Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6216372, 11 pages
https://doi.org/10.1155/2022/6216372

Research Article

Computation Offloading in Multi-UAV-Enhanced Mobile Edge Networks: A Deep Reinforcement Learning Approach

Bin Li,[1,2] **Shiming Yu**,[1] **Jian Su**,[1] **Jianghong Ou**,[3] **and Dahua Fan**[3]

[1]*School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China*
[2]*Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications), Ministry of Education, Nanjing 210003, China*
[3]*Starway Communication, No. 31, Kefeng Road, Guangzhou Science City, Guangzhou 510663, China*

Correspondence should be addressed to Jian Su; sj890718@gmail.com

Received 5 December 2021; Revised 11 February 2022; Accepted 23 February 2022; Published 7 March 2022

Academic Editor: Shu Fu

In this paper, we investigate an unmanned aerial vehicle- (UAV-) enhanced mobile edge computing network (MUEMN), where multiple UAVs are deployed as aerial edge servers to provide computing services for ground moving equipment (GME). Each GME is trained to simulate movement by a Gauss-Markov random model in this MUEMN. Under the condition of limited energy cost, UAV dynamically plans its flight position according to the movement trend of GME. Our objective is to minimize the total energy consumption of GME by jointly optimizing the offloading decisions of GME and the flight positions of UAVs. More explicitly, we model the optimization problem as a Markov decision process and achieve real-time offloading decisions via deep reinforcement learning algorithm according to the dynamic system state, where the asynchronous advantage actor-critic (A3C) framework with asynchronous characteristics is leveraged to accelerate the learning process. Finally, numerical results confirm that our proposed A3C-based offloading strategy can effectively reduce the total of energy consumption of GME and ensure the continuous operation of the GME.

1. Introduction

Mobile users usually have limited computing capabilities and battery storages; it is challenging to provide a satisfactory computing service and achieve a low service delay when they face with the emerging applications with computation-intensive features [1–3]. In this context, mobile edge computing (MEC) is considered as a key technology to mitigate these issues [4]. With the help of MEC, mobile devices have the option to offload their computing tasks to nearby edge servers with powerful computing capabilities, enabling the demands for lower energy consumption [5, 6] and reduced latency. Nevertheless, the location of MEC server is usually fixed and cannot be changed flexibly according to the needs of mobile users, which restricts the extension of MEC [7, 8]. At present, frequent occurrence of natural disasters may destroy basic communication facilities on the ground, which makes it difficult for rescue communication efforts. Compared with the general communication infrastructure, unmanned aerial vehicles (UAVs) are highly flexibility and inexpensive, enabling reliable communication. UAVs equipped with MEC servers greatly enhance the application scalability of the traditional MEC model [9, 10].

With the development and maturity of UAV-related technologies, they have been paid much attention in disaster rescue, mineral mining, geological exploration and other wireless scenarios [11, 12]. On the one hand, in regions with incomplete or damaged basic communication facilities, where large-scale outdoor activities are required within a short period of time, UAVs can be deployed in the air on demand to enhance network connectivity and provide reliable communication services. On the other hand, in many civilian application scenarios, such as live broadcast and video shooting, the flow of people tends to be huge, and the offloading of various data generated by mobile devices in these areas to the cloud or base stations (BSs) can trigger high latency [13]. Fortunately, UAVs equipped with the computing resources can serve as the edge nodes to relieve

the pressure on computing resources and improve the user experience. As such, joint development of UAV technology and MEC model, i.e., adopting UAVs to enhance mobile edge computing capabilities, is a promising direction for MEC development.

