiority of the proposed TLRL approach over counterparts.

The rest of this paper is organized as follows: Section II briefly reviews the related work and emphasizes the novelties of this paper. Section III introduces the system model and problem formulation of the considered multi-UAV assisted MEC. In Section IV, a problem reformulation based on multi-agent stochastic game is constructed. Section V proposes the TLRL approach to optimize trajectory planning, energy renewal and application placement for multiple UAVs. Simulation results are provided in Section VI, followed by the conclusion in Section VII.

II. Related Work

Due to the rapid development of information and communication technologies, UAVs have been widely employed to serve as edge servers for IoT devices in MEC system, and it has attracted a myriad of attentions recently. For instance, Zhang et al. in [22] used UAVs as computing nodes and relay nodes to reduce average user delay. Liao et al. in [23] presented a new UAV-assisted edge computing framework to reduce the computation offloading pressure of users and ground base station. Liu et al. in [24] proposed a new online UAV edge server schedule scheme, which can be used to schedule tasks to appropriate hovering positions by geographically merging tasks into several hot spots. Yu et al. in [25] proposed an innovative MEC system for UAVs involving interaction between IoT devices, UAVs and edge cloud. However, in most of these works, dynamic application placement in multi-UAV assisted MEC was neglected.

The long-term energy efficiency of UAVs is considered as an objective for multi-UAV assisted MEC in many works, with some of them focusing on trajectory planning, energy renewal or application placement. In terms of trajectory planning, Wang et al. in [26] proposed a multi-agent deep reinforcement learning based trajectory control algorithm for managing the trajectory of each UAV to jointly optimize the geographical fairness among all the user equipments (UEs), the fairness of each UAV's UE-load and the overall energy consumption. Besides, Xu et al. in [27] optimized three-dimensional (3D) UAV trajectories to minimize the average weighted sum energy consumption. In terms of the energy renewal, Chen et al. in [28] developed a mixed-integer programming (MIP) model and an equivalent mixed-integer linear programming (MILP) model for UAV-enabled MEC to minimize both the total energy consumption and service time. Furthermore, Wang et al. in [29] jointly designed UAVs' path planning over users' locations and charging stations for providing high-quality MEC services. In terms of the application placement, as far as we know, there are only some existing works on content dissemination [30]–[32], [33], while few works mentioned dynamic application placement on UAVs. Moreover, the long-term energy efficiency optimization of UAVs often involves multiple decision variables, which can be formulated as a joint optimization problem. However, no one has considered the joint optimization of trajectory planning, energy renewal and application placement simultaneously.

For addressing the system uncertainties, the stochastic game has been widely employed in the scheduling problem of multi-UAV assisted MEC. Seid et al. in [34] described the problem as an extension of Markov decision process (MDP) for stochastic games to minimize computing costs of energy and delay in the long-term. Ning et al. in [35] decomposed the problem of system computing cost minimization under dynamic environment by formulating two stochastic games for multi-user computing offload and edge server deployment respectively. Chen et al. in [12] represented the competitive interaction of scheduling local and remote task computing as a stochastic game between mobile users. However, these works commonly considered one stochastic game for all, which may not be efficient when decisions are multi-dimensional and tightly coupled.

In summary, unlike all existing works, this paper delves into the following issues specified to multi-UAV assisted MEC.

- Instead of assuming that UAVs' application deployment are fixed or almost never updated, we study a novel model such that UAVs are allowed to dynamically fly back to the depot for updating their application placement.
- We consider a joint optimization problem of trajectory planning, energy renewal and application placement for multi-UAV assisted MEC, which has never been discussed in the literature
- We first reformulate the joint optimization problem as three coupled multi-agent stochastic games, and then propose a novel triple-learner based approach which can efficiently reach the corresponding equilibriums.

III. System Model and Problem Formulation

In this section, the system model of the considered multi-UAV assisted MEC is first described. Then, the task offloading model, UAV computation model, UAV propulsion energy model and UAV energy renewing model are formulated in detail. All important notations are listed in Table I.

A. Overview

Consider a multi-UAV assisted MEC deployed in a target region, as illustrated in Fig. 1, consisting of a group of UAVs (acting as mobile edge servers) $\mathcal{M}$ with cardinality of $|\mathcal{M}| = M$ and a set of randomly scattered IoT devices $\mathcal{N}$ with cardinality of $|\mathcal{N}| = N$. There is a depot located at the edge of the target region, which can be used by UAVs for both energy replenishment and updating application placement through wired connections. A time-slotted operation framework is studied, in which we define $t \in \{1, 2, ..., T\}$ as the index of time slot. We consider that all IoT devices randomly generate their offloading tasks in each time slot, and the type of each task is also generated randomly. The target region is equally divided into small squared grids with the side length of q. Similar to [36], we assume that the downlink transmission range of each UAV is $\frac{\sqrt{2}}{2}q$, which totally covers a grid (for feeding back computation outcomes). At any time slot, each grid can only be covered by one UAV serving IoT devices to avoid collisions. Each IoT device is associated with a certain UAV hovering located at the same grid for offloading tasks through wireless communications[1], while each

TABLE I
IMPORTANT NOTATIONS IN THIS PAPER

Symbol	**Meaning**
$\mathcal{M}$	Set of UAVs
$\mathcal{N}$	Set of IoT devices
$\mathcal{U}$	Positions set of UAVs
$\mathcal{I}$	Positions set of IoT devices
T	The amount of time slot
m	Index of $\mathcal{M}$
n	Index of $\mathcal{N}$
t	Index of time slot
q	Length of the squared grid
V	Velocity of each UAV
C	The amount of the task types
c	Index of the task types
x_m, y_m	X-axis and the Y-axis of UAV m
x_n, y_n	X-axis and the Y-axis of IoT device n
$d_{m,n}$	Distance between IoT device n and UAV m
H	Flight altitude of each UAV
$\mathcal{G}_m$	Set of IoT devices served by UAV m
a, b	Environmental constants
f, c^{light}	Carrier frequency and the speed of light
η_{LoS}, η_{NLoS}	Losses corresponding to the LoS and non-LoS
δ	LoS probability
γ	SINR of the channel
μ	Instantaneous achievable rate
$\boldsymbol{w}$	Applications placed in UAVs
$\boldsymbol{v}$	Task requests of IoT devices
f_m^U	Computing capacity of UAV m
p_n^{tran}	Transmission power of IoT device n
ϖ	Power spectral density of noise
$\lambda_{m,n}$	Path loss between IoT device $n \in \mathcal{G}_m$ and UAV m
ε	Indicator of UAVs renewing energy
B	Channel bandwidth of IoT device n to UAV m
D_n	Size of each task offloaded by IoT device n
ξ	Effective capacitance coefficient of each UAV
P_m^{pro}	Propulsion power of UAV m
t^{hover}	Hovering time of each UAV
t_m^{renew}	Renewing time of UAV m
$t_{m,n}^{off}$	Offloading time of IoT devices n to UAV m
t_m^{comp}	Computing time of UAV m
$Task_m^{comp}$	Data size of tasks computed by UAV m
S_m	Maximum amount of applications of UAV m
E_m^{total}	Energy capacity of UAV m.
E_m^{comp}	Energy consumption of tasks computed by UAV m
E_m^{pro}	Propulsion energy consumption of UAV m
E_m^{return}	Energy consumption of UAV m returning to the depot
E_m^{remain}	Remaining energy of UAV m
E^{effi}	Energy efficiency of all UAVs

[1] In the future, we will further consider two scenarios below if this setting is relaxed. 1) Each IoT device is associated with a UAV located at the other grid, then its task results may be transmitted back through several relay nodes. 2) Each IoT device is associated with multiple UAVs which are located at the same grid, then the altitudes of these UAVs may also need to be carefully optimized to avoid collisions.

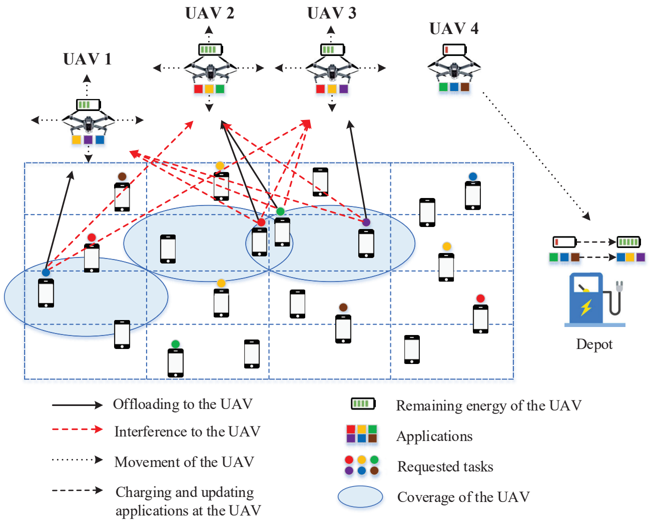

Fig. 1. An illustration of the considered multi-UAV assisted MEC.

UAV could provide multiple IoT devices with MEC service within its downlink transmission range. Since all IoT devices are required to offload their tasks to their associated UAVs via uplink communications using the same frequency band B, each IoT device will interfere other UAVs when the IoT device is offloading a task to its associated UAV. The set of IoT device served by UAV m can be denoted as $\mathcal{G}_m$. At the beginning of each time slot t, every UAV decides whether to return to the depot according to the state information of all UAVs. Specifically, if a UAV does not return to the depot, it will independently select a direction among forward, backward, left and right, and move to the center of another adjacent grid with a constant velocity V. Then the UAV hovers over this grid within the slot to receive and compute tasks from IoT devices. Note that all tasks are considered to be delay-sensitive [37], and each UAV only receives tasks which can be computed by applications that has already been placed in the UAV. Additionally, since the size of results is much smaller than offloaded tasks size, the delay and energy consumption for results sending back from a UAV to IoT devices are omitted in this paper. In contrast, if a UAV chooses to return to the depot, it will renew energy in the depot. Besides, the UAV will also update its application placement in the depot for better serving IoT devices. After these two processes (i.e., renewing energy and updating application placement), the UAV will fly back to the original region it served.

B. Offloading Model

In this subsection, we study the offloading model of IoT devices in the considered multi-UAV assisted MEC. Let $\mathcal{U}_m(t) = (x_m^U, y_m^U)$ and $\mathcal{I}_n = (x_n^I, y_n^I)$ denote the position of UAV m and the position of IoT device n in horizontal coordinates at time slot t, respectively. Then, The distance between IoT device n and UAV m at time slot t can be mathematically expressed as

$$d_{m,n}(t) = \sqrt{(x_m^U - x_n^I)^2 + (y_m^U - y_n^I)^2 + H^2}, \qquad (1)$$

where H denotes a fixed flight altitude of each UAV. Following the literature [38], the line-of-sight (LoS) probability between IoT device $n \in \mathcal{G}_m$ and UAV $m \in \mathcal{M}$ at time slot t is given by

$$\delta_{m,n}(t) = a \cdot exp(-b(arctan(H/d_{m,n}(t)) - a)), \qquad (2)$$

where a and b are constant values depending on the environment. Then, the path loss between IoT device $n \in \mathcal{G}_m$ and UAV $m \in \mathcal{M}$ and at time slot t can be expressed as

$$\begin{aligned} \lambda_{m,n}(t) = \; & 20log(\sqrt{H^2 + d_{m,n}(t)^2}) \\ & + \delta_{m,n}(t)(\eta_{LoS} - \eta_{NLoS}) \\ & + 20log[(4\pi f)/c^{light}] + \eta_{NLoS}, \end{aligned} \qquad (3)$$

where f and c^{light} signify the carrier frequency and the speed of light, respectively; η_{LoS} and η_{NLoS} are the losses corresponding to the LoS and non-LoS, respectively.

