Deep Reinforcement Learning Approach for UAV-Assisted Mobile Edge Computing Networks

Sangwon Hwang[*,†], Juseong Park[‡], Hoon Lee[§], *Member, IEEE*, Mintae Kim[*], and Inkyu Lee[*], *Fellow, IEEE*
[*]School of Electrical Engineering, Korea University, Seoul, Korea
[†]Central Technology Appraisal Institute, Korea Technology Finance Corporation, Seoul, Korea
[‡]Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA
[§]Department of Information and Communications Engineering, Pukyoung National University, Busan, Korea
Email: [*]{tkddnjs3510, wkd2749, inkyu}@korea.ac.kr, [‡]juseong.park@utexas.edu, [§]hlee@pknu.ac.kr

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001011

***Abstract*—This paper studies a deep reinforcement learning (DRL) approach for the unmanned aerial vehicle (UAV)-assisted mobile edge computing (MEC) networks where a UAV-mounted server offloads computation tasks of mobile users (MUs). We aim at minimizing the energy consumption of the MUs by adjusting UAV mobility, UAV-MU association, computation resource allocation, and task offloading rules. This requires an online and joint optimization of different types of variables constructing heterogeneous solution spaces. To realize real-time optimization strategies, we propose an online DRL method based on the twin-delayed deep deterministic policy gradient (TD3) framework. The joint optimization of heterogeneous action variables is tackled by a novel actor neural network that partitions the high-dimensional action set into several solution spaces. In addition, the proposed TD3 framework achieves adaptability to new task offloading requests through our proposed training and execution strategy. Numerical results verify the effectiveness of the proposed DRL architecture over benchmark schemes.**

I. Introduction

Mobile edge computing (MEC) systems have regarded as a promising solution for addressing latency-critical missions of mobile users (MUs) [1]. In the MEC system, MUs can offload their computational tasks to nearby edge servers. As a result, the computation latency and energy consumption of MUs are significantly reduced. However, it incurs additional communication overhead for exchanging task packets between edge servers and MUs. For this reason, the MEC systems have been tackled along with the efficient communication strategies for task transmission [2], [3].

Unmanned aerial vehicles (UAVs), which are capable of extending coverage of cellular systems [4], can also be applied for the efficient MEC network design [5]. UAV-mounted edge servers can be dispatched to the designated places to provide efficient computing services to MUs in a rapid and flexible manner [6]. To this end, it is essential to optimize the UAV traveling paths and task offloading strategies jointly. The design of UAV-assisted MEC networks has been investigated in [7], [8] using an iterative successive convex approximation (SCA) algorithm. The SCA algorithm should be executed in an offline manner by assuming fixed network configuration. This is, however, not suitable for practical MEC systems with time-varying random dynamics, i.e., locations of ground users, temporal changes in task volume, and stochastic properties of propagation environments.

To deal with the dynamically changing environment, the deep reinforcement learning (DRL) technique has been recently introduced to UAV-aided MEC networks [9]–[11]. The DRL framework has been originally developed to solve Markov decision processes (MDPs) using neural network (NN) enabled agent units. This approach transforms the optimization problems of UAV-assisted MEC networks as model-free decision-making problems under time-varying environments. A joint design of UAV trajectory and offloading scheduling has been tackled in [9]. A deep Q-network (DQN) method is employed in [12] to minimize the energy consumption and computing latency as well as maximize the number of processed tasks. However, the DQN approach is feasible only for discrete actions, whereas the trajectory of the UAV server is a continuous-valued optimization variable. To overcome this challenge, the conventional DQN methods [9], [10] simply quantize the moving space of the UAV into several grids, thereby resulting in inevitable performance degradation.

Therefore, the ability of handling continuous-valued action spaces is an essential requirements of the DRL solution suitable for the UAV-aided MEC networks. One possible approach is the deep deterministic policy gradient (DDPG) framework [13]. A recent work [11] has applied the DDPG framework for the UAV-aided MEC system to minimize the energy consumption of MUs. However, the DDPG agent only identifies the UAV trajectory, whereas other MEC features such as task scheduling are optimized using classical optimization algorithms. Such a separated design obviously degrades the performance of the entire MEC network. Combining many continuous variables related to the UAV trajectory and task offloading strategy in the MEC network leads to a cumbersome action space. This action space induces the agent to easily fall into local optimal points due to the

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A5A1027646).

978-1-6654-3540-6/22 © 2022 IEEE

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.