The current phase of research works on UAV-assisted mobile edge computing is divided into two categories: single/multiple UAV deployment [14] and latency reduction or energy reduction [15, 16]. Note that the ideal layout of the UAV can optimize the total coverage of the UAV, thereby maximizing network advantages. Nevertheless, despite being interesting, the UAV has size and weight constraints, and limited energy profoundly affects sustainable operations. To do this, the flight state of the UAV must be studied to optimize the use of UAV energy. Guo and Liu in [17] designed a single UAV-assisted mobile edge computing network. Under the UAV energy consumption constraint, the authors derived a suboptimal UAV trajectory layout by introducing block coordinate descent and successive convex approximation methods. Distinguished from [17], Liu et al. in [18] employed the Gauss-Markov random model (GMRM) to simulate the mobility of ground moving equipment (GME) and continuously adapted the UAV flight trajectory in the light of the time-varying location of terminal users to promote the quality of service for each mobile terminal user. The performance of the UAV-enabled MEC network is quite limited when a single UAV is used as a computation server in the large-scale scenarios, which motivates the deployment of multiple UAVs. Unlike single UAV deployment, multi-UAV-assisted MEC has more complex trajectories. In [19], Wang et al. synthesized the inter-UAV collision problem and presented a differential evolution algorithm with an elimination operator to optimize the layout of multiple UAVs. Shang and Liu in [20] obtained the target of minimizing the sum energy consumption of users by jointly optimizing users' association, resource allocation, and UAV layout. They further recommended the coordinate descent algorithm to decompose the energy consumption minimization problem into several subproblems to explore the suboptimal solution. In [21], Guo et al. studied a UAV-assisted MEC network with the goal of minimizing the sum delay of all users, adopting the theories of successive convex approximation and difference of convex programming to obtain the suboptimal solution. However, most of the literature defaults to static ground users; the work on jointly optimizing multiple UAV positions and offloading decisions considering ground user movement remains relatively scarce.

Sparked by the above-mentioned observations, in this paper, we propose an MUEMN architecture to provide edge computing for GME. We optimize the task offloading decisions of GME and the flight locations of UAVs in the network to achieve the goal of minimizing the total energy consumption of all GME. The resultant optimization problem is a mixed-integer nonconvex problem, and we propose a deep reinforcement learning- (DRL-) based asynchronous advantage actor-critic (A3C) algorithm, which asynchronously trains optimal computational offloading decisions for all GME in different environments and then uniformly uploads the training parameters to the global network to update the parameters and continuously train them to finally obtain optimal network parameters.

Specifically, the main contributions of this paper can be summarized as follows:

(1) Considering the dilemma of the traditional MEC model, we propose a multi-UAV-enhanced MEC network. Different from the fixed setting of ground equipment in most work, the ground equipment in our network follows the GMRM and moves within a certain period of time. UAVs continuously optimize their flight position with reference to the movement trend of GME

(2) We comprehensively consider the issues of UAV signal coverage, collisions between UAVs, and UAV energy consumption in the multi-UAV scenario. Under the constraints of these background issues, we introduce the A3C algorithm to find the suboptimal solution that minimizes the total energy consumption of all GME and derive the optimal computing task offload decisions and flight positions of UAVs

(3) Numerical results show that under the constraint of calculation delay, as the size of the calculation task increases, the offloading strategy based on the traditional algorithm is difficult to effectively reduce the total energy consumption of GME. In this paper, the proposed A3C algorithm with asynchronous characteristics can generate an effective offloading strategy

2. System Model and Problem Formulation

We describe the network model, communication model, computation model, flying model, and problem formulation in this section.

2.1. Network Model. We consider a multi-UAV-enhanced mobile edge computing network (MUEMN), including M UAVs deployed with MEC servers, $\mathcal{M} = \{1, 2, 3, \cdots, M\}$, and K GME, $\mathcal{K} = \{1, 2, 3, \cdots, K\}$. The network model is shown in Figure 1. We assume that the UAVs with limited energy can provide task offloading service for K GME within a certain period. Without loss of generality, the GME k and the UAV m serve one-to-one during this period, and all tasks must be guaranteed to be completed within the specified time period L. To simulate the mobility of GME and UAVs, we divide the calculation time of nonexecuting tasks in the period L into T frames, and the time of each frame t is uniform, which is denoted as $t = \{0, 1, 2, \cdots, T\}$. In this paper, the UAVs are assumed to be flying at a constant altitude H without frequenting ups and downs and maintain communication with K GME in each frame through the periodic time division multiple access (TDMA) protocol. Similar to prior studies, we use 3D Cartesian coordinate system to simulate the position of each node, and the coordinate unit is meter. Note that the 3D position of the UAV m is $\mathbf{U}_m^{\mathrm{u}}(t) = (x_m^{\mathrm{u}}(t), y_m^{\mathrm{u}}(t), H)$, whereas the two