Since a common frequency band is reused among all links, the signal-to-interference-plus-noise ratio (SINR) at UAV $m \in \mathcal{M}$ with regard to the uplink communication of IoT device $n \in \mathcal{G}_m$ at time slot t can be written as

$$\gamma_{m,n}(t) = \frac{\boldsymbol{v}_n(t)\boldsymbol{w}_m(t)^\top p_n^{tran} 10^{\frac{-\lambda_{m,n}(t)}{10}}}{\sum_{i=1\setminus\{n\}}^{N} \boldsymbol{v}_n(t)\boldsymbol{w}_m(t)^\top p_i^{tran} 10^{\frac{-\lambda_{m,n}(t)}{10}} + \varpi}, \qquad (4)$$

where p_n^{tran} is the transmission power of IoT device n, and ϖ indicates the power spectral density (PSD) of noise. Note that, in this work, we assume that all tasks are independent with each other, while we may follow [39] to extend the model for accommodating a more complicated scenario with task dependency. At time slot t, we consider that IoT device $n \in \mathcal{G}_m$ can offload no more than one task to its associated UAV m. Let $\boldsymbol{v}_n(t) = \{v_{n,1}(t), v_{n,2}(t), ..., v_{n,C}(t)\}$, where $c \in \{1, 2, ..., C\}$ is the index of the type of task, and $v_{n,c}(t) = 1$ signifies that IoT device n requests to offload task c, and $v_{n,c}(t) = 0$, otherwise. Meanwhile, the applications placed in UAV m can be defined as $\boldsymbol{w}_m(t) = \{w_{m,1}(t), w_{m,2}(t), ..., w_{m,C}(t)\}$, where $w_{m,c}(t) \in \{0, 1\}$ signifies whether UAV m places the application or not, and $w_{m,c}(t) = 1$ means that UAV m places the application which can compute task c, and $w_{m,c}(t) = 0$, otherwise. This implies that the type of each task can be only computed by one type of application.

Following the path loss model [38], the instantaneous achievable rate of IoT device n offloading tasks to UAV m at time slot t can be expressed as

$$\mu_{m,n}(t) = Blog_2(1 + \gamma_{m,n}(t)). \qquad (5)$$

Note that, any UAV $m \in \mathcal{M}$ can only process the types of tasks fitting the types of its placed applications. Based on these, the time of IoT device $n \in \mathcal{G}_m$ offloading to UAV m at time slot t can be written as

$$t_{m,n}^{off}(t) = \frac{\boldsymbol{v}_n(t)\boldsymbol{w}_m(t)^\top D_n}{\mu_{m,n}(t)}, \qquad (6)$$

where D_n is the size of each task offloaded by IoT device n.

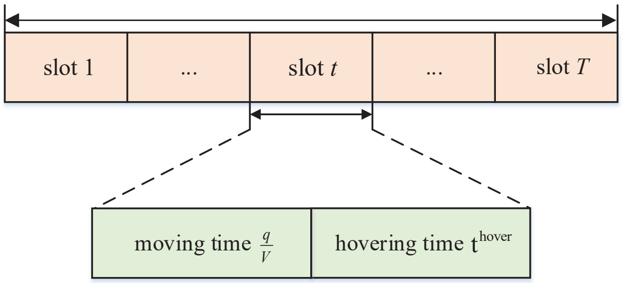

Fig. 2. An illustration of task offloading and computation protocol for UAV.

C. UAV Computation Model

As shown in Fig. 2, we consider a task offloading and computation protocol for each UAV m as follows. During UAV m providing MEC service for IoT devices $n \in \mathcal{G}_m$, the duration of each time slot t can be decomposed into the UAV moving time $\frac{q}{V}$ and the UAV hovering time t^{hover}. Within each time slot t, we ask UAV $m \in \mathcal{M}$ hovers over the center of a certain grid to provide MEC services with time duration t^{hover}, and $t_{m,n}^{off}(t) < t^{hover} < |t|, \forall n \in \mathcal{G}_m, m \in \mathcal{M}$, which means that t^{hover} is large enough for UAV m to receive any task offloaded by any IoT device and is shorter than the duration of a time slot. Then, the time of tasks computed by UAV $m \in \mathcal{M}$ can be expressed as

$$\begin{aligned} t_m^{comp}(t) = \quad & \min\{\textstyle\sum_{n \in \mathcal{G}_m} \boldsymbol{v}_n(t)\boldsymbol{w}_m(t)^\top \frac{D_n}{f_m^U}, \\ & t^{hover} - \min\{t_{m,n}^{off}(t)\}_{n \in \mathcal{G}_m}\}, \end{aligned} \tag{7}$$

where f_m^U is the computing capacity of UAV m (in the number of CPU cycles per second), and $t^{hover} - \min\{t_{m,n}^{off}(t)\}_{n \in \mathcal{G}_m}$ indicates that UAV m starts edge computing since the first task is totally received. The size of tasks computed by UAV m can be written as

$$Task_m^{comp}(t) = t_m^{comp}(t) f_m^U, \tag{8}$$

Correspondingly, the energy consumption of UAV $m \in \mathcal{M}$ for computing tasks at time slot t is calculated as

$$E_m^{comp}(t) = \xi (f_m^U)^2 Task_m^{comp}(t), \tag{9}$$

where ξ denotes effective capacitance coefficient of each UAV.

D. UAV Propulsion Energy Model

In this paper, we consider a rotary-wing UAV propulsion power model which depends on the velocity V [40]. For rotary-wing UAVs, the propulsion power of UAV m is

$$\begin{aligned} & P_m^{pro}(V) = \\ & \tfrac{1}{2}(\tfrac{S_f}{R_s A})\rho R_s A V^3 + (\tfrac{\varphi_e}{8}\rho R_s A \Omega_e^3 R_e^3)(1 + \tfrac{3V^2}{(\Omega_e R_e)^3}) \\ & +((1+\kappa_p)\tfrac{(gM_{UAV})^{\frac{3}{2}}}{\sqrt{2\rho A}})(\sqrt{1 + \tfrac{V^4}{4(\sqrt{\frac{gM_{UAV}}{2\rho A}})^2}} - \tfrac{V^2}{2(\sqrt{\frac{gM_{UAV}}{2\rho A}})})^{\frac{1}{2}}, \end{aligned} \tag{10}$$

where the descriptions of parameters in (10) are listed in Table II, and their settings are accordingly [40]. Then, the propulsion energy consumption of UAV m can be expressed as $E_m^{pro} = P_m^{pro}(V)\frac{q}{V} + P_m^{pro}(0)t^{hover}$, which indicates that E_m^{pro} consists of the horizontal moving energy and the hovering energy at each time slot t.

E. UAV Energy and Application Placement Model

Let E_m^{total}, $E_m^{return}(t)$ and $E_m^{remain}(t)$ be the energy capacity of UAV m, the energy consumption of UAV m returning to the depot and the remaining energy of UAV m at the end of time slot t, respectively. $E_m^{return}(t)$ can be written as

$$E_m^{return}(t) = P_m^{pro}(V)\frac{d_m^{return}(t)}{V}, \tag{11}$$

where $d_m^{return}(t)$ indicates the distance between UAV m and depot at time slot t. The energy consumption of UAVs consists of computing energy, propulsion energy and returning energy. If UAV m chooses to return to the depot, it will renew energy to support continuous MEC service. Hence, the remaining energy of UAV m at time slot t can be formulated as

$$\begin{aligned} & E_m^{remain}(t) = E_m^{remain}(t-1) - \varepsilon_m(t)(E_m^{pro} + E_m^{comp}(t)) \\ & -(1-\varepsilon_m(t))E_m^{return}(t) \end{aligned} \tag{12}$$

where $\varepsilon_m(t) \in \{0, 1\}$ stands for the decision that whether UAV $m \in \mathcal{M}$ chooses to whether return to the depot at the beginning of each time slot t, and $\varepsilon_m(t) = 0$ means that UAV m decides to return to the depot at time slot t, and $\varepsilon_m(t) = 1$ otherwise. Since the energy renewal time is much longer than the flight time between the original region and the depot, the flight time is omitted. Additionally, to guarantee the QoS of IoT devices, each type of application should be placed in at least one UAV hovering over the target region at each time slot t, i.e.,

$$\sum_{m=1}^{M} w_{m,c}(t)\varepsilon_m(t) \geq 1, \forall c \in C. \tag{13}$$

After replenishing its energy and updating its application placement, UAV m will fly back to the original region it was located and continue to provide MEC services. Note that, the total size of applications placed at UAV $m \in \mathcal{M}$ should be smaller than its storage capacity, which can be written as

$$\sum_{c=1}^{C} w_{m,c}(t) \leq S_m, \tag{14}$$

where S_m indicates the maximum amount of applications placed in UAV m.

F. Problem Formulation

We aim to solve the problem of joint optimization of multiple UAVs' trajectory planning, energy renewal and application placement, with the aim of maximizing the energy efficiency of all UAVs, i.e., the total amount of offloaded tasks computed by all UAVs over their total energy consumption, which can be mathematically expressed as

$$E^{effi}(t) = \sum_{m=1}^{M} \frac{\varepsilon_m(t) Task_m^{comp}(t)}{|E_m^{remain}(t-1) - E_m^{remain}(t)|}. \tag{15}$$

TABLE II
UAV PROPULSION ENERGY MODEL

Parameter	Descriptions
φ_e	Blade drag coefficient
Ω_e	Blade angular velocity
R_e	Rotor radius
ρ	Air density
κ_p	Induced power factor
R_s	Rotor solidity
g	Gravity acceleration
A	Rotor disc area
M_{UAV}	UAV mass
S_f	Fuselage equivalent flat plate area

Then, the joint optimization of multiple UAVs' trajectory planning, energy renewal and application placement can be formulated as

$$[\mathcal{P}1]: \max_{\mathcal{U}_m(t),\boldsymbol{w}_m(t),\varepsilon_m(t)} \lim_{T\to+\infty} \frac{1}{T}\sum_{t=1}^{T} E^{effi}(t) \tag{16}$$

$$s.t., (13), (14),$$

$$\sum_{c=1}^{C} v_{n,c} \leq 1, \tag{17}$$

$$\varepsilon_m(t) \in \{0,1\}, \forall m \in \mathcal{M}, \tag{18}$$

$$w_{m,c}(t) \in \{0,1\}, \forall m \in \mathcal{M}, \forall c \in C, \tag{19}$$

$$E_m^{remain}(0) = E_m^{total}, m \in \mathcal{M} \tag{20}$$

$$|\mathcal{U}_m(t) - \mathcal{U}_m(t-1)|^2 \varepsilon_m(t) = q^2, m \in \mathcal{M}, \tag{21}$$

$$(x_m^U(t) - x_m^U(t-1))(y_m^U(t) - y_m^U(t-1))\kappa_m(t) = 0, \tag{22}$$

$$|\mathcal{U}_m(t) - \mathcal{U}_{m'}(t)|^2 \geq q^2, m' \in \mathcal{M}\backslash\{m\}, \tag{23}$$

where constraint (17) means that each IoT device requests to offload at most one task at eash time slot; constraint (20) indicates that the initial energy of each UAV equals its energy capacity; constraint (21) and constraint (22) imply that each UAV can only move to the center of adjacent grid if it does not return to the depot; constraint (23) indicates that each grid can only be covered by one UAV to avoid potential collisions. In the following section, we will first analyze this optimization problem, and then propose a novel approach to derive the corresponding solution.

IV. PROBLEM REFORMULATION BASED ON MULTI-AGENT STOCHASTIC GAME

In this section, we show how the optimization problem $[\mathcal{P}1]$ can be reformulated based on multi-agent stochastic game.

A. Game Statement

Since UAVs are intelligent, to solve problem $[\mathcal{P}1]$, we can allow each UAV to make its own decisions while regulate the underlying cooperation and competition among them. Specifically, UAVs are expected to cooperatively conduct the trajectory planning, energy renewal and application placement to maximize the energy efficiency of all UAVs while guaranteeing QoS of IoT devices. Meanwhile, allowing UAVs to make decisions by themselves may also lead to competitions as follows.