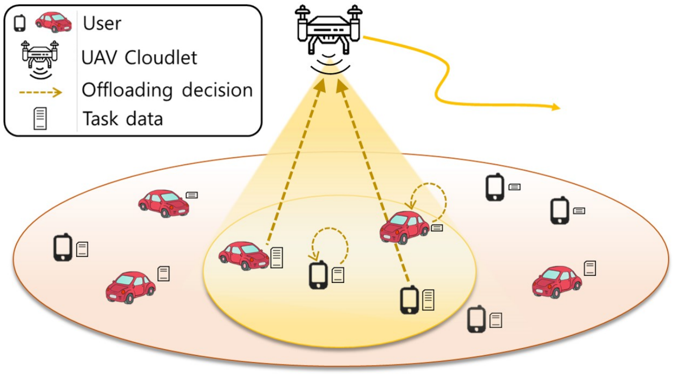

Fig. 1. UAV-assisted MEC system

failure of searching for the optimal action and, at worst, makes it almost impossible to learn stable policies in the training.

This paper investigates a DRL strategy for UAV-assisted MEC networks where computational tasks of MUs are handled by a UAV server flying in the three-dimensional (3D) space. We minimize the energy consumption of MUs via the joint optimization of UAV mobility, UAV-MU association, computing resource allocation, and task offloading rules subject to latency, maximum UAV speed, and maximum CPU cycle constraints. To deal with a cumbersome action space inherit from the heterogeneous types of continuous-valued variables, we propose a twin delayed DDPG (TD3) [14] based framework, which contains the novel actor NN installed at the UAV server. In particular, an output layer of the actor NN is carefully designed such that action variables are sequentially interrelated with each other. By doing so, the joint optimization of coupled action variables can be split into several small action sets.

In a practical UAV-assisted MEC system, new task offloading requests from the MUs can arise while the UAV travels. To reflect this aspect of the system, we propose random position initialization of the UAV in the training stage and then multiple executions of the proposed algorithm whenever new task offloading is requested. Specifically, the agent can terminate the session and then initiate a new session with the current information of the UAV, which has not been considered in the past works [9]–[11]. By doing so, our proposed method achieves adaptability to new task offloading requests while the UAV is on the move. Numerical results verify the superiority of the proposed TD3 approach in terms of the total energy consumption of MUs compared to existing DRL techniques.

II. System Model

We consider a UAV-assisted MEC system shown in Fig. 1 in which the UAV-mounted cloudlet flies over the network area to offer edge computing services for N ground MUs. The time-slotted protocol is adopted where each frame block of length τ_f is divided into T time slots each of duration $\tau = \tau_f/T$. Let $\mathcal{N} \triangleq \{1, \cdots, N\}$ be the set of MUs. MU $n \in \mathcal{N}$ is desired to complete its computational task of randomized volume I_n by each frame, i.e., the latency constraint is given by τ_f. As shown in Fig. 2, the task processing of MUs is handled over T time slots, and each time slot consists of three subsequential phases: task upload, computation, and task download phases. Each MU first decides offloading strategy. Offloaded tasks are first conveyed to the UAV in the task upload phase. Received tasks can be executed at the UAV. The processing outputs of the UAV are forwarded to intended MUs in the task download phase. In what follows, we explain the details of the UAV-aided MEC system.

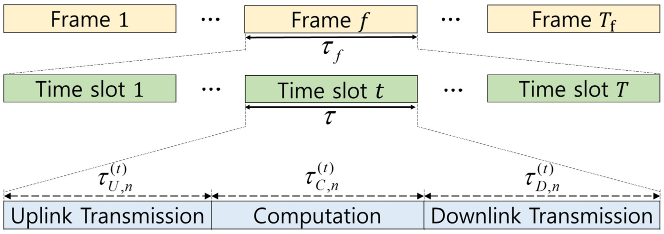

Fig. 2. Protocol of the MEC system

A. Mobility Model

Both the UAV and MUs are assumed to change their positions time to time. Unlike the UAV server which follows the optimized trajectory, the travelling paths of the MUs are determined according to their individual missions and are assumed to be randomly distributed. Locations of MU $n \in \mathcal{N}$ and the UAV at time slot t are represented by the 3D Cartesian coordinate vectors $\mathbf{u}_n^{(t)} = (u_{x,n}^{(t)}, u_{y,n}^{(t)}, 0)$ and $\mathbf{q}^{(t)} = (q_x^{(t)}, q_y^{(t)}, q_z^{(t)})$, respectively. Let $\eta^{(t)} \in (0, 2\pi]$, $\nu^{(t)} \in [0, \pi]$ and $v^{(t)} \in [0, v_{max}]$ be the azimuth angle, elevation angle, and speed of the UAV at time slot t respectively, where v_{max} stands for the maximum speed constraint. Then, the current UAV position $\mathbf{q}^{(t)}$ is given by

$$\mathbf{q}^{(t)} = \mathbf{q}^{(t-1)} + \tau v^{(t)} \Delta_{\mathbf{q}}^{(t)}, \tag{1}$$

where $\Delta_{\mathbf{q}}^{(t)}$=$(\sin \nu^{(t)} \cos \eta^{(t)}, \sin \nu^{(t)} \sin \eta^{(t)}, \cos \nu^{(t)})$.