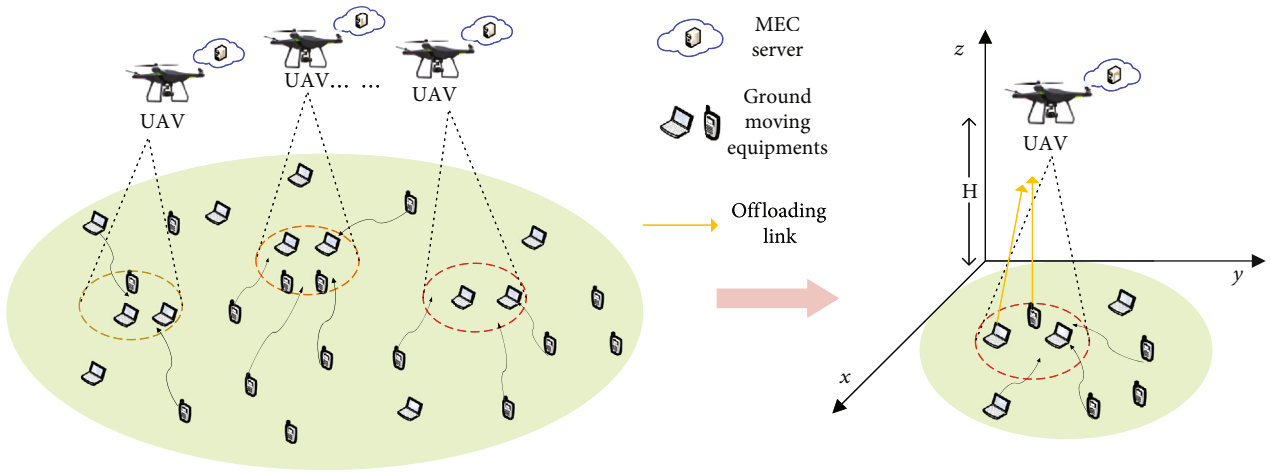


Figure 1: Task offloading network model for GME in multi-UAV scenarios.

UAVs need to meet the constraint $d_{\min}^{uu} \le d_{m_1 m_2}^{uu}(t)$, where $d_{\min}^{uu}$ represents the minimum allowable distance between two adjacent UAVs, and $d_{m_1 m_2}^{uu}(t) = \sqrt{(x_{m_1}^{u}(t) - x_{m_2}^{u}(t))^2 + (y_{m_1}^{u}(t) - y_{m_2}^{u}(t))^2}, \forall m_1, m_2 \in \mathcal{M}, m_1 \neq m_2$ represents the spacing between two adjacent UAVs.

In this MUEMN, we consider that all GME has random positions at $t = 0$ and do not change their positions within $\triangle_{t,t+1}$. Based on the GMRM [22], the movement speed and direction angle of the GME k at the tth ($t > 0$) frame are denoted as

$$v_k(t) = \tau_1 v_k(t-1) + (1-\tau_1)\bar{v}_k + \sqrt{1-\tau_1^2}\Omega_k, \quad (1)$$

$$\alpha_k(t) = \tau_2 \alpha_k(t-1) + (1-\tau_2)\bar{\alpha}_k + \sqrt{1-\tau_2^2}\Psi_k, \quad (2)$$

where $0 \le \tau_1, \tau_2 \le 1$ indicate the parameters for adjusting the state of the previous frame and $\bar{v}_k$ and $\bar{\alpha}_k$ stand for the average velocity and movement direction angle of the GME k, respectively. Also, Ω_k and Ψ_k follow two uncorrelated random Gaussian distributions with different mean-variance to simulate the random mobility of the GME k. From (1) and (2), the 3D X-coordinate and 3D Y-coordinate of the GME k at the tth frame can be deduced as

$$\begin{aligned} x_k(t) &= x_k(t-1) + v_k(t-1)\triangle_{t-1,t} \cos(\alpha_k(t-1)), \\ y_k(t) &= y_k(t-1) + v_k(t-1)\triangle_{t-1,t} \sin(\alpha_k(t-1)). \end{aligned} \quad (3)$$

To sum up, the 3D position coordinates of the GME k at the tth frame is $\mathbf{G}_k(t) = (x_k(t), y_k(t), 0)$. The visualized 3D model of the network unit can be referred to the right side of Figure 1.

2.2. Communication Model. In this paper, the line-of-sight wireless channels between GME and UAVs are more dominant than other channel impairments due to the high altitudes of UAVs. Therefore, the channel link between the GME k and the UAV m can be denoted by the free-space path loss model as follows:

$$h_{k,m}^{ul}(t) = \frac{\beta_0}{(x_m^u(t) - x_k(t))^2 + (y_m^u(t) - y_k(t))^2 + H^2}, \quad (4)$$

where β_0 is the channel power gain at a reference distance of 1 m.