1) For trajectory planning, each UAV would maximize its energy efficiency by moving to a grid with intensive computation requirements, which may result in collisions among UAVs.
2) For energy renewal, each UAV may prefer to serve more computation offloading requests in maximizing its energy efficiency while do not return to the depot for replenishing energy until its battery is exhausted, causing constraint (13) to collapse.
3) For application placement, each UAV tends to place applications that are requested most frequently to maximize its own energy efficiency, while ignoring the QoS requirements of IoT devices (making some of them starving).

Additionally, considering the uncertainty that the future environment information (e.g., task requirements of IoT devices) is not available to UAVs, to this end, we reformulate the joint optimization problem $[\mathcal{P}1]$ as three coupled multi-agent stochastic games as follows.

B. Game Formulation

Firstly, we define the multi-agent stochastic game as a tuple $\langle\mathcal{M},\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R}\rangle$ [10] in view of the discussion above.
1) $\mathcal{M}$ indicates the set of agents.
2) $\mathcal{S}$ indicates the set of environment states. At time slot t, the environment state is denoted as $s(t)$.
3) $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_M\}$ indicates the set of joint action, where $\mathcal{A}_m$ represents the set of individual actions of agent m. The joint action at time slot t is denoted as $\boldsymbol{a}(t) \in \mathcal{A}$, while the individual action of agent m is denoted as $a_m(t) \in \mathcal{A}_m$. Hence, the joint action can be written as $\boldsymbol{a}(t) = \{a_1(t), ..., a_M(t)\}$.
4) $\mathcal{P}$ indicates the set of state transition probabilities. $\mathcal{P}_{ss'}(\boldsymbol{a}(t))$ signifies the state transition probability from state s to s' by taking the joint action $\boldsymbol{a}(t) \in \mathcal{A}$.
5) $\mathcal{R} = \{\mathcal{R}_1, ..., \mathcal{R}_M\}$ indicates the reward function, where $\mathcal{R}_m(t)$ signifies the set of immediate reward of agent m at time slot t.

As mentioned above, problem $[\mathcal{P}1]$ can be reformulated as three coupled multi-agent stochastic games, namely, TPSG $\langle\mathcal{M},\mathcal{S}^{TPSG},\mathcal{A}^{TPSG},\mathcal{P}^{TPSG},\mathcal{R}^{TPSG}\rangle$, ERSG $\langle\mathcal{M},\mathcal{S}^{ERSG},\mathcal{A}^{ERSG},\mathcal{P}^{ERSG},\mathcal{R}^{ERSG}\rangle$ and APSG $\langle\mathcal{M},\mathcal{S}^{APSG},\mathcal{A}^{APSG},\mathcal{P}^{APSG},\mathcal{R}^{APSG}\rangle$, Particularly, for TPSG, each UAV $m \in \mathcal{M}$ will choose an action individually based on the current environment states $s^{TPSG}(t) \in \mathcal{S}^{TPSG}$ at the beginning of each time slot t, and then form a joint action $\boldsymbol{a}^{TPSG}(t) \in \mathcal{A}^{TPSG}$. After executing the joint action, rewards will be obtained according to $\mathcal{R}^{TPSG}$, and the environment states will turn to be next ones following $\mathcal{P}^{TPSG}$. The descriptions of ERSG and APSG are similar to TPSG, and are omitted here for conciseness.

Note that, TPSG, ERSG and APSG are inherently coupled. To be more specific, the joint action of ERSG $\boldsymbol{a}^{ERSG}(t) \in \mathcal{A}^{ERSG}$ at time slot t decides UAVs to whether return to the depot for energy replenishment, and such decisions affect the joint action of TPSG $\boldsymbol{a}^{TPSG}(t) \in \mathcal{A}^{TPSG}$ or APSG $\boldsymbol{a}^{APSG}(t) \in \mathcal{A}^{APSG}$ at time slot t in selecting which direction to move or which applications to update. In turn, the joint action of TPSG $\boldsymbol{a}^{TPSG}(t) \in \mathcal{A}^{TPSG}$ at time slot t affect the joint action of ERSG $\boldsymbol{a}^{ERSG}(t) \in \mathcal{A}^{ERSG}$ at the next time slot $t+1$, because the remaining energy of UAVs depends on UAV' propulsion energy consumption and the amount of tasks they processed at the time slot t. Additionally, they also affect the joint action of APSG $\boldsymbol{a}^{APSG}(t+t') \in \mathcal{A}^{APSG}$ in future time slots $t+t'$, $t'>0$, because different trajectories lead to different histories of providing MEC services, which influences UAV' decisions in updating which types of applications. Besides, the joint action of APSG $\boldsymbol{a}^{APSG}(t+t') \in \mathcal{A}^{APSG}$ at time slot t would affect the joint action of TPSG $\boldsymbol{a}^{TPSG}(t+t') \in \mathcal{A}^{TPSG}$ at the next time slot $t+1$, because the trajectory planning has to take into account the types of tasks that UAVs can process. In the following subsection, we propose a novel approach, called TLRL, to obtain equilibriums of these three coupled multi-agent stochastic games.

V. TLRL Approach

It is worth noting that, the transitions of states and actions of TPSG, ERSG, and APSG satisfy the Markov property because all joint action, i.e., flight directions $\boldsymbol{a}^{TPSG}(t)$, decisions of whether returning to the depot $\boldsymbol{a}(t)^{ERSG}$, and decisions of which applications to place $\boldsymbol{a}(t)^{APSG}$, at time slot t only depend on the environment states at time slot t, i.e., $\boldsymbol{s}^{TPSG}(t)$, $\boldsymbol{s}^{ERSG}(t)$ and $\boldsymbol{s}^{APSG}(t)$, and thereby, in this paper, we characterize each UAV's strategic decision process in TPSG, ERSG and APSG by three respective Markov decision processes (MDPs).

MDP for each UAV in TPSG: With the aim of finding the optimal trajectories for all UAVs, the individual decision making problem for each UAV $m \in \mathcal{M}$ in TPSG can be modelled as an MDP $(\mathcal{S}^{TPSG}, \mathcal{A}_m^{TPSG}, \mathcal{R}_m^{TPSG}, \mathcal{P}^{TPSG})$.

1) Environment State for Each UAV in TPSG: The environment state $s^{TPSG}(t) \in \mathcal{S}^{TPSG}$ for each UAV $m \in \mathcal{M}$ in TPSG at time slot t consists of all UAVs' positions $\mathcal{U}_m(t)$, $m \in \mathcal{M}$ and application placement $\boldsymbol{w}_m(t)$, $m \in M$, which can be expressed as $s^{TPSG}(t) = (\mathcal{U}_m(t), \boldsymbol{w}_m(t))_{m\in\mathcal{M}}$.

2) Action for Each UAV in TPSG: At time slot t, UAV $m \in \mathcal{M}$ chooses an action $a_m^{TPSG}(t) \in \mathcal{A}_m^{TPSG}$, where $\mathcal{A}_m^{TPSG}$ is the action set of UAV m in TPSG consisting of four possible actions, i.e., moving forward, backward, left or right.

3) Reward of Each UAV in TPSG: The immediate reward of UAV $m \in \mathcal{M}$ at time slot t is given by

$$\mathcal{R}_m^{TPSG}(t) = \frac{\kappa_m(t) Task_m^{comp}(t)}{E_m^{comp}(t) + E_m^{pro}}, \tag{24}$$

where the numerator indicates the size of tasks computed by UAV m at time slot t, and the denominator represents the energy consumption of UAV m at time slot t. The reward function (24) can guide UAVs to provide better MEC services for IoT devices.

4) State Transition Probabilities of UAVs in TPSG: The state transition probability from state s^{TPSG} to state $s^{TPSG'}$ by taking the joint action $\boldsymbol{a}^{TPSG}(t) = (a_1^{TPSG}(t), a_2^{TPSG}(t), ..., a_M^{TPSG}(t))$ can be expressed as $\mathcal{P}_{s^{TPSG}, s^{TPSG'}}^{TPSG}(\boldsymbol{a}^{TPSG}(t)) = Pr(s^{TPSG}(t+1) = s^{TPSG'} | s^{TPSG}(t) = s^{TPSG}, \boldsymbol{a}^{TPSG}(t))$.

MDP for each UAV in ERSG: With the aim of designing the optimal schedule of energy renewal for all UAVs, the individual decision making problem for each UAV $m \in \mathcal{M}$ in ERSG can be modelled as an MDP $(\mathcal{S}^{ERSG}, \mathcal{A}_m^{ERSG}, \mathcal{R}_m^{ERSG}, \mathcal{P}^{ERSG})$.

1) Environment State for Each UAV in ERSG: The environment state $s^{ERSG}(t) \in \mathcal{S}^{ERSG}$ for each UAV $m \in \mathcal{M}$ in ERSG at time slot t consists of all UAVs' remaining energy $E_m^{remain}(t)$, $m \in \mathcal{M}$ and positions $\mathcal{U}_m(t)$, $m \in \mathcal{M}$, which can be expressed as $s^{ERSG}(t) = (E_m^{remain}(t), \mathcal{U}_m(t))_{m\in\mathcal{M}}$.

2) Action for Each UAV in ERSG: At time slot t, UAV $m \in \mathcal{M}$ chooses an action $a_m^{ERSG}(t) \in \mathcal{A}_m^{ERSG}$, where $\mathcal{A}_m^{ERSG}$ is the action set of UAV m in ERSG consisting of two actions, i.e., deciding to return to the depot with $\kappa_m(t) = 0$, and $\kappa_m(t) = 1$ otherwise.

3) Reward of Each UAV in ERSG: The immediate reward of UAV $m \in \mathcal{M}$ at time slot t is given by

$$R_m^{ERSG}(t) = \begin{cases} -10, & \text{if constraint (13) is violated,} \\ \kappa_m(t), & \text{otherwise.} \end{cases} \tag{25}$$

This reward function can prompt UAVs to hover over the target region providing MEC services while avoiding the violation of constraint (13).

The definition of state transition probabilities of UAVs in ERSG $\mathcal{P}^{ERSG}$ is similar to that in TPSG and is omitted here for conciseness.

MDP for each UAV in APSG: With the aim of producing the optimal policy for updating the application placement of all UAVs, the individual decision making problem for each UAV $m \in \mathcal{M}$ in APSG can be defined as an MDP $(\mathcal{S}^{APSG}, \mathcal{A}_m^{APSG}, \mathcal{R}_m^{APSG}, \mathcal{P}^{APSG})$.

1) Environment State for Each UAV in APSG: The environment state $s^{APSG}(t) \in \mathcal{S}^{APSG}$ for each UAV $m \in \mathcal{M}$ in APSG at time slot t consists of applications placed in all UAVs $\boldsymbol{w}_m(t), m \in \mathcal{M}$ and the amount of the task requests from IoT devices covered by UAV m before time slot t, i.e., $\theta_m(t) = \sum_{\tau=1}^{t} \sum_{n\in\mathcal{G}_m} \boldsymbol{v}_n(\tau), m \in \mathcal{M}$, and thus we have $s^{APSG}(t) = (\boldsymbol{w}_m(t), \theta_m(t))_{m\in\mathcal{M}}$.

2) Action for Each UAV in APSG: At time slot t, UAV $m \in \mathcal{M}$ chooses an action $a_m^{APSG}(t) \in \mathcal{A}_m^{APSG}$, where $a_m^{APSG}(t)$ signifies that UAV m selects S_m types of tasks from the total C types of tasks.

3) Reward of Each UAV in APSG: The immediate reward of UAV $m \in \mathcal{M}$ in APSG at time slot t is given by

$$R_m^{APSG}(t) = \frac{e(t)}{C} \sum_{\tau=1}^{t} \sum_{n\in\mathcal{G}_m} \boldsymbol{v}_n(\tau) \boldsymbol{w}_m(\tau)^\top, \tag{26}$$

where $e(t)$ indicates the number of application types placed in all UAVs at time slot t. This reward function would guide UAVs to update more popular but diverse applications according to the history of providing MEC services.