B. Transmission Model

The channel gain $h_n^{(t)}$ between the UAV and MU n is expressed as $h_n^{(t)} = \rho_n^{(t)} \tilde{h}_n^{(t)}$, where $\rho_n^{(t)}$ and $\tilde{h}_n^{(t)}$ with $\mathbb{E}[\tilde{h}_n^{(t)}] = 1$ denote the large-scale and small-scale channel gains at time slot t, respectively. According to the availability of line-of-sight (LoS) links, $\rho_n^{(t)}$ is modeled as [15], [16]

$$\rho_n^{(t)} = \begin{cases} (d_n^{(t)})^{-\alpha}/(\rho_0 \chi_{\text{LoS}}), & \text{with probability } P_n^{(t)} \\ (d_n^{(t)})^{-\alpha}/(\rho_0 \chi_{\text{NLoS}}), & \text{otherwise,} \end{cases} \tag{2}$$

where ρ_0 is the average signal attenuation at the reference distance, χ_{LoS} and χ_{NLoS} respectively account for the excessive path loss coefficient of the LoS and non-LoS cases with $\chi_{\text{NLoS}} > \chi_{\text{LoS}} > 1$, α is the path loss exponent, and

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.

$d_n^{(t)} \triangleq ||\mathbf{u}_n^{(t)} - \mathbf{q}^{(t)}||$ indicates the distance between the UAV and MU n. The LoS probability $P_n^{(t)}$ is computed as [17]

$$P_n^{(t)} = \frac{1}{1 + K_1 \exp\left(-K_2[\nu_n^{(t)} - K_1]\right)}, \tag{3}$$

where K_1 and K_2 are constants relying on the propagation environment and $\nu_n^{(t)} \triangleq (180/\pi) \cdot \tan^{-1}(\sqrt{(d_n^{(t)}/q_z^{(t)})^2 - 1})$ is the elevation angle between the UAV and MU n.

We focus on the long-term channel gain since it dominates the quality of practical air-to-ground links [16]. By using (2) and the fact $\mathbb{E}[\tilde{h}_n^{(t)}] = 1$, the expected channel gain $g_n^{(t)} \triangleq \mathbb{E}[h_n^{(t)}]$ between the UAV and MU n is obtained as [18]

$$g_n^{(t)} = (d_n^{(t)})^{-\alpha}/(\rho_0 \bar{\chi}), \tag{4}$$

where $\bar{\chi} \triangleq P_n^{(t)} \chi_{\text{LoS}} + (1 - P_n^{(t)}) \chi_{\text{NLoS}}$. We employ the time division duplexing (TDD) mode over reciprocal channel gains $g_n^{(t)}$. The total system bandwidth is exclusively assigned to the UAV for supporting their scheduled MUs via the frequency division multiple access (FDMA) protocol. Denoting $b_n^{(t)}$ as the spectrum band allocation of the UAV to MU n, the corresponding uplink and downlink rates are respectively expressed by

$$R_{U,n}^{(t)} = b_n^{(t)} \log_2\left(1 + \frac{p_U g_n^{(t)}}{b_n^{(t)} N_0}\right), R_{D,n}^{(t)} = b_n^{(t)} \log_2\left(1 + \frac{p_D g_n^{(t)}}{b_n^{(t)} N_0}\right), \tag{5}$$

where p_U and p_D are uplink and downlink transmit power at the MUs and the UAV, respectively, and N_0 stands for the noise power.

C. Task Offloading

It is assumed that the volume of task $I_n, \forall n \in \mathcal{N}$, is evenly executed for the remaining time slots. Thus, the task size $D_n^{(t)}$ of MU n at time slot t is written by

$$D_n^{(t)} = \left(1 - \frac{t}{T}\right) D_n, \tag{6}$$

where D_n stands for the total volume of task I_n. We employ the partial task offloading strategy [19]. Provided that the UAV is scheduled to MU n, at the beginning of time slot t, MU n offloads $\beta_n^{(t)} \in [0, 1]$ portion of the data volume $\bar{D}_n \triangleq D_n/T$ to the UAV. The remaining $1 - \beta_n^{(t)}$ portion needs to be processed at MU n locally. Such a UAV-MU association status can be represented by a binary number $\lambda_n^{(t)}$. With $\lambda_n^{(t)}$ and B, the frequency resources allocation of the UAV to MU n is given as $b_n^{(t)} = B / \sum_{n=1}^N \lambda_n^{(t)}$ for $\sum_{n=1}^N \lambda_n^{(t)} \neq 0$, otherwise $b_n^{(t)}$ becomes zero.