Since each UAV can receive the offloaded task from at most one GME in each frame, the communication interference between channels can be neglected. As a result, the uplink transmission data rate between GME k and UAV m in a certain frame is calculated as

$$R_{k,m}^{ul}(t) = B \log_2\left(1 + \frac{h_{k,m}^{ul}(t)p_k}{\sigma^2}\right), \quad (5)$$

where B is the available channel bandwidth, p_k is the transmission power of the GME k, and σ^2 denotes the Gaussian noise power.

2.3. Computation Model. Considering that all GME distributed in the MUEMN generate a computationally intensive, latency-sensitive task $W_k = \{L_k, C_k, t_k^{\max}\}$, where L_k denotes the data size for calculating the offload task, C_k stands for the number of CPU cycles required to calculate each bit of task data, and $t_k^{\max}$ expresses the maximum tolerable task latency. The UAVs collaborate with each other to provide computing services to GME. Herein, $a_{k,m} \in \{0, 1\}$ is used to denote the GME k task offloading decision variable, where $a_{k,m} = 0$ indicates that the GME k chooses to perform local computation, and $a_{k,m} = 1$ expresses that the GME k chooses task offloading to the UAV m.

2.3.1. Local Computing. When the GME k decides to perform the calculation locally, the calculation execution time can be expressed as

$$t_k^{\mathrm{loc}} = \frac{L_k C_k}{f_k^{\mathrm{loc}}}, \tag{6}$$

where f_k^{loc} is the local computing power of the GME k. Correspondingly, the energy consumed by local calculation can be calculated as

$$E_k^{\mathrm{loc}} = \rho_k^{\mathrm{loc}} L_k C_k \left(f_k^{\mathrm{loc}}\right)^2, \tag{7}$$

where ρ_k^{loc} marked as the chip correlation coefficient of the GME k.

2.3.2. UAV Edge Computing. When the GME k moves into the coverage area of the UAV m, i.e., the constraint $d_{k,m}^{\mathrm{gu}}(t) \le R$ is satisfied and the UAV m becomes an option for the GME k to offload the computational task, where $d_{k,m}^{\mathrm{gu}}(t) = \sqrt{(x_k(t) - x_m^{\mathrm{u}}(t))^2 + (y_k(t) - y_m^{\mathrm{u}}(t))^2}, \forall k \in \mathcal{K}, \forall m \in \mathcal{M}$ denotes the horizontal distance between the GME k and the UAV m, $R = H \tan \vartheta$ indicates the coverage radius of each UAV, and ϑ is UAV antenna elevation angle [23]. When the GME k is in the coverage of multiple UAVs, the GME k randomly selects a UAV to offload the computational task. The process of offloading computational tasks from a GME to a UAV is divided into three main steps: (1) the GME offloads the computing task to a selected UAV; (2) the selected UAV receives the computational task and performs the calculation; (3) the selected UAV returns the results to the corresponding GME. As a result, the amount of data returned is small enough to be negligible. Therefore, the transmission time required for the GME k to offload the computational task to the UAV m, the energy consumption transmitted by the GME k, and the energy consumption received by the UAV m are expressed, respectively, as

$$\begin{aligned} t_{k,m}^{\mathrm{tr}}(t) &= \frac{L_k}{R_{k,m}^{\mathrm{ul}}(t)}, \\ E_{k,m}^{\mathrm{tr}}(t) &= p_k \frac{L_k}{R_{k,m}^{\mathrm{ul}}(t)}, \\ E_{k,m}^{\mathrm{re}}(t) &= p_m^{\mathrm{u}} \frac{L_k}{R_{k,m}^{\mathrm{ul}}(t)}, \end{aligned} \tag{8}$$

where p_m^{u} is the receiving power of the UAV m.