The definition of state transition probabilities of UAVs in APSG $\mathcal{P}^{APSG}$ is similar to that in TPSG and is omitted here for conciseness.

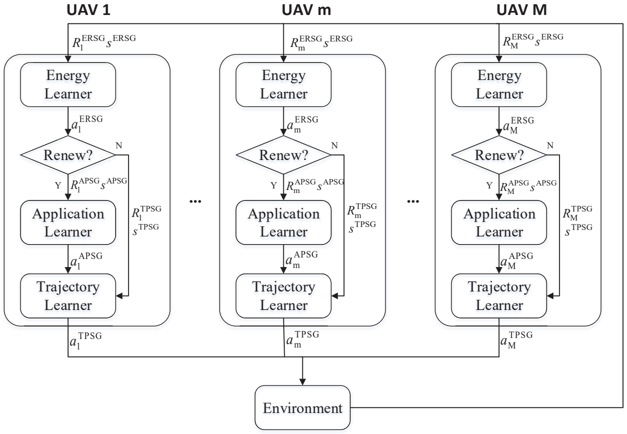
UAV 1
UAV m
UAV M
$R_1^{\mathrm{ERSG}}s^{\mathrm{ERSG}}$
$R_m^{\mathrm{ERSG}}s^{\mathrm{ERSG}}$
$R_M^{\mathrm{ERSG}}s^{\mathrm{ERSG}}$
Energy Learner
a_1^{ERSG}
a_m^{ERSG}
a_M^{ERSG}
Renew?
N
Y
$R_1^{\mathrm{APSG}}s^{\mathrm{APSG}}$
$R_m^{\mathrm{APSG}}s^{\mathrm{APSG}}$
$R_M^{\mathrm{APSG}}s^{\mathrm{APSG}}$
Application Learner
R_1^{TPSG} s^{TPSG}
R_m^{TPSG} s^{TPSG}
R_M^{TPSG} s^{TPSG}
a_1^{APSG}
a_m^{APSG}
a_M^{APSG}
Trajectory Learner
a_1^{TPSG}
a_m^{TPSG}
a_M^{TPSG}
...
Environment

Fig. 3. Illustration of TLRL approach for multiple UAVs trajectory planning, energy renewal and application placement.

Based on the above three MDP formulations, we develop a novel triple learner (i.e., trajectory learner, energy learner and application learner) based reinforcement learning approach to obtain equilibriums of these three coupled multi-agent stochastic games. Specifically, each UAV learns the optimal Q value of each state-action pair, and obtain the optimal local policies for trajectory learner, energy learner, and application learner. The key components of the three coupled stochastic games for the multiple UAVs trajectory planning, energy renewal and application placement studied in this paper is given in Fig. 3. The energy learner learns to design the schedule of returning to renew energy at each time slot. The application learner learns to choose the strategy of updating application placement at each time slot. The trajectory learner learns to choose the direction to move along at each time slot. As shown in Fig. 3, the output of the energy learner determines whether a UAV renew energy or not, which is regarded as one of the inputs of the trajectory learner. The trajectory learner will select a direction that the UAV moves along at each time slot only if the UAV does not renew energy. If the UAV renews energy, the application learner will update application placement, which is regarded as one of the inputs of the trajectory learner. Thus, due to the coupling, these three learners have to run in a back-and-forth manner.

1) Settings for Trajectory Learner: The policy $\pi_m^{TPSG}: \mathcal{S}^{TPSG} \longrightarrow \mathcal{A}_m^{TPSG}$ of the trajectory learner in UAV m, meaning a mapping from the environment state set to the action set, signifies a probability distribution of actions $a_m^{TPSG} \in \mathcal{A}_m^{TPSG}$ in a given state s^{TPSG}. Particularly, for UAV m in state $s^{TPSG} \in \mathcal{S}^{TPSG}$, the trajectory policy of the trajectory learner in UAV m can be presented as $\pi_m^{TPSG}(s^{TPSG}) = \{\pi_m^{TPSG}(s^{TPSG}, a_m^{TPSG}) | a_m^{TPSG} \in \mathcal{A}_m^{TPSG}\}$, where $\pi_m^{TPSG}(s^{TPSG}, a_m^{TPSG})$ is the probability of UAV m selecting action a_m^{TPSG} in state s^{TPSG}.

In Q-learning, the process of building trajectory policy π_m^{TPSG} is significantly affected by trajectory learner's Q function, and the Q function of the trajectory learner in UAV m is the expected reward by executing action $a_m^{TPSG} \in \mathcal{A}_m^{TPSG}$ in state $s^{TPSG} \in \mathcal{S}^{TPSG}$ under the given policy π_m^{TPSG}, which can be expressed by

$$\begin{aligned}&Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, \pi_m^{TPSG}) = \\ &\mathbb{E}(\textstyle\sum_{\tau=0}^{\infty} \sigma^{\tau} \mathcal{R}_m^{TPSG}(t+\tau+1) | s^{TPSG}(t) = s^{TPSG}, \\ &\boldsymbol{a}(t)^{TPSG} = \boldsymbol{a}^{TPSG}, \pi_m^{TPSG}),\end{aligned} \tag{27}$$

where σ is a constant discounted factor with $\sigma \in [0, 1]$, and the value of (27) are termed as action value, i.e., Q value. In (27), we consider the long-term reward of UAV m, namely, the sum of immediate reward at the current time slot.

At the beginning of time slot t, trajectory learner in UAV $m \in \mathcal{M}$ selects an action $a_m^{TPSG}(t) \in \mathcal{A}_m^{TPSG}$ according to its Q function. For striking a balance between exploration and exploitation, in this work, we consider an ϵ-greedy exploration strategy for the trajectory learner. Specifically, the trajectory learner in UAV $m \in \mathcal{M}$ selects a random action $a_m^{TPSG} \in \mathcal{A}_m^{TPSG}$ in state $s^{TPSG} \in \mathcal{S}^{TPSG}$ with probability ϵ and selects the best action a_m^{TPSG*} with probability $(1-\epsilon)$, where the best action has $Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG*}, \pi_m^{TPSG}) \geq Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, \pi_m^{TPSG})$, $\forall \boldsymbol{a}^{TPSG} \in \mathcal{A}^{TPSG}$ with a_m^{TPSG*} being the m-th element of $\boldsymbol{a}^{TPSG*}$. Besides, if the later described energy learner in UAV m selects to return to the depot, the trajectory learner will not choose any action in $\mathcal{A}_m^{TPSG}$. Then, the probability of selecting action $a_m^{TPSG} \in \mathcal{A}_m^{TPSG}$ in state s^{TPSG} can be expressed by

$$\begin{aligned}&\pi_m^{TPSG}(s^{TPSG}, a_m^{TPSG}) \\ &= \begin{cases} 0, \text{if UAV } m \text{ decides to return to the depot}, \\ 1-\epsilon, \text{if } Q_m^{TPSG}(s^{TPSG}, \cdot, \cdot) \text{ of } a_m^{TPSG} \text{ is the highest}, \\ \epsilon, \text{otherwise}. \end{cases}\end{aligned} \tag{28}$$

In the Q value update step of Q-learning, the trajectory learner in each UAV $m \in \mathcal{M}$ follows the update rule:

$$\begin{aligned}&Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, t+1) = \\ &Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, t) + \beta^{TPSG}(\mathcal{R}_m^{TPSG}(t) + \\ &\sigma \max_{\boldsymbol{a}^{TPSG'} \in \mathcal{A}^{TPSG}} Q_m^{TPSG}(s^{TPSG'}, \boldsymbol{a}^{TPSG'}, t) \\ &-Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, t)),\end{aligned} \tag{29}$$

where β^{TPSG} denotes the learning rate in TPSG.

2) Settings for Energy Learner: Similar to the trajectory learner, the policy of energy learner in UAV $m \in \mathcal{M}$ is expressed as $\pi_m^{ERSG}: \mathcal{S}^{ERSG} \longrightarrow \mathcal{A}_m^{ERSG}$, and its definitions are similar to the one in trajectory learner.

Here, the Q function of the energy learner in UAV $m \in \mathcal{M}$ is the expected reward by executing action $a_m^{ERSG} \in \mathcal{A}_m^{ERSG}$ in state $s^{ERSG} \in \mathcal{S}^{ERSG}$ under the given policy π_m^{ERSG}, which can be expressed by

$$\begin{aligned}&Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG}, \pi_m^{ERSG}) = \\ &\mathbb{E}(\textstyle\sum_{\tau=0}^{\infty} \sigma^{\tau} \mathcal{R}_m^{ERSG}(t+\tau+1) | s^{ERSG}(t) = s^{ERSG}, \\ &\boldsymbol{a}(t)^{ERSG} = \boldsymbol{a}^{ERSG}, \pi_m^{ERSG}).\end{aligned} \tag{30}$$

The energy learner in UAV $m \in \mathcal{M}$ selects an action $a_m^{ERSG} \in \mathcal{A}_m^{ERSG}$ (i.e., whether return to the depot or not) also according the ϵ-greedy exploration strategy. Then, the probability of selecting action $a_m^{ERSG} \in \mathcal{A}_m^{ERSG}$ in state

s^{ERSG} can be expressed by

$$\pi_m^{ERSG}(s^{ERSG}, a_m^{ERSG}) = \begin{cases} 1-\epsilon, \text{if } Q_m^{ERSG}(s^{ERSG}, \cdot, \cdot) \text{ of } a_m^{ERSG} \text{ is the highest,} \\ \epsilon, \text{otherwise.} \end{cases} \tag{31}$$

In the Q value update step of Q-learning, the energy learner in each UAV $m \in \mathcal{M}$ follows the update rule:

$$\begin{aligned} & Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG}, t+1) = \\ & Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG}, t) + \beta^{ERSG}(\mathcal{R}_m^{ERSG}(t) + \\ & \sigma \max_{\boldsymbol{a}^{ERSG'} \in \mathcal{A}^{ERSG}} Q_m^{ERSG}(s^{ERSG'}, \boldsymbol{a}^{ERSG'}, t) \\ & -Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG}, t)), \end{aligned} \tag{32}$$

where β^{ERSG} denotes the learning rate in ERSG.

3) Settings for Application Learner: Similar to the trajectory learner, the policy of application learner in UAV $m \in \mathcal{M}$ is expressed as $\pi_m^{APSG} : \mathcal{S}^{APSG} \longrightarrow \mathcal{A}_m^{APSG}$.

Here, the Q function of the application learner in UAV $m \in \mathcal{M}$ is the expected reward by executing action $a_m^{APSG} \in \mathcal{A}_m^{APSG}$ in state $s^{APSG} \in \mathcal{S}^{APSG}$ under the given policy π_m^{APSG}, which can be expressed by

$$\begin{aligned} & Q_m^{APSG}(s^{APSG}, \boldsymbol{a}^{APSG}, \pi_m^{APSG}) = \\ & \mathbb{E}(\textstyle\sum_{\tau=0}^{\infty} \sigma^{\tau} \mathcal{R}_m^{APSG}(t+\tau+1) | s^{APSG}(t) = s^{APSG}, \\ & \boldsymbol{a}(t)^{APSG} = \boldsymbol{a}^{APSG}, \pi_m^{APSG}). \end{aligned} \tag{33}$$

The application learner in UAV $m \in \mathcal{M}$ selects an action $a_m^{APSG} \in \mathcal{A}_m^{APSG}$ also according the ϵ-greedy exploration strategy. Then, the probability of selecting action $a_m^{APSG} \in \mathcal{A}_m^{APSG}$ in state s^{APSG} can be expressed by

$$\pi_m^{APSG}(s^{APSG}, a_m^{APSG}) = \begin{cases} 1-\epsilon, \text{if } Q_m^{APSG}(s^{APSG}, \cdot, \cdot) \text{ of } a_m^{APSG} \text{ is the highest,} \\ \epsilon, \text{otherwise.} \end{cases} \tag{34}$$

In the Q value update step of Q-learning, the application learner in each UAV $m \in \mathcal{M}$ follows the update rule:

$$\begin{aligned} & Q_m^{APSG}(s^{APSG}, \boldsymbol{a}^{APSG}, t+1) = \\ & Q_m^{APSG}(s^{APSG}, \boldsymbol{a}^{APSG}, t) + \beta^{APSG}(\mathcal{R}_m^{APSG}(t) + \\ & \sigma \max_{\boldsymbol{a}^{APSG'} \in \mathcal{A}^{APSG}} Q_m^{APSG}(s^{APSG'}, \boldsymbol{a}^{APSG'}, t) \\ & -Q_m^{APSG}(s^{APSG}, \boldsymbol{a}^{APSG}, t)), \end{aligned} \tag{35}$$

where β^{APSG} denotes the learning rate in APSG.