MU n offloads the task of size $\beta_n^{(t)} \bar{D}_n$ to the UAV. Thus, the latency of the task upload phase $\tau_{U,n}^{(t)}$ from MU n to the UAV, the computation phase $\tau_{C,n}^{(t)}$, the task download phase $\tau_{D,n}^{(t)}$ from the UAV to MU n are calculated as

$$\tau_{U,n}^{(t)} = \frac{\delta_U C \beta_n^{(t)} \bar{D}_n}{R_{U,n}^{(t)}}, \tau_{C,n}^{(t)} = \frac{C \beta_n^{(t)} \bar{D}_n}{f_n^{(t)}}, \tau_{D,n}^{(t)} = \frac{\delta_D C \beta_n^{(t)} \bar{D}_n}{R_{D,n}^{(t)}}, \tag{7}$$

where $\beta_n^{(t)} > 0$ for $\lambda_n^{(t)} > 0$, otherwise $\beta_n^{(t)}$ becomes zero. Here, C denotes a constant for converting bits to CPU cycles. A constant δ_U includes the communication overheads such as channel coding. A constant δ_D is the ratio of output to input task sizes, i.e., the task of MU n processed by the UAV has the volume $\delta_D \beta_n^{(t)} \bar{D}_n$.

To handle a task offloaded from MU n of size $\beta_n^{(t)} \bar{D}_n$, the UAV allocates the CPU frequency $f_n^{(t)}$. Each UAV is subject to the computing resource constraint which restricts the total CPU cycles of the UAV to $f_{\max}$ [11]. We thus have

$$\sum_{n=1}^{N} f_n^{(t)} \leq f_{\max}. \tag{8}$$

The upload, computation, and downlink phases should be completed in each time slot having fixed length τ. This imposes the latency constraint written by

$$\tau_{U,n}^{(t)} + \tau_{C,n}^{(t)} + \tau_{D,n}^{(t)} \leq \tau. \tag{9}$$

D. Problem Formulation

This paper aims at minimizing the energy consumption of all MUs under the latency constraint (9) by optimizing the UAV mobility $\mathbf{Q} = \{v^{(t)}, \eta^{(t)}, \nu^{(t)}, \forall n, t\}$, UAV-UE association status $\mathbf{\Lambda} = \{\lambda_n^{(t)}, \forall n, t\}$, CPU cycles $\mathbf{F} = \{f_n^{(t)}, \forall n, t\}$, and task offloading strategy $\mathbf{B} = \{\beta_n^{(t)}, \forall n, t\}$. At time t, the energy consumption $E_n^{(t)}$ of MU n is given by

$$E_n^{(t)} = p_U \tau_{U,n}^{(t)} + \vartheta \frac{(C(1 - \beta_n^{(t)})\bar{D}_n)^3}{\tau^2}, \tag{10}$$

where the first term stands for the transmission energy in the task upload phase and the second term is induced by the local task computation of size $(1 - \beta_n^{(t)})\bar{D}_n$ with ϑ being the effective switched capacitance [19]. The total energy minimization problem can be formulated as

$$\begin{aligned}
\textbf{(P)}: \min_{\mathbf{Q},\mathbf{\Lambda},\mathbf{F},\mathbf{B}} \quad & \frac{1}{T} \sum_{t=1}^{T} \sum_{n=1}^{N} E_n^{(t)} \\
\text{s.t.} \quad & v^{(t)} \in [0, v_{\max}],\ \eta^{(t)} \in [0, 2\pi),\ \nu^{(t)} \in [0, \pi], && \text{(11a)} \\
& \lambda_n^{(t)} \in \{0, 1\}, && \text{(11b)} \\
& f_n^{(t)} \geq 0, && \text{(11c)} \\
& \beta_n^{(t)} \in [0, 1], && \text{(11d)} \\
& (8), (9). && \text{(11e)}
\end{aligned}$$

The above problem also falls into a class of nonconvex mixed integer nonlinear programming (MINLP) due to the nonconvex objective function and the inclusion of both integer- and continuous-valued variables. To tackle this challenging problem, we propose a DRL approach for addressing $(\mathbf{P})$ in the following sections.

III. Proposed DRL Approach

This section presents the TD3 method shown in Fig. 3 which handles the optimization task $(\mathbf{P})$. The TD3 technique has been widely adopted for handling RL tasks with continuous

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.

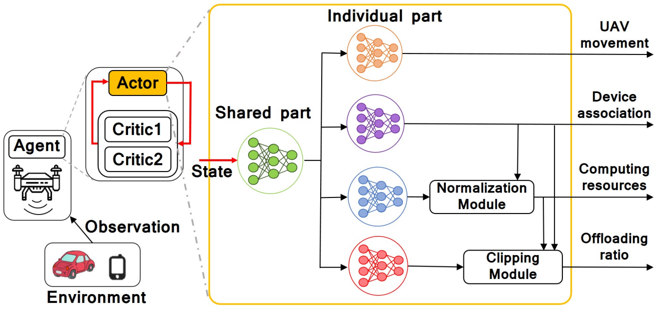

Fig. 3. The actor architecture of the proposed TD3 method

action spaces [14]. Such a property is suitable for our formulation $(\mathbf{P})$ which includes continuous optimization variables $v^{(t)}$, $\eta^{(t)}$, $\nu^{(t)}$, $\lambda_n^{(t)}$ $f_n^{(t)}$, and $\beta_n^{(t)}$. The TD3 technique realizes the actor-critic framework that parameterizes an actor generating a deterministic decision policy into an NN. In the following, we propose the TD3 framework which can jointly handle the heterogeneous optimization variables of $(\mathbf{P})$.