2.4. Flying Model

2.4.1. The Energy Consumption of Edge Computing. For a UAV with limited energy to work continuously, we need to constrain the UAV's energy. In this paper, the energy consumption of the UAV is divided into three main components: (1) reception energy consumption and calculated energy consumption (collectively known as edge computing energy consumption); (2) UAV flight energy consumption; (3) UAV hovering energy consumption. Let f_m^{u} and ρ_m^{u} be the computational power and the chip correlation coefficient of UAV m, respectively. Correspondingly, the time required and the energy consumed for the task calculation of the UAV m can be calculated as

$$t_{k,m}^{\mathrm{cal}} = \frac{L_k C_k}{f_m^{\mathrm{u}}}, \tag{9}$$

$$E_{k,m}^{\mathrm{cal}} = \rho_m^{\mathrm{u}} L_k C_k (f_m^{\mathrm{u}})^2. \tag{10}$$

According to (9) and (10), the edge computing energy consumption can be derived as

$$E_{k,m}^{\mathrm{edg}}(t) = E_{k,m}^{\mathrm{re}}(t) + E_{k,m}^{\mathrm{cal}} = p_m^{\mathrm{u}} \frac{L_k}{R_{k,m}^{\mathrm{ul}}(t)} + \rho_m^{\mathrm{u}} L_k C_k (f_m^{\mathrm{u}})^2. \tag{11}$$

2.4.2. The Energy Consumption of UAV Flying. Given that the UAV is flying at a constant altitude H, there is no change in the gravitational potential energy of the UAV in this paper. To this end, the UAV flight energy consumption only needs to consider kinetic energy, the flight speed, and energy consumption of the UAV m at the tth frame given by

$$\begin{aligned} v_m^{\mathrm{u}}(t) &= \frac{\mathbf{U}_m^{\mathrm{u}}(t) - \mathbf{U}_m^{\mathrm{u}}(t-1)}{\triangle}, \\ E_m^{\mathrm{f}}(t) &= \frac{1}{2} w \triangle \| v_m^{\mathrm{u}}(t) \|^2, \end{aligned} \tag{12}$$

where w is the effective weight of the UAV and $\triangle$ denotes the duration of each frame.

2.4.3. The Energy Consumption of UAV Hovering. The UAV receives a task offload request from a GME within the signal coverage area and will switch from flight state to hover state for the entire edge computing cycle. In this paper, the task offloading consists of two main phases: task transfer and execution of task calculation, and the calculation is reflected as

$$t_{k,m}^{\mathrm{edg}}(t) = t_{k,m}^{\mathrm{tr}}(t) + t_{k,m}^{\mathrm{cal}}. \tag{13}$$

To simplify the problem analysis, the energy consumed by the UAV m hovering E_m^{st} is considered as a constant.

By reason of the foregoing, under the premise that the total energy of the UAV mE_m^{u} is limited, the UAV m operation needs to satisfy the energy constraint

$$\sum_{k \in \mathcal{K}} a_{k,m} \left(E_{k,m}^{\mathrm{edg}}(t) + E_m^{\mathrm{st}} t_{k,m}^{\mathrm{edg}}(t)\right) + \sum_{t=1}^{T} E_m^{\mathrm{f}}(t) \le E_m^{\mathrm{u}}. \tag{14}$$

2.5. Problem Formulation. In this paper, we aim to minimize the total energy consumption of all GME for multi-UAV-enhanced MEC network by jointly optimizing the offloading decision variable $\mathbf{a} \triangleq \{a_{k,m}, \forall k \in \mathcal{K}, \forall m \in \mathcal{M}\}$ and UAV location $\{(x_m^{\mathrm{u}}, y_m^{\mathrm{u}})\}$. As such, the corresponding optimization problem can be formulated as

$$\begin{aligned}
\min_{\mathbf{a},\{(x_m^{\mathrm{u}},y_m^{\mathrm{u}})\}} \quad & \sum_{k=1}^{K}\sum_{m=1}^{M}\left((1-a_{k,m})E_k^{\mathrm{loc}}+a_{k,m}E_{k,m}^{\mathrm{tr}}\right),\\
\text{s.t.} \quad & \mathrm{C1}:\sum_{k\in K}a_{k,m}\left(E_{k,m}^{\mathrm{edg}}(t)+E_m^{\mathrm{st}}t_{k,m}^{\mathrm{edg}}(t)\right)+\sum_{t=1}^{T}E_m^{\mathrm{f}}(t)\le E_m^{\mathrm{u}},\forall k\in\mathscr{K},\forall m\in\mathscr{M},\\
& \mathrm{C2}:a_{k,m}d_{k,m}^{\mathrm{gu}}\le R,\forall k\in\mathscr{K},\forall m\in\mathscr{M},\\
& \mathrm{C3}:d_{\min}^{\mathrm{uu}}\le d_{m_1m_2}^{\mathrm{uu}},\forall m_1,m_2\in\mathscr{M},m_1\ne m_2,\\
& \mathrm{C4}:(1-a_{k,m})\frac{L_kC_k}{f_k^{\mathrm{loc}}}+a_{k,m}\left(t_{k,m}^{\mathrm{tr}}+t_{k,m}^{\mathrm{cal}}\right)\le t_k^{\max},\forall k\in\mathscr{K},\forall m\in\mathscr{M},\\
& \mathrm{C5}:a_{k,m}\in\{0,1\},\forall k\in\mathscr{K},\forall m\in\mathscr{M},\\
& \mathrm{C6}:\sum_{m=1}^{M}a_{k,m}=1,\forall k\in\mathscr{K},\forall m\in\mathscr{M},
\end{aligned}\tag{15}$$