Each UAV runs three Q-learning algorithms to learn the optimal Q values of each state-action pair. We obtain the optimal local policies according to the trajectory learner, the energy learner and the application learner. In summary, the TLRL approach is detailed illustrated in Algorithm 1.

Initially, all UAVs in set M are chosen to initialize the Q values of TPSG, APSG and ERSG, respectively, and the maximal iteration counter $LOOP$ is set. In each iteration process, the Q values of TPSG, APSG and ERSG are shared among all UAVs. Specifically, each UAV $m \in \mathcal{M}$ decides whether to return to the depot according to policy $\pi_m^{ERSG}(s^{ERSG}, \cdot)$. If UAV m returns to depot, it will update its application placement according to policy $\pi_m^{APSG}(s^{APSG}, \cdot)$. Otherwise, it will choose a direction to fly in the target region according to policy $\pi_m^{TPSG}(s^{TPSG}, \cdot)$. Then, the rewards of TPSG, ERSG and APSG are obtained according to (24), (25) and (26), respectively. Finally, the action-values of TPSG, ERSG and APSG are updated according to (29), (32) and (35), which are shared among all UAVs.

Algorithm 1: TLRL Approach

```
for $m = 1$ to $M$ do
    Initialize Q value $Q_m^{ERSG}(s^{ERSG}, a_m^{ERSG}) = 0$,
      $\forall s^{ERSG} \in \mathcal{S}^{ERSG}$, $a_m^{ERSG} \in \mathcal{A}_m^{ERSG}$ and
      $Q_m^{APSG}(s^{APSG}, a_m^{APSG}) = 0$, $\forall s^{APSG} \in \mathcal{S}^{APSG}$,
      $a_m^{APSG} \in \mathcal{A}_m^{APSG}$ and $Q_m^{TPSG}(s^{TPSG}, a_m^{TPSG}) = 0$,
      $\forall s^{TPSG} \in \mathcal{S}^{TPSG}$, $a_m^{TPSG} \in \mathcal{A}_m^{TPSG}$.
Set the maximal iteration counter $LOOP$ and $loop = 0$.
for $loop < LOOP$ do
    Set $t = 0$.
    for $m = 1$ to $M$ do
        Send $Q_m^{ERSG}$, $Q_m^{APSG}$ and $Q_m^{TPSG}$ to other UAVs.
    while $t \leq T$ do
        Observe state $s$.
        for $m = 1$ to $M$ do
            UAV $m$ selects $a_m^{ERSG}$ according to
              $\pi_m^{ERSG}(s^{ERSG}, \cdot)$.
            if UAV $m$ returns to depot then
                UAV $m$ selects $a_m^{APSG}$ according to
                  $\pi_m^{APSG}(s^{APSG}, \cdot)$.
            else
                UAV $m$ selects $a_m^{TPSG}$ according to
                  $\pi_m^{TPSG}(s^{TPSG}, \cdot)$.
        Obtain the rewards $\mathcal{R}_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG})$,
          $\mathcal{R}_m^{APSG}(s^{APSG}, \boldsymbol{a}_m^{APSG})$ and
          $\mathcal{R}_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG})$.
        Update $Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG})$,
          $Q_m^{APSG}(s^{APSG}, \boldsymbol{a}_m^{APSG})$ and
          $Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG})$
        Send $Q_m^{ERSG}$, $Q_m^{APSG}$ and $Q_m^{TPSG}$ to other UAVs.
        Set $t = t + 1$.
    Set $loop = loop + 1$.
```

The Convergence of TLRL Approach: As recognized in [41], [42], when the limit of Q value $\lim_{t\to\infty} Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, t)$, the limit of Q value $\lim_{t\to\infty} Q_m^{ERSG}(s^{ERSG}, \boldsymbol{a}^{ERSG}, t)$ and the limit of Q value $\lim_{t\to\infty} Q_m^{APSG}(s^{APSG}, \boldsymbol{a}^{APSG}, t)$ converge to the optimal Q value $Q^{TPSG^*}(s^{TPSG}, \boldsymbol{a}^{TPSG})$, $Q^{ERSG^*}(s^{ERSG}, \boldsymbol{a}^{ERSG})$ and $Q^{APSG^*}(s^{APSG}, \boldsymbol{a}^{APSG})$ respectively, the TLRL approach is converged.

***Lemma** 1:* A random iterative process $\Delta^{t+1}(x) = (1 - \lambda(x))\Delta^t(x) + \psi(x)\Phi^t(x)$ converges to zeros with probability 1 under the following conditions:

1)The state space is finite.

2)$\sum_t \lambda^t(x) = \infty$, $\sum_t \psi^t(x) = \infty$, $\sum_t (\lambda(x))^2 = \infty$, $\sum_t (\psi(x))^2 = \infty$ and $E\{\psi(x)|\Lambda^t\} \leq E\{\lambda(x)|\Lambda^t\}$.

3)$||E\{\Phi^t(x)|\Lambda^t\}||_W \leq \zeta ||\Delta^t||_W$, where $\zeta \in (0, 1)$.

4)$Var\{\Phi^t(x)|\Lambda^t\} \leq Z(1 + ||\triangle^t||_W)^2$, where Z is a constant.

Proof: Please refer to Appendix A. ■

***Theorem** 1:* The TLRL approach can achieve

$$\begin{aligned} & \mathbb{P}(lim_{t\to\infty} Q_m^{TPSG}(s^{TPSG}, \boldsymbol{a}^{TPSG}, t) = \\ & Q^{TPSG^*}(s^{TPSG}, \boldsymbol{a}^{TPSG})) = 1, \forall m \in \mathcal{M}, \\ & s^{TPSG} \in \mathcal{S}^{TPSG}, \boldsymbol{a}^{TPSG} \in A^{TPSG}. \end{aligned} \tag{36}$$

TABLE III
SIMULATION PARAMETERS

Parameter	**Value**	**Parameter**	**Value**
Num. of UAVs M	3	UAV altitude H	$120m$
Num. of IoT devices N	300	Bandwidth B	$10MHz$
Num. of task types C	10	Task size D_n	$2Mbits$
Hovering time t^{hover}	$5s$	UAV velocity V	$20m/s$
UAV storage limit S_m	6	Coefficient ξ	10^{-18}
Constant a	9.6117	Constant b	0.1581
Carrier frequency f	$3GHz$	PSD of noise ϖ	$-174dBm/Hz$
Length of a grid q	100m	Target region	$10^3m \times 10^3m$
UAV computing capacity f_m^U	$2Mbps$	UAV propulsion power p_n^{tran}	$0.2W$

Proof: Please refer to Appendix B. ■

***Theorem** 2:* The TLRL approach can achieve

$$\mathbb{P}(lim_{t\to\infty}Q_m^{ERSG}(s^{ERSG},\boldsymbol{a}^{ERSG},t) = Q^{ERSG^*}(s^{ERSG},\boldsymbol{a}^{ERSG})) = 1, \forall m \in \mathcal{M}, s^{ERSG} \in \mathcal{S}^{ERSG}, \boldsymbol{a}^{ERSG} \in A^{ERSG}. \tag{37}$$

Proof: The proof of this theorem is analogous to that of theorem 1, and thus its detailed procedure is omitted. ■

***Theorem** 3:* The TLRL approach can achieve

$$\mathbb{P}(lim_{t\to\infty}Q_m^{APSG}(s^{APSG},\boldsymbol{a}^{APSG},t) = Q^{APSG^*}(s^{APSG},\boldsymbol{a}^{APSG})) = 1, \forall m \in \mathcal{M}, s^{APSG} \in \mathcal{S}^{APSG}, \boldsymbol{a}^{APSG} \in A^{APSG}. \tag{38}$$

Proof: The proof of this theorem is analogous to that of theorem 1, and thus its detailed procedure is omitted. ■

The Complexity of TLRL Approach: Hereafter, we analyze the complexity of the proposed TLRL approach, which is critical in multi-UAV assisted MEC. The complexity of the proposed TLRL approach depends on the size of information exchanged among UAVs when conducting trajectory planning, energy renewal and application placement. In the proposed TLRL approach, the information exchanged among UAVs consists of the state of each UAV, and its size is determined by the size of state space in trajectory learner, energy learner and application learner, respectively. As mentioned above, the size of the state space in trajectory learner is impacted by the number of UAVs, the number of possible positions of each UAV and the number of application types placed in each UAV. The size of the state space of energy learner is impacted by the number of UAVs. The size of the state space of application learner is impacted by the number of UAVs and the amount of application types of each UAV. Therefore, for given multi-UAV assisted MEC, the complexity of the proposed TLRL approach can keep constant with the increase of the density of IoT devices in the target region, meaning that the proposed TLRL approach is scalable.

VI. SIMULATION RESULTS

In this section, simulations are conducted to evaluate the performance of the proposed TLRL approach. Table III lists the values of all simulation parameters, and the propulsion power model follows [40]. Similar settings have also been employed in [36], [43]. Note that some parameters may vary according to different evaluation scenarios. For comparison purpose, we introduce an energy efficient oriented trajectory planning (EOTP) algorithm and an existing algorithm called decentralized multiple UAVs cooperative reinforcement learning (DMUCRL) [9] algorithm as benchmarks:

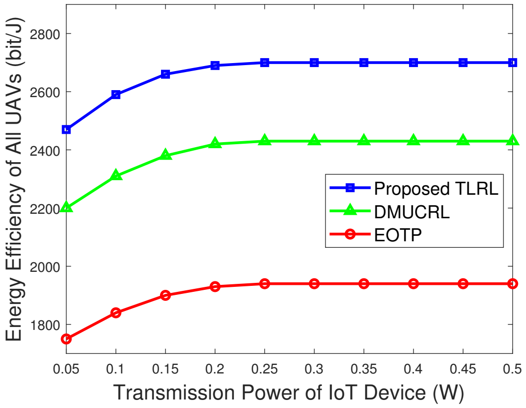

Fig. 4. Comparison on energy efficiency of all UAVs with different IoT device transmission power under DMUCRL EOTP and the proposed TLRL.

- DMUCRL [36]: DMUCRL is originally designed to maximize the energy efficiency of UAVs in downlink content sharing by controlling all UAVs to work collaboratively based on a double Q-learning (where each UAV contains a trajectory learner and an energy learner), while it ignores the management of application placement in UAVs.
- EOTP: EOTP determines the trajectories of all UAVs with the aim of maximizing the energy efficiency but asks UAVs to return to the depot for energy renewal only when their batteries are exhausted, and EOTP does not enable the update of application placement either.

Fig. 4 investigates the energy efficiency of all UAVs with different IoT transmission power under DMUCRL, EOTP and the proposed TLRL. It can be intuitively observed that the energy efficiency of all UAVs first increases and then becomes stable with the increase of IoT devices' transmission power. This is because with the larger transmission power, IoT devices would offload more tasks to their associated UAVs, and thereby increasing the amount of tasks processed by UAVs. However, since the computing capacity of each UAV is still limited, such increasing trend slows down as the limit is approaching. More importantly, this figure shows that the proposed TLRL outperforms both DMUCRL and EOTP. The reason is that i) each UAV under EOTP returns to the depot directly once its energy is exhausted neglecting the QoS requirements of IoT devices; ii) each UAV's applications are fixed placed under DMUCRL, making it capable of serving very limited IoT devices; and iii) our proposed TLRL well addresses the shortcomings of DMUCRL and EOTP by jointly optimizing all UAVs' trajectory planning, energy renewal and application placement.