A. Proposed TD3 Method

The UAV is realized by an agent consisting of an actor NN and two critic NNs, whose parameters are represented by ϕ, θ_1 and θ_2, respectively. In addition, each NN has the associated target NNs with parameters ϕ', θ_1', and θ_2', respectively. The actor NN of the UAV yields the deterministic value $\mu_\phi(s^{(t)})$ of its action $a^{(t)}$ for a given state $s^{(t)}$. The action $a^{(t)}$ of the UAV includes its optimization variables of $(\mathbf{P})$. The critic NN models the action-value function $Q_\theta^{\mu_\phi}(s^{(t)}, a^{(t)})$ evaluated over the deterministic policy $\mu_\phi(\cdot)$ of the actor NN. In the following, we first transform the original problem in $(\mathbf{P})$ into a MDP by formalizing the action, state and reward.

1) Action: The actor NN takes its action $a^{(t)}$ as

$$a^{(t)} = \mu_\phi(s^{(t)}) + \epsilon, \tag{12}$$

where $s^{(t)}$ is the state variable of the UAV to be described and $\epsilon \sim \mathcal{N}(0, \sigma_a^2)$ represent the Gaussian random variable with variance σ_a^2, which is added for improving the exploration ability of the actor NN. The action $a^{(t)}$ is defined as

$$a^{(t)} \triangleq \left(v^{(t)}, \eta^{(t)}, \nu^{(t)}, \{\tilde{\lambda}_n^{(t)}, \tilde{f}_n^{(t)}, \tilde{\beta}_n^{(t)}, \forall n \in \mathcal{N}\}\right), \tag{13}$$

where tilde symbols represent outputs of individual DNN corresponding with optimization variables as shown in Fig. 3. To ensure feasibility, we apply the scaled sigmoid activation function to the outputs.

To handle the binary association variable $\lambda_n^{(t)}$, the actor NN first outputs a relaxation $\tilde{\lambda}_n^{(t)} \in [0, 1]$. To satisfy the binary association variable constraints in (11b), we introduce nonlinear flooring operation to the relaxed variables as

$$\lambda_n^{(t)} = \left\lfloor \tilde{\lambda}_n^{(t)} + 0.5 \right\rfloor. \tag{14}$$

For the computation resource allocation, the actor NN generates the variable $\tilde{f}_n^{(t)} \in [0, 1]$ representing the portion of the CPU cycle assigned for MU n. With the optimized association variable $\lambda_n^{(t)}$ and the fact that $\sum_{n=1}^N f_n^{(t)} = f_{max}$ holds at the optimum, we readily obtain the CPU cycle $f_n^{(t)}$ for the associated MUs. In the *normalization module* of Fig. 3, the CPU cycle $f_n^{(t)}$ is computed as

$$f_n^{(t)} = \frac{\lambda_n^{(t)} \tilde{f}_n^{(t)}}{\sum_{n=1}^N \lambda_n^{(t)} \tilde{f}_n^{(t)}} f_{\text{max}}, \tag{15}$$

where $\sum_{n=1}^N \lambda_n^{(t)} = 0$ stands for the case in which no MUs are associated with the UAV, thereby utilizing no computational resources.

Next, we discuss the strategy of identifying the task offloading strategy $\beta_n^{(t)}$, which is related to the association variable $\lambda_n^{(t)}$ and CPU cycle $f_n^{(t)}$. The actor NN of the UAV produces the variable $\tilde{\beta}_n^{(t)} \in [0, 1]$. In the *clipping module* of Fig. 3, the offloading ratio $\beta_n^{(t)}$ is derived as

$$\beta_n^{(t)} = \frac{\tau \tilde{\beta}_n^{(t)}}{\bar{D}_n} \left(\frac{C}{\lambda_n^{(t)} f_n^{(t)}} + \frac{\delta_D}{R_{D,n}^{(t)}} + \frac{\delta_U}{R_{U,n}^{(t)}} \right)^{-1}, \tag{16}$$

for $\lambda_n^{(t)} \neq 0$ and $f_n^{(t)} \neq 0$, otherwise $\beta_n^{(t)}$ becomes zero. By doing so, the optimized the offloading ratio $\beta_n^{(t)}$ can always satisfy the latency constraint (9). Interrelating the actor's output variables helps search for desirable policies in a cumbersome action space resulting from combining numerous continuous-valued random variables of the UAV-MEC system.