where constraint C1 regulates the use of the UAV energy, constraint C2 indicates the coverage of the UAV signal, constraint C3 ensures the minimum distance between adjacent UAVs to prevent collisions, constraint C4 denotes the maximum latency allowed for the computing task, constraint C5 refers to the binary constraint, and C6 guarantees that each GME connects to at most one UAV. It can be clearly seen that Problem (15) is a mixed-integer nonlinear and nonconvex problem due to the nonconvex objective function and the constraint, which is challenging to solve and requires highly computational complexity to find a globally optimal solution utilizing classical mathematical tools. To this end, appropriate algorithms need to be designed for solving this type of problem efficiently [24]. In the following sections, we propose an A3C-based computational offloading algorithm to obtain suboptimal solution.

3. Proposed DRL-Based Approach: A3C

In this paper, we intend to use DRL-based A3C algorithm [25] to explore unknown environments, where GME goes through different task offloading decisions and UAVs learn from feedback by trying different moves. Continuously, the global network optimizes task offloading decisions and location moves until a suboptimal solution is obtained.

3.1. An Overview of A3C Algorithm. Compared with the traditional deep reinforcement learning algorithms, the A3C algorithm optimizes and improves the actor-critic (AC) algorithm [26]. Based on this, the A3C algorithm solves the problem that the AC algorithm is difficult to converge and achieves fast convergence, which can meet our needs. In detail, the AC algorithm uses an approximate value function to guide the policy parameter updates, and its single-step update can speed up the convergence. However, despite being effective, the AC algorithm requires a complete sequence of states, and iteratively updates the policy function separately, so that it is not easy to converge. As shown in Figure 2, the A3C algorithm utilizes its asynchronous feature to start multiple threads at the same time, while the agents learn by interacting with the environments in multiple threads separately. Each thread will complete the training independently and uploads the training data to the global model parameters in an asynchronous manner. At the same time, the model parameters of the threads are periodically synchronized with the global model parameters, and then, a new round of training is performed with the new parameters.

3.2. A3C-Based Offloading of Computing Task. In the MUEMN model, the GME with computational tasks may choose to compute locally or offload to UAVs in the current signal coverage area within each frame. Subject to the anti-collision constraint, energy constraint, and delay constraint, we aim to minimize the total energy consumption of all GME. The objective optimization problem can be modelled as an MDP by offloading GME tasks.

An MDP consists of a five-tuple: MDP $= \langle \mathscr{S}, \mathscr{A}, \mathscr{P}, \mathscr{R}, \gamma \rangle$, where $\mathscr{S}$ denotes the set of states of the environment, $\mathscr{A}$ describes the set of actions, $\mathscr{P}$ indicates the state transfer probability, $\mathscr{R}$ expresses the reward function, and γ is the decay coefficient. The MDP formulation of the MUEMN is as follows.

The state space in the MUEMN is described as

$$\mathscr{S}=\{s_t|s_t=\{\mathbf{U}_m^{\mathrm{u}}(t),\mathbf{G}_k(t),E_m^{\mathrm{u}}\}\},\quad \forall k\in\mathscr{K},\forall m\in\mathscr{M}.\tag{16}$$