Fig. 5 illustrates the energy efficiency of all UAVs with different UAV hovering time under DMUCRL, EOTP and the proposed TLRL. It can be observed that, the energy efficiency of all UAVs first increases with the UAV hovering time, and then decreases. This is because with the growth of UAV

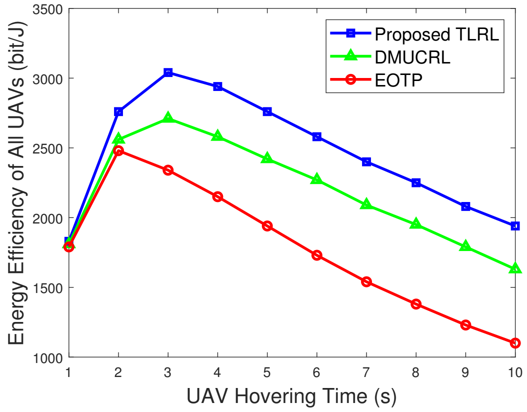

Fig. 5. Comparison on energy efficiency of all UAVs with different UAV hovering time under DMUCRL EOTP and the proposed TLRL.

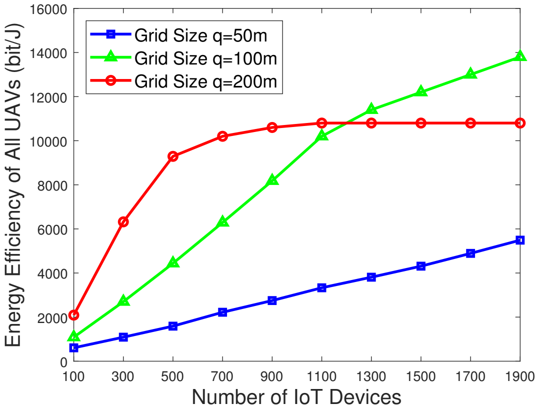

Fig. 7. Comparison on energy efficiency of all UAVs with different number of IoT devices among grid size 50 m, 100 m and 200 m.

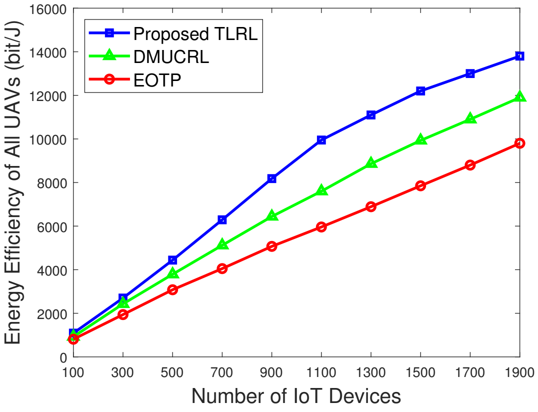

Fig. 6. Comparison on energy efficiency of all UAVs with different number of IoT devices under DMUCRL EOTP and the proposed TLRL.

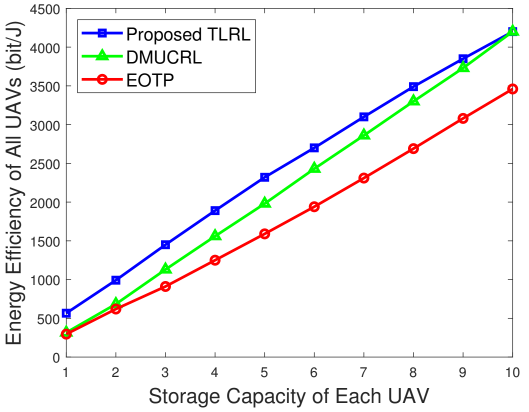

Fig. 8. Comparison on energy efficiency of all UAVs with different storage capacities of each UAV under DMUCRL EOTP and the proposed TLRL.

hovering time, more offloaded tasks from IoT devices can be computed by UAVs during hovering. However, when all tasks have been completely processed by UAVs, they will become idle and consume hovering energy over the target region until hovering time expires. Additionally, it is also shown that the proposed TLRL outperforms both DMUCRL and EOTP, and the explanations for this are similar to those for Fig. 4.

Fig.6 examines the energy efficiency of all UAVs with different number of IoT devices under DMUCRL, EOTP and the proposed TLRL. It can be observed that, the energy efficiency of all UAVs increases monotonically with the number of IoT devices. This is because more offloading requests are generated by IoT devices with the growth of the number of IoT devices, resulting in more tasks are received and computed by UAVs. Moreover, it is also shown that the proposed TLRL outperforms DMUCRL and EOTP, and the explanations for this are similar to those for Fig. 4.

Fig. 7 shows the energy efficiency of all UAVs with different number of IoT devices under different grid size settings (i.e., $q = 50\ m$, $q = 100\ m$ and $q = 200\ m$). We can see that, for grids size 50 m and 100 m, the energy efficiency of all UAVs increases with the number of IoT devices. This is because with the number of IoT devices increasing, more tasks will be offloaded to UAVs, so that more tasks may be processed, resulting in the increase of energy efficiency. Besides, it is also shown that the energy efficiency of all UAVs in grids size 50 m is less than in grids size 100 m and 200 m ones. This is because the larger the grid, the more IoT devices covered by UAVs. Additionally, It can be seen also that, the energy efficiency of all UAVs increases firstly and then becomes stable with the number of IoT devices increasing under grid size 200 m. This is because when UAVs cannot complete all received computation tasks in the hovering time, the energy efficiency of all UAVs will not increase with the number of task requests.

Fig. 8 illustrates the the energy efficiency of all UAVs with different storage capacities of each UAV under DMUCRL, EOTP and the proposed TLRL. It is shown that, the energy efficiency of all UAVs increases monotonically with the storage capacity of each UAV. The reason is that with the increase of storage capacity, more types of applications can be placed in each UAV, so that more tasks may be processed, resulting in the increase of energy efficiency. Additionally, we can also intuitively observe that the proposed TLRL outperforms DMUCRL and EOTP when the storage capacity of each UAV is less than 10 (the amount of task types $C = 10$), and the explanation for this is that the application learner cannot

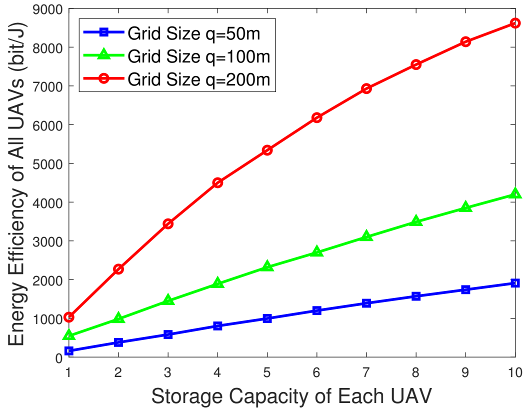

Fig. 9. Comparison on energy efficiency of all UAVs with different storage capacities of each UAV among grid size 50 m, 100 m and 200 m.

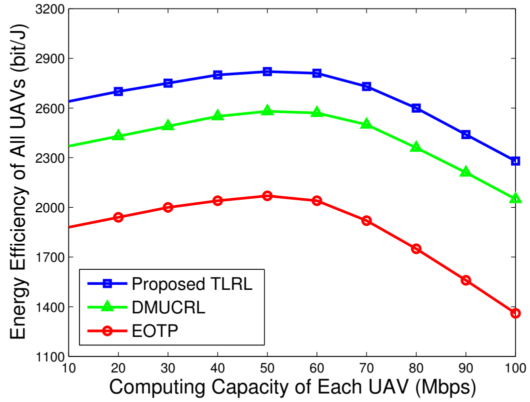

Fig. 10. Comparison on energy efficiency of all UAVs with different computing capacities of each UAV under DMUCRL EOTP and the proposed TLRL.

influence the energy efficiency of all UAVs when each UAV places all types of applications.

Fig. 9 shows all UAV's energy efficiency with different UAV storage capacities under different grid size settings. Specifically, UAVs can adjust their downlink transmission ranges so as to adjust the size q of grids they can cover. It can be seen from Fig. 9 that the larger the grid size is, the higher energy efficiency of all UAVs is obtained. This is because with a larger grid size, more IoT devices are included in a grid, and thereby each UAV can potentially process more offloaded tasks. Besides, it is also shown that the energy efficiency of all UAVs increases monotonically with the storage capacity of each UAV. The reason is that with the increase of storage capacity, more types of applications can be placed in each UAV, so that more tasks may be processed, resulting in the increase of energy efficiency.

Fig. 10 illustrates the energy efficiency of all UAVs with different UAV computing capacities under DMUCRL, EOTP and the proposed TLRL. It can be observed that, the energy efficiency of all UAVs first slowly increases with the UAV computing capacity, and then decreases. This is because with the growth of UAV computing capacity, more offloaded tasks from IoT devices can be computed by UAVs during hovering. However, when all tasks have been completely processed by UAVs, they will become idle and consume hovering energy over the target region until hovering time expires. Moreover, the energy consumption increases with the increase of UAV computing capacity, resulting in that energy efficiency of all UAVs decreases. Additionally, it is also shown that the proposed TLRL outperforms both DMUCRL and EOTP, and the explanations for this are similar to those for Fig. 4.

VII. Conclusion

In this paper, an energy efficient scheduling problem for multi-UAV assisted MEC has been studied. With the aim of maximizing the long-term energy-efficiency of all UAVs, a joint optimization of UAVs' trajectory planning, energy renewal and application placement is formulated. By taking the inherent cooperation and competition among UAVs, we reformulate such optimization problem as three coupled multi-agent stochastic games, and then propose a novel TLRL approach for reaching equilibriums. Moreover, we analyze the convergence and discuss the complexity of the proposed TLRL approach. Simulation results show that, compared to counterparts, the proposed TLRL approach can significantly increase the energy efficiency of all UAVs.

Appendix

A. Proof of Lemma 1

The iteration process of TLRL approach for any state-action pair $(s^{TPSG},\boldsymbol{a}^{TPSG})$ at time slot t can be denoted by $\{Q_m^{TPSG}(s^{TPSG},\boldsymbol{a}^{TPSG},t+1)\}$, which can be written as

$$\begin{aligned}&\overline{Q}^{TPSG}(s^{TPSG},\boldsymbol{a}^{TPSG},t)\\&=\tfrac{1}{M}\textstyle\sum_{m=1}^{M}Q_m^{TPSG}(s^{TPSG},\boldsymbol{a}^{TPSG},t),\forall t\geq 0.\end{aligned}\tag{39}$$

For conciseness, the action and state in the bracket are omitted in this proof, i.e., $Q_m^t=Q_m^{TPSG}(s^{TPSG},\boldsymbol{a}^{TPSG},t)$, $\overline{Q}^t=\overline{Q}^{TPSG}(s^{TPSG},\boldsymbol{a}^{TPSG},t)$, for the immediate reward $\mathcal{R}_m^t=\mathcal{R}_m(s^{TPSG},\boldsymbol{a}^{TPSG},s^{TPSG'},t)$, and the Q value of next time slot $Q_m^{t'}=Q_m^{TPSG}(s^{TPSG'},\boldsymbol{a}^{TPSG'},t)$. Thus, (29) can be rewritten as

$$\begin{aligned}&Q_m^{t+1}=Q_m^t\\&+\beta^{TPSG}(\mathcal{R}_m^t+\sigma\max_{\boldsymbol{a}^{TPSG'}\in\mathcal{A}^{TPSG}}Q_m^{t'}-Q_m^t).\end{aligned}\tag{40}$$

Furthermore, according to (39), we have

$$\begin{aligned}&\overline{Q}^{t+1}=(1-\lambda)\overline{Q}^t\\&+\lambda\tfrac{1}{M}\textstyle\sum_{m=1}^{M}(\mathcal{R}_m^t+\sigma\max\limits_{\boldsymbol{a}^{TPSG'}\in\mathcal{A}^{TPSG}}Q_m^{t'}).\end{aligned}\tag{41}$$

By subtracting Q^* from both sides of (41), we can obtain

$$\begin{aligned}&\overline{Q}^{t+1}-Q^*=(1-\lambda)(\overline{Q}^t-Q^*)\\&+\lambda(\tfrac{1}{M}\textstyle\sum_{m=1}^{M}(\mathcal{R}_m^t+\sigma\max\limits_{\boldsymbol{a}^{TPSG'}\in\mathcal{A}^{TPSG}}Q_m^{t'})-Q^*).\end{aligned}\tag{42}$$

Note that the temporal difference algorithm in (42) can be seen as a random process mentioned in Lemma 1 with $\triangle^{t+1}=\overline{Q}^t-Q^*$, $\Phi^t=\frac{1}{M}\sum_{m=1}^{M}(\mathcal{R}_m^t+\sigma\max\limits_{\boldsymbol{a}^{TPSG'}\in\mathcal{A}^{TPSG}}Q_m^{t'})-Q^*$ and $\lambda=\psi$. Hence, the condition 1) and 2) in Lemma 1 are satisfied. For satisfying the condition 3) and 4) in Lemma 1, we give the proof of the temporal difference algorithm in (42).