2) State: The state $s^{(t)}$ of the UAV at time slot t is constructed as a concatenation of internal state $s_I^{(t)}$ and external states $s_E^{(t)}$ obtained from interactions with the MUs. The internal state $s_I^{(t)}$ consists of the previous position $\mathbf{q}^{(t-1)}$ and MU associations $\tilde{\lambda}_n^{(t-1)}$, $\forall n$. The external state $s_{E,m}^{(t)}$ is collected from the MUs, which is composed of $\mathbf{u}_n^{(t-1)}$, $D_n^{(t-2)}$, $\beta_n^{(t)} \bar{D}_n$ and D_n, $\forall n$.

3) Reward: The goal of the TD3 architecture is to maximize the reward function by optimizing the actor and critic NNs. To address problem $(\mathbf{P})$, our reward includes the energy consumption of all MUs. At time slot $t+1$, the reward $r^{(t+1)}$ of the UAV is defined as

$$r^{(t+1)} = -\sum_{n=1}^N E_n^{(t)}. \tag{17}$$

B. Learning strategy

We discuss the training policy to optimize the actor and critic NNs. At each training iteration, the UAV randomly samples a mini-batch set $\mathcal{B}$ from its replay buffer $\mathcal{M}$ consisting of multiple transition samples. For simplicity, let $\mathbf{e} = (s, a, r, s')$ be a particular transition sample of the UAV where s' denotes the one-step forward state. When storing samples, the various location of the initial UAV should be gathered to improve the adaptability.

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.

The two critic NNs $Q_{\theta_i}(\cdot)(i=1,2)$ are optimized to minimize the loss function $L(\theta_i)$ written by

$$L(\theta_i)=\frac{1}{|\mathcal{B}|}\sum_{e\in\mathcal{B}}\big(y-Q_{\theta_i}(s,a)\big)^2, \tag{18}$$

where y stands for the target of the evaluated Q-value. To reduce the variance of the value estimate, which leads to stable target values, the Q-target y is obtained by adding the discounted minimum target critic NNs $Q_{\theta_i'}(s',\tilde{a})$ among $Q_{\theta_i'}, i=1,2$ to reward r. Here, $\tilde{a}$ is the output of the target actor NN $\mu_{\phi'}(\cdot)$, which is written by

$$\tilde{a}=\mu_{\phi'}(s')+\text{clip}(\epsilon_{\tilde{a}},c_{\tilde{a}}), \tag{19}$$

where $\text{clip}(z,c)\triangleq\max\{\min\{z,c\},c\}$ defines the clipping operator, c equals the clipping parameter, and $\epsilon_{\tilde{a}}\sim\mathcal{N}(0,\sigma_{\tilde{a}}^2)$ represents the zero-mean Gaussian random variable with variance $\sigma_{\tilde{a}}^2$. Such an operation resolves the overfitting issue and improves the generalization ability of the UAV agent.

On the contrary, the actor NN $\mu_\phi(\cdot)$ is desired to maximize the policy objective function $J(\phi)$ given by

$$J(\phi)=\frac{1}{|\mathcal{B}|}\sum_{s\in\mathcal{B}}Q_{\theta_1}\big(s,\mu_\phi(s)\big). \tag{20}$$

The objective in (20) quantifies the average Q-value of the actor NN through the first critic NN. Maximizing the average Q-value leads to finding the optimal policy of the actor NN, which enables the UAV to behave with the optimized variables in $(\mathbf{P})$.

As a result, the mini-batch SGD update of the critic NNs $Q_{\theta_i}(\cdot)$ and the actor NN are obtained as

$$\theta_i\leftarrow\theta_i-\alpha_{\text{C}}\nabla_{\theta_i}L(\theta_i),\ \phi\leftarrow\phi+\alpha_{\text{A}}\nabla_\phi J(\phi), \tag{21}$$

where α_{C}, α_{A} are their respective learning rates and ∇_z denotes the gradient operator with respect to a variable z.

The target critic NNs $Q_{\theta_i'}(\cdot)$ and the target actor NN $\mu_{\phi'}(\cdot)$ are also updated periodically. Similar to the actor NN, the target NNs are optimized less frequently to guarantee the stable Q-value estimate. These are softly updated from their pairs, i.e., $Q_{\theta_i}(\cdot)$ and $\mu_\phi(\cdot)$, with memory parameters ζ_{C} and ζ_{A}. The update strategies of the target NNs are obtained as

$$\theta_i'\leftarrow\zeta_{\text{C}}\theta_i+(1-\zeta_{\text{C}})\theta_i',\ \phi'\leftarrow\zeta_{\text{A}}\phi+(1-\zeta_{\text{A}})\phi'. \tag{22}$$

After the training, the only actor NN is employed as the real-time decision maker of the UAV, which can calculate their decision variables with locally available state variables.