The action space in the MUEMN consists of two kinds of actions, i.e., local computation and offloading to the UAV, expressed as follows

$$\mathscr{A}=\{a_k(t)|a_k(t)=\{0,1\}\},\quad \forall k\in\mathscr{K}.\tag{17}$$

The state transfer and action decision of the GME in the MUEMN is only related to the positions of GME and UAVs and the energy states of UAVs, so the state transfer probability can be expressed as

$$\mathscr{P}_{ss'}=\mathscr{P}\left(s_{t+1}=s'|s_{t=s}\right).\tag{18}$$

To minimize the total energy consumption of all GME, we consider designing a reward function, which assigns a negative reward if the action taken by the GME k in the state of the current frame satisfies constraints C1-C6. Briefly, the reward function can be calculated as

$$r(s_t,a_t)=-\left((1-a_{k,m})E_k^{\mathrm{loc}}+a_{k,m}E_{k,m}^{\mathrm{tr}}\right).\tag{19}$$

On the contrary, if the GME k violates the constraints, we will punish it. For instance, the GME k local calculation violates the delay constraint, we will do the following processing for its local calculation energy consumption

$$r(s_t,a_t)=-\frac{t_k^{\mathrm{loc}}}{t_k^{\max}}E_k^{\mathrm{loc}}.\tag{20}$$

With regard to the optimization Problem (15), it can be observed that the value sequence of the binary decision variables directly affects the suboptimal solution of the optimization problem. We pass the state of the environment to the local network to obtain a sequence of task offloading decisions and then accumulate the reward value adopting

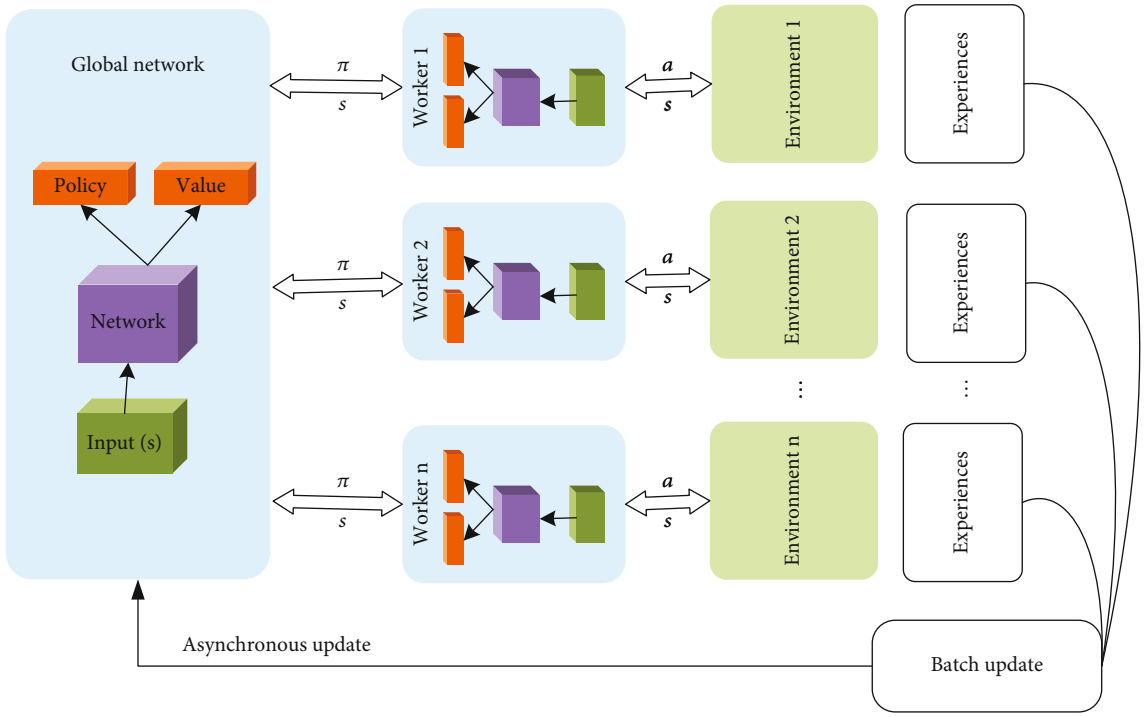


Figure 2: A3C algorithm asynchronous training framework.

the reward function. Multiple threads proceed asynchronously in this manner, leaving the training parameters to the global network for coordination. Ultimately, an optimal network parameter and a suboptimal reward value are derived. As shown in Algorithm 1, we give the detailed steps of the optimal network parameters for the A3C-based offloading strategy in the MUEMN.