According to Proposition 5.1 in [42], we know that $\mathcal{F}(\cdot)$ is a contraction mapping and Q^* is the unique fixed point of operator $\mathcal{F}(\cdot)$, where $\mathcal{F}(\cdot)$ is given by

$$\begin{aligned}\mathcal{F}(Q) = & \sum_{s^{TPSG'} \in \mathcal{S}^{TPSG}} P^{TPSG}_{s^{TPSG} s^{TPSG'}}(\boldsymbol{a}^{TPSG}) \\ & (\tfrac{1}{M}\textstyle\sum_{m=1}^{M} \mathcal{R}^t_m(s^{TPSG}, \boldsymbol{a}^{TPSG}, s^{TPSG'}) \\ & +\sigma \max_{a^{TPSG'} \in \mathcal{A}^{TPSG}} Q(s^{TPSG'}, \boldsymbol{a}^{TPSG'})).\end{aligned} \tag{43}$$

Thus, we have $\mathcal{F}(Q^*) = Q^*$ and

$$\begin{aligned}&||\mathcal{F}(Q_1(s^{TPSG},\boldsymbol{a}^{TPSG}))-\mathcal{F}(Q_2(s^{TPSG},\boldsymbol{a}^{TPSG}))||_\infty \\ &=||Q_1(s^{TPSG},\boldsymbol{a}^{TPSG})-Q_2(s^{TPSG},\boldsymbol{a}^{TPSG})||_\infty.\end{aligned} \tag{44}$$

This further gives

$$\begin{aligned}&E\{\Phi^t\} = \sum_{s^{TPSG'} \in \mathcal{S}^{TPSG}} P^{TPSG}_{s^{TPSG} s^{TPSG'}}(\boldsymbol{a}^{TPSG}) \\ &(\tfrac{1}{M}\textstyle\sum_{m=1}^{M} \mathcal{R}^t_m + \sigma \displaystyle\max_{a^{TPSG'} \in \mathcal{A}^{TPSG}} \overline{Q}^{t'} - Q^*) \\ &= \mathcal{F}(\overline{Q}^t) - Q^*.\end{aligned} \tag{45}$$

Then, we can obtain $||E\{\Phi^t\}||_\infty = ||\mathcal{F}(\overline{Q}^t) - \mathcal{F}(Q^*)||_\infty \leq \sigma||\overline{Q}^t - Q^*||_\infty$ according to the properties of the contraction mapping. Let $||\cdot||_\infty$ replace $||\cdot||_W$ in Lemma 1, the condition 3) in Lemma 1 is satisfied.

For the condition 4) in Lemma 1, we have

$$\begin{aligned}&E\{\Phi^t\} = \sum_{s^{TPSG'} \in \mathcal{S}^{TPSG}} P^{TPSG}_{s^{TPSG} s^{TPSG'}}(\boldsymbol{a}^{ERSG}) \\ &(\tfrac{1}{M}\textstyle\sum_{m=1}^{M} \mathcal{R}_m + \sigma \displaystyle\max_{a^{TPSG'} \in \mathcal{A}^{TPSG}} \overline{Q}^{t'} - Q^*) \\ &= \mathcal{F}(\overline{Q}^t) - Q^*,\end{aligned} \tag{46}$$

by which $\boldsymbol{Var}\{\Phi^t\} \leq Z(1 + ||\overline{Q}^t - Q^*||^2_W)$ can be clearly proved for a constant Z due to the fact that $\frac{1}{M}\sum_{m=1}^{M} \mathcal{R}^t_m$ is bounded [30]. Hence, the condition 4) in Lemma 1 is satisfied. This completes the proof of Lemma 1, and thus we can obtain $\mathbb{P}(lim_{t\to\infty}\overline{Q}(s^{TPSG}, \boldsymbol{a}^{TPSG}) = Q^{TPSG^*}(s^{TPSG}, \boldsymbol{a}^{TPSG})) = 1$.

B. Proof of Theorem 1

Similar to Lemma 1, the action and state in the bracket are omitted in this proof.

Since the Q value of state-action pair $(s^{TPSG}, \boldsymbol{a}^{TPSG})$ is updated if and only if the joint action $\boldsymbol{a}^{TPSG}$ occurs at state s^{TPSG}, $\{j\}, \forall j \geq 0$ is denoted as the sequence of updating state-action pair $(s^{TPSG}, \boldsymbol{a}^{TPSG})$ for trajectory learner. Hence, we have

$$\boldsymbol{Q}^{j+1} = (\boldsymbol{Y}_M - \beta^{TPSG}\boldsymbol{Y}_M)\boldsymbol{Q}^j\beta(\boldsymbol{R}^j + \boldsymbol{U}^j), \tag{47}$$

where $\boldsymbol{Q}^{j+1} = (Q^{j+1}_1, ..., Q^{j+1}_m)^\top$, and $\boldsymbol{Y}_M$ is the $M \times M$ identity matrix. In (47), we can obtain $\boldsymbol{R}^j = (\mathcal{R}^j_1, ..., \mathcal{R}^j_M)^\top$ and $\boldsymbol{U}^j = (\sigma \max_{\boldsymbol{a}^{TPSG'} \in \mathcal{A}^{TPSG}} Q^{j'}_1, ..., \sigma \max_{\boldsymbol{a}^{TPSG'} \in \mathcal{A}^{TPSG}} Q^{j'}_M)^\top$. Furthermore, we can obtain

$$\begin{aligned}&\boldsymbol{Q}^{j+1} - \overline{\boldsymbol{Q}}^{j+1} = \\ &(\boldsymbol{Y}_M - \beta^{TPSG}\boldsymbol{Y}_M)(\boldsymbol{Q}^j - \overline{\boldsymbol{Q}}^j) + \beta^{TPSG}(\hat{\boldsymbol{R}}^j + \hat{\boldsymbol{U}}^j),\end{aligned} \tag{48}$$

where $\mathbf{1}_M$ denotes the M-dimensional column vectors of ones, and then we can obtain $\overline{\boldsymbol{Q}}^j = \overline{Q}^j\mathbf{1}_M$. Additionally, we can also obtain $\hat{\boldsymbol{R}}^j = (\boldsymbol{Y}_M - (\frac{1}{M})\mathbf{1}_M(\mathbf{1}_M)^\top)\boldsymbol{R}^j$ and $\hat{\boldsymbol{U}}^j = (\boldsymbol{Y}_M - (\frac{1}{M})\mathbf{1}_M(\mathbf{1}_M)^\top)\boldsymbol{U}^j$. Hence, we have

$$\begin{aligned}||\boldsymbol{Q}^{j+1} - \overline{\boldsymbol{Q}}^{j+1}|| &= ||(\boldsymbol{Y}_M - \beta^{TPSG}\boldsymbol{Y}_M)\boldsymbol{Q}^j \\ &\quad -\overline{\boldsymbol{Q}}^{j+1}|| + ||\beta^{TPSG}(\boldsymbol{R}^j + \boldsymbol{U}^j)|| \\ &\leq ||\boldsymbol{Y}_M(\boldsymbol{Q}^j - \overline{\boldsymbol{Q}}^j)|| \\ &\quad +\beta^{TPSG}||(\boldsymbol{Q}^j - \overline{\boldsymbol{Q}}^j)|| \\ &\quad +\beta^{TPSG}||(\hat{\boldsymbol{R}}^j - \hat{\boldsymbol{U}}^j)|| \\ &\overset{(\varrho)}{\leq} (1 - X_j + \beta^{TPSG})||\boldsymbol{Q}^j - \overline{\boldsymbol{Q}}^j|| \\ &\quad +\beta^{TPSG}(||\hat{\boldsymbol{R}}^j|| + ||\hat{\boldsymbol{U}}^j||),\end{aligned} \tag{49}$$

where (ϱ) follows the Lemma 4.4 in [44] and $X_j \to 0$ as $j \to \infty$ with $X_j \in [0, 1]$. Since $\beta^{TPSG} \to 0$ as $j \to \infty$, it can be obtained that $(1 - X_j + \beta^{TPSG}) \to 0$ as $j \to \infty$. Hence, we can obtain that $\mathbb{P}(lim_{t\to\infty}||\boldsymbol{Q}^j - \overline{\boldsymbol{Q}}^j|| = 0) = 1$. Namely,

$$\begin{aligned}&\mathbb{P}(lim_{t\to\infty}Q^E_m(s^{TPSG},\boldsymbol{a}^{TPSG}) = \overline{Q}(s^{TPSG},\boldsymbol{a}^{TPSG})) \\ &= 1, \forall m \in \mathcal{M}, s^{TPSG} \in \mathcal{S}^{TPSG}, \boldsymbol{a}^{TPSG} \in \mathcal{A}^{TPSG}.\end{aligned} \tag{50}$$

Additionally, we have $\mathbb{P}(lim_{t\to\infty}\overline{Q}(s^{TPSG}, \boldsymbol{a}^{TPSG}) = Q^{TPSG^*}(s^{TPSG}, \boldsymbol{a}^{TPSG})) = 1$ according to Lemma 1. Therefore, we can obtain $\mathbb{P}(lim_{t\to\infty}Q(s^{TPSG}, \boldsymbol{a}^{TPSG}) = Q^{TPSG^*}(s^{TPSG}, \boldsymbol{a}^{TPSG})) = 1$, and this completes the proof of Theorem 1.

Acknowledgments

This was supported by the China National Key R&D Program under grant No. 2022YFB2901600, and by National Natural Science Foundation of China (NSFC) under grants No. 62002164 and No. 62176122.

References

[1] C. Yi, J. Cai, T. Zhang, K. Zhu, B. Chen, and Q. Wu, "Workload re-allocation for edge computing with server collaboration: A cooperative queueing game approach," *IEEE Trans. Mobile Comput.*, 2021.

[2] K. Zhu, Y. Zhou, C. Yi, and R. Wang, "Computation resource configuration with adaptive QoS requirements for vehicular edge computing: A fluid-model based approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 21 148–21 162, Nov. 2022.

[3] J. chen, C. Yi, J. Li, K. Zhu, and J. Cai, "Proc. IEEE ICC," in *A Triple Learner Based Energy Efficient Scheduling for Multi-UAV Assisted Mobile Edge Computing*, Jan. 2023.

[4] H. Qiu, K. Zhu, N. C. Luong, C. Yi, D. Niyato, and D. I. Kim, "Applications of auction and mechanism design in edge computing: A survey," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 2, pp. 1034–1058, Jun. 2022.