IV. Simulation Results

We present numerical results validating the proposed TD3 approach for the UAV-aided MEC system. Our simulation setups are summarized in Table I. The twin critic NNs have four fully-connected layers each with $(512,256,128,64)$ neurons. The actor NN has three layers with $(512,256,128)$ neurons and two layers with $(64,32)$ neurons employed for the shared part and the individual part, respectively. Training is processed with 10^6 episodes each having $T_f=1$ frame. During the training procedure, the initial position of the UAV and the MUs are randomly set. The MUs move randomly according to the the Gauss-Markov process [20]. The trained actor NN is tested with 10^4 episodes with consecutive frames $T_f\geq 1$, where the MUs demand new task offloading requests.

TABLE I
Simulation Parameters

Symbol	Settings	Symbol	Settings
τ, T	0.2 s, 10	$\chi_{\text{LoS}}, \chi_{\text{NLoS}}$	3 dB, 23 dB
D_n	$[1.2, 12]$ Mbits	K_1, K_2	$11.95, 0.14$
C, ϑ	$1550, 10^{-28}$	$\|\mathcal{B}\|, \gamma$	$256, 0.99$
ρ_0, α	-38 dB, 2	δ_U, δ_D	$1, 0.2$
B, N_0	40 MHz, -130 dBm	$\sigma_a^2, \sigma_{\tilde{a}}^2, c_{\tilde{a}}$	$0.2, 0.2, 0.5$
$v_{\max}$	50 m/s	$\alpha_{\text{C}}, \alpha_{\text{A}}$	$5\times 10^{-3},\ 1\times 10^{-2}$
p_U, p_D	1 W, 10 W	$\zeta_{\text{C}}, \zeta_{\text{A}}$	$5\times 10^{-3},\ 5\times 10^{-3}$

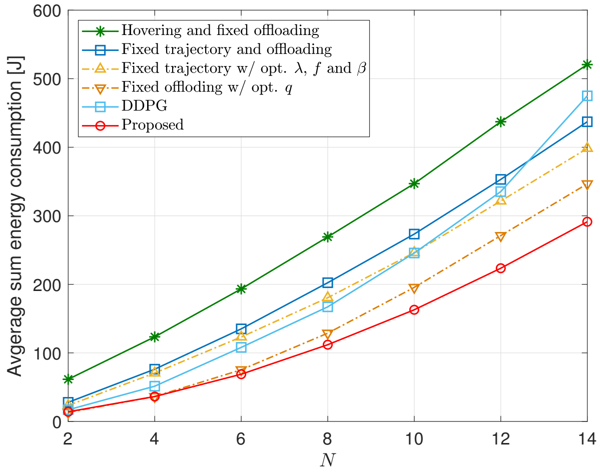

Fig. 4. Average sum energy consumption with respect to N

We consider the following benchmark schemes.

- *Hovering and fixed offloading*: The UAV hovers at a fixed point and evenly allocates the CPU cycles to all MUs.
- *DDPG* [11]: The conventional DDPG method is extended for our scenario.
- *Fixed trajectory and offloading*: The UAV simply moves to the centroid of all MUs. All MUs are associated with the UAV, the computation resources are equally allocated, and all MUs offload their task with $\tilde{\beta}_n^{(t)}=1,\forall n$.
- *Fixed trajectory with optimized $\boldsymbol{\lambda}$, $\boldsymbol{f}$ and $\boldsymbol{\beta}$*: The UAV-MU association, the computation resource allocation, and the task offloading are jointly optimized while the UAV simply moves to the centroid of all MUs.
- *Fixed offloading with optimized $\boldsymbol{q}$*: The trajectory of the UAV is optimized with evenly allocated computing resources. All MUs are simply assumed to associate with the UAV so that they offload their task with $\tilde{\beta}_n^{(t)}=1,\forall n$.

Figure 4 compares the average sum energy consumption performance with respect to the number of the MUs N with $f_{\max}=50$ GHz. The performance gap between the proposed TD3 method and benchmark schemes becomes larger

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.

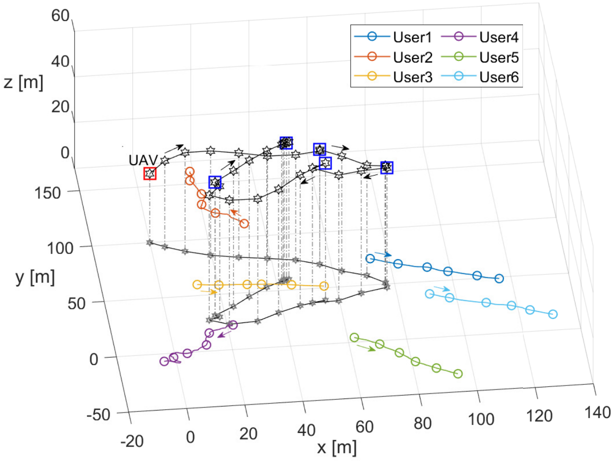

Fig. 5. 3D trajectory of the single UAV in test process

as N increases. Especially, a naive actor architecture of the conventional DDPG method becomes more inappropriate in handling heterogeneous action variables. The deficient exploration ability of the agent results in the failure of learning desirable policy in the enormous action space. This shows the effectiveness of the proposed TD3 approach with our new actor design which can efficiently decide heterogeneous action variables concurrently even with a large N.