[5] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing," *IEEE Trans. Mobile Comput.*, vol. 21, no. 10, pp. 3536–3550, Oct. 2020.

[6] C. Yi, S. Huang, and J. Cai, "Joint resource allocation for Device-to-Device communication assisted fog computing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 1076–1091, Mar. 2021.

[7] C. Yi, J. Cai, K. Zhu, and R. Wang, "A queueing game based management framework for fog computing with strategic computing speed control," *IEEE Trans. Mobile Comput.*, vol. 21, no. 5, pp. 1537–1551, Mar. 2022.

[8] H. Wang, J. Wang, G. Ding, J. Chen, F. Gao, and Z. Han, "Completion time minimization with path planning for fixed-wing UAV communications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 7, pp. 3485–3499, Jul. 2019.

[9] C. Zhan and Y. Zeng, "Completion time minimization for multi-UAV-enabled data collection," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4859–4872, Oct. 2019.

[10] X. Hu, K. Wong, K. Yang, and Z. Zheng, "UAV-assisted relaying and edge computing: Scheduling and trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 69, no. 10, pp. 4738–4752, Oct. 2019.

[11] J. Ji, K. Zhu, C. Yi, and D. Niyato, "Energy consumption minimization in UAV-assisted mobile-edge computing systems: Joint resource allocation and trajectory design," *IEEE Internet Things J.*, vol. 8, no. 10, p. 8570–8584, Dec. 2021.

[12] X. Chen, C. Wu, T. Chen, Z. Liu, M. Bennis, and Y. Ji, "Age of information-aware resource management in UAV-assisted mobile-edge computing systems," in *Proc. IEEE GLOBECOM*, Dec. 2020.

[13] Y. Zhao, Z. Li, N. Cheng, R. Zhang, B. Hao, and X. Shen, "UAV deployment strategy for range-based space-air integrated localization network," in *Proc. IEEE GLOBECOM*, 2019, pp. 1–6.

[14] L. Yang, H. Yao, X. Zhang, J. Wang, and Y. Liu, "Multi-UAV deployment for MEC enhanced IoT networks," in *Proc. IEEE ICCC*, 2020, p. 436–441.

[15] J. Chen, C. Yi, R. Wang, K. Zhu, and J. Cai, "Learning aided joint sensor activation and mobile charging vehicle scheduling for energy-efficient WRSN-based industrial IoT," *IEEE Trans. Veh. Technol.*, 2022.

[16] K. Zhu, J. Yang, Y. Zhang, J. Nie, W. Lim, H. Zhang, and Z. Xiong, "Aerial refueling: Scheduling wireless energy charging for UAV enabled data collection," *IEEE Trans. Green Commun. Netw.*, pp. 1–1, Apr. 2022.

[17] M. Li, L. Liu, Y. Gu, Y. Ding, and L. Wang, "Aerial refueling: Scheduling wireless energy charging for UAV enabled data collection," *IEEE Internet of Things J.*, vol. 9, no. 5, pp. 3522–3532, Jul. 2021.

[18] L. Lv, C. Zheng, L. Zhang, C. Shan, Z. Tian, X. Du, and M. Guizani, "Contract and lyapunov optimization-based load scheduling and energy management for UAV charging stations," *IEEE Trans. Green Commun. Netw.*, vol. 5, no. 3, pp. 1381–1394, Sept. 2021.

[19] W. Jaafar and H. Yanikomeroglu, "Dynamics of laser-charged UAVs: A battery perspective," *IEEE Internet of Things J.*, vol. 8, no. 13, pp. 1381–1394, Jul. 2021.

[20] S. Jung, W. Yun *et al.*, "Orchestrated scheduling and multi-agent deep reinforcement learning for cloud-assisted multi-UAV charging systems," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 5362–5377, June. 2021.

[21] M. Zhao, Q. Shi, and M. Zhao, "Efficiency maximization for UAV-enabled mobile relaying systems with laser charging," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3257–3272, May. 2020.

[22] L. Zhang and N. Ansari, "Latency-aware IoT service provisioning in uav-aided mobile-edge computing networks," *IEEE Internet of Things J.*, vol. 7, no. 10, p. 10573–10580, Oct. 2020.

[23] Z. Liao, Y. Ma, J. Huang, J. Wang, and J. Wang, "Hotspot: A UAV-assisted dynamic mobility-aware offloading for mobile-edge computing in 3-D space," *IEEE Internet of Things J.*, vol. 8, no. 30, p. 10940–10952, Jul. 2021.

[24] Y. Liu, S. Xie, and Y. Zhang, "Cooperative offloading and resource management for UAV-enabled mobile edge computing in power IoT system," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, p. 12229–12239, Oct. 2020.

[25] 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 J.*, vol. 7, no. 4, p. 3147–3159, Apr. 2020.

[26] L. Wang, K. Wang, C. Pan, W. Xu, and N. Aslam, "Multi-agent deep reinforcement learning-based trajectory planning for multi-UAV assisted mobile edge computing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, p. 73–84, May. 2021.

[27] Y.Xu, T. Zhang, Y. Liu, D. Yang, L. Xiao, and M. Tao, "Cellular-connected multi-UAV MEC networks: An online stochastic optimization approach," *IEEE Trans. Commun.*, vol. 70, no. 10, p. 6630–6647, Aug. 2022.

[28] J. Chen and J. Xie, "Joint task scheduling, routing, and charging for multi-uav based mobile edge computing," in *Proc. IEEE ICC*, Aug. 2022.

[29] K. Wang, X. Zhang, L. Duan, and J. Tie, "Multi-UAV cooperative trajectory for servicing dynamic demands and charging battery," *IEEE Trans. Mobile Comput.*, vol. 22, no. 3, pp. 1599–1614, Mar. 2021.

[30] J. Luo, J. Song, F. Zheng, L. Gao, and T. Wang, "User-centric UAV deployment and content placement in cache-enabled multi-UAV networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, p. 5656–5660, Feb. 2022.

[31] C. Yi, S. Huang, and J. Cai, "An incentive mechanism integrating joint power, channel and link management for social-aware D2D content sharing and proactive caching," *IEEE Trans. Mobile Comput.*, vol. 17, no. 4, pp. 789–802, Apr. 2018.

[32] S. Huang, C. Yi, and J. Cai, "A sequential posted price mechanism for D2D content sharing communications," in *Proc. IEEE GLOBECOM*, Nov. 2016.

[33] F. Fazel, J. Abouei, M. Jaseemuddin, A. Anpalagan, and K. N. Plataniotis, "Secure throughput optimization for cache-enabled multi-UAVs networks," *IEEE Internet Things J.*, vol. 9, no. 10, p. 7783–7801, Sept. 2022.

[34] A. Seid, G. Boateng, B. Mareri, G. Sun, and W. Jiang, "Multi-agent DRL for task offloading and resource allocation in multi-UAV enabled IoT edge network," *IEEE Trans. Netw. Serv. Man.*, vol. 18, no. 4, pp. 4531–4547, Dec. 2021.

[35] Z. Ning, Y. Yang, X. Wang, L. Guo, X. Gao, S. Guo, and G. Wang, "Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing," *IEEE Trans. Mobile Comput.*, Nov. 2021.

[36] C. Zhao, J. Liu, M. Sheng, W. Teng, Y. Zheng, and J. Li, "Multi-UAV trajectory planning for energy-efficient content coverage: A decentralized learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 3193–3207, Oct. 2021.

[37] C. Yi, J. Cai, and Z. Su, "A multi-user mobile computation offloading and transmission scheduling mechanism for delay-sensitive applications," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 29–43, Jan. 2020.

[38] H. Mei, K. Yang, Q. Liu, and K. Wang, "Joint trajectory-resource optimization in UAV-enabled edge-cloud system with virtualized mobile clone," *IEEE Internet of Things J.*, vol. 7, no. 7, pp. 5906–5921, Jul. 2020.

[39] B. Xu, Z. Kuang, J. Gao, L. Zhao, and C. Wu, "Joint offloading decision and trajectory design for UAV-enabled edge computing with task dependency," *IEEE Trans. Wireless Commun.*, 2022.

[40] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, p. 2329–2345, Apr. 2019.

[41] T. Jaakkola, M. I. Jordan, and S. P. Singh, "Convergence of stochastic iterative dynamic programming algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 1994, p. 703–710.

[42] S. Kar, J. M. F. Moura, and H. V. Poor, "QD-learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations," *IEEE Trans. Signal Process.*, vol. 61, no. 7, p. 1848–1862, Apr. 2013.

[43] B. Liu, Y. Wan, F. Zhou, Q. Wu, and R. Hu, "Resource allocation and trajectory design for MISO UAV-assisted MEC networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 4933–4948, May. 2022.

[44] S. Kar, J. M. Moura, and H. V. Poor, "Distributed linear parameter estimation: Asymptotically efficient adaptive strategies," *SIAM J. Control Optim.*, vol. 51, no. 3, p. 2200–2229, Jan. 2013.

Jialiuyuan Li is currently pursuing the Ph.D. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China. His research interests include game theory, stochastic game, reinforcement learning and their applications in various wireless networks including edge computing, industrial IoT and UAV systems.

Changyan Yi (S'16-M'18) is currently a Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), and is also affiliated with the Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China. He received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Manitoba, MB, Canada, in 2018. From September 2018 to August 2019, he worked as a research associate in University of Manitoba, MB, Canada. He was awarded Changkong Scholor of NUAA in 2018, and Chinese Government Award for Outstanding Students Abroad in 2017. His research interests include game theory, queueing theory, machine learning and their applications in various wireless networks including edge/fog computing, IoT, 5G and beyond.

Jiayuan Chen is currently pursuing the M.S. degree with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China. His research interests include machine learning (e.g., reinforcement learning) and mechanism design with applications in resource management and decision making for various wireless networks and mobile services including edge/fog computing, industrial IoT, vehicular/UAV systems and digital twin.

Kun Zhu (M'16) received the Ph.D. degree from School of Computer Engineering, Nanyang Technological University, Singapore, in 2012. He was a research fellow with the Wireless Communications Networks and Services Research Group in University of Manitoba, Canada, from 2012 to 2015. He is currently a Professor in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics (NUAA), and Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China. He is also a Jiangsu specially appointed professor. His research interests include resource allocation in 5G, wireless virtualization, and self-organizing networks. He has published more than fifty technical papers and has served as TPC for several conferences. He won several research awards including IEEE WCNC 2019 Best paper awards, ACM China rising star chapter award.

Jun Cai (M'04-SM'14) received the Ph.D. degree from the University of Waterloo, ON, Canada, in 2004. From June 2004 to April 2006, he was with McMaster University, Canada, as a Natural Sciences and Engineering Research Council of Canada (NSERC) Postdoctoral Fellow. From July 2006 to December 2018, he has been with the Department of Electrical and Computer Engineering, University of Manitoba, Canada, where he was a full Professor and the NSERC Industrial Research Chair. Since January 2019, he has joined the Department of Electrical and Computer Engineering, Concordia University, Canada, as a full Professor and the PERFORM Centre Research Chair. His current research interests include edge/fog computing, ehealth, radio resource management in wireless communication networks, and performance analysis. Dr. Cai served as the Technical Program Committee (TPC) Co-Chair for IEEE GreenCom 2018; Track/Symposium TPC Co-Chair for the IEEE VTC-Fall 2019, IEEE CCECE 2017, IEEE VTC-Fall 2012, IEEE Globecom 2010, and IWCMC 2008; the Publicity Co-Chair for IWCMC 2010, 2011, 2013, 2014, 2015, 2017, 2020; and the Registration Chair for QShine 2005. He also served on the editorial board of IEEE Internet of Things Journal, IET Communications, and Wireless Communications and Mobile Computing. He received the Best Paper Award from Chinacom in 2013, the Rh Award for outstanding contributions to research in applied sciences in 2012 from the University of Manitoba, and the Outstanding Service Award from IEEE Globecom 2010.