Figure 5 illustrates the optimized UAV trajectories using the proposed TD3 approach. The UAV agent is executed during consecutive $T_f = 4$ frames, where the MUs demand new task offloading requests four times. The red square indicates the initial location of the UAV, and UAV positions sampled at the end of every frame are marked by four blue squares. During the entire frame block, the proposed TD3 method can realize dynamic trajectory control and offloading strategy. The UAV dynamically prioritizes some MUs and is dedicated to serving the prioritized MUs. As new task offloading requests of the MUs arise, the behavior of the UAV is constantly changing with the wished MUs. The UAV agent can adjust to the highly fluctuating time-varying dynamics from randomized movements and task offloading requests of the MUs.

V. Conclusion

This paper has proposed a DRL method for joint optimization of UAV mobility, UAV-MU association, computing resource allocation, and offloading strategy in UAV-assisted MEC networks. A novel actor NN structure enables the concurrent decision of heterogeneous action variables, which improves the exploration ability of the training algorithm. Also, it can adapt to new task offloading requests of the MUs while the UAV still travels. Numerical results have verified the effectiveness of the proposed DRL framework for offloading strategy of the UAV server to benchmark schemes.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, “A survey on mobile edge computing: The communication perspective,” *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart.,2017.

[2] T. Ouyang, Z. Zhou, and X. Chen, “Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing,” *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2333–2345, Oct. 2018.

[3] S. Eom, H. Lee, J. Park, and I. Lee, “Asynchronous protocol designs for energy efficient mobile edge computing systems,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 1013–1018, Jan. 2021.

[4] H. Lee, S. Eom, J. Park, and I. Lee, “UAV-aided secure communications with cooperative jamming,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 10, pp. 9385–9392, Oct. 2018.

[5] N. H. Motlagh, T. Taleb, and O. Arouk, “Low-altitude unmanned aerial vehicles-based internet of things services: Comprehensive survey and future perspectives,” *IEEE Internet Things J.*, vol. 3, no. 6, pp. 899–922, Dec. 2016.

[6] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, “Completion time and energy optimization in the UAV-enabled mobile-edge computing system,” *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7808–7822, Aug. 2020.

[7] S. Jeong, O. Simeone, and J. Kang, “Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2049–2063, Mar. 2018.

[8] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, “Energy efficient resource allocation in UAV-enabled mobile edge computing networks,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4576–4589, Sept. 2019.

[9] L. Zhang *et al.*, “Task offloading and trajectory control for UAV-assisted mobile edge computing using deep reinforcement learning,” *IEEE Access*, vol. 9, pp. 53 708–53 719, April 2021.

[10] Q. Liu, L. Shi, L. Sun, J. Li, M. Ding, and F. Shu, “Path planning for UAV-mounted mobile edge computing with deep reinforcement learning,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5723–5728, May 2020.

[11] 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.*, early access, doi: 10.1109/TMC.2021.3059691.

[12] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[13] T. P. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” *in Proc. Int. Conf. on Learning Representations (ICLR)*, 2016.

[14] S. Fujimoto, H. Hoof, and D. Meger, “Addressing function approximation error in actor-critic methods,” *in Proc. Int. Conf. on Machine Learning (ICML)*, pp. 1582–1591, 2018.

[15] M. Mozaffari *et al.*, “Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs,” *IEEE Trans. Wireless Commun.*, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[16] Y. Zeng, J. Xu, and R. Zhang, “Energy minimization for wireless communication with rotary-wing UAV,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2329–2345, April 2019.

[17] A. Al-Hourani, S. Kandeepan, and S. Lardner, “Optimal LAP altitude for maximum coverage,” *IEEE Wireless Commun. Lett.*, vol. 3, no. 6, pp. 569–572, Dec. 2014.

[18] M. Mozaffari *et al.*, “Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7574–7589, Nov. 2017.

[19] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, “Mobile-edge computing: Partial computation offloading using dynamic voltage scaling,” *IEEE Trans. Commun.*, vol. 64, no. 10, p. 4268, Oct. 2016.

[20] S. Batabyal and P. Bhaumik, “Mobility models, traces and impact of mobility on opportunistic routing algorithms: A survey,” *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1679–1707, 3rd Quart.,2015.

Authorized licensed use limited to: Beijing Information Science & Tech Univ. Downloaded on February 10,2023 at 09:09:10 UTC from IEEE Xplore. Restrictions apply.