3.3. Calculating Offloading Decision Generation. In particular, we introduce the interaction process of a certain thread's environment state sequence and action sequence in this subsection. For the computational task L_k generated by the GME k at tth frame, we consider the position of the GME k, the positions of the UAVs, and the UAVs' energy states as a set of state. Further, we input the state sequence $\mathcal{S}$ into the local network model of a thread, which is trained by the network to produce an action sequence $\mathcal{A}$, the elements of which correspond to the task offloading decisions of each of the K GME.

4. Numerical Results

4.1. Simulation Configurations. In this section, the simulation results are presented to evaluate the performance of our proposed A3C algorithm. We compare A3C with the following there commonly used baseline methods:

(1) Greedy: when the GME is in the coverage area of the UAV, the GME selects either local execution or UAV execution for the computation task depending on the magnitude of the local computation delay and transmission delay [27]

(2) Random: the GME within UAV signal coverage can randomly select the object of computational task execution, i.e., local execution or UAV execution

(3) DQN: the neural network accepts the environment state to calculate the value function and then uses the ε-greedy strategy to output the task offload decisions [28]

In the simulation, the software environment is Python 3.7 with TensorFlow and Visual Studio Code, and the hardware environment is a computer with Intel Core i5-9500 CPU and RAM 8.0 GB. Consider that the simulation scenario consists of M UAVs and K GME, and the area is a 300 m ◊ 300 m square single cell area. The horizontal plane flight altitude of the UAV $H = 80$ m. The effective weight of the UAV is set to 10 kg, the energy budget of the UAV E_m^{u} is set to 200 kJ, and the hovering energy consumption E_m^{st} is set as 200 W [29]. In addition, we set the total duration of each task completion cycle as $L = 10$ s, and the part of equipment that moves freely during this time can be divided into $T = 50$ frames; thus, the duration of each frame can be expressed as $\triangle = (L - \max\{a_{k,m} t_{k,m}^{\mathrm{cal}}, \forall k \in \mathcal{K}, \forall m \in \mathcal{M}\})/T$. Furthermore, we assume that the channel power gain β_0 at the reference distance of 1 m is set to -50 dB. The available bandwidth B is set to be 40 MHz and the noise power $\sigma^2 = 10^{-16}$ W. The coefficients related to the GME and the UAVs are set as $\rho_k^{\mathrm{loc}} = \rho_m^{\mathrm{u}} = 10^{-28}$.

Regarding the size of the computational tasks, we assume that they are randomly arranged in a certain interval. Meanwhile, the computing power of the GME k is set to $f_k^{\mathrm{loc}} = 0.5$ G cycles/s, and the computational capability of the UAV m

Input: The decay value of the reward γ, global shared count N, and global maximum shared count $N_{\max}$.
Output: Optimal network parameters θ and ω as well as the reward value $\mathscr{R}(s_n, a_n)$.
Initialization: Actor network parameter θ and critic network parameter ω in the global shared parameters, actor network parameter θ', and critic network parameter ω' in this thread;
Initialize local count $n = 1$;
repeat
  Reset gradient of local actor network and critic network: $d\theta \longleftarrow 0, d\omega \longleftarrow 0$;
  Synchronize parameters from the global network to this thread network: $\theta' = \theta, \omega' = \omega$;
  $n_{\text{start}} = n$;
  Initialize state s_n;
  repeat
    Based on the strategy $\pi(a_n|s_n;\theta')$ select out action a_n;
    Execute action a_n to get reward value r_n and new state s_{n+1};
    $N \longleftarrow N + 1, n \longleftarrow n + 1$;
  until s_n is the terminal state or $n - n_{\text{start}} == n_{\max}$;
  Calculate the value of $Q(s, n)$ for state s_n at the last count n:
  $$Q(s,n) = \begin{cases} 0, & s_n \text{ is the terminal state,} \\ V(s_n, \omega'), & \text{otherwise;} \end{cases}$$
  for $i \in (n-1, n-2, \cdots, n_{\text{start}})$ **do**
    $Q(s, i) = r_i + \gamma Q(s, i+1)$;
    Calculate the cumulative gradient of local actor parameter θ:
    $d\theta \longleftarrow d\theta + \nabla_{\theta'} \log \pi(a_n|s_n;\theta')(Q(s,i) - V(S_i, \omega'))$
    Calculate the cumulative gradient of local critic parameter ω:
    $d\omega \longrightarrow d\omega + (\partial(Q(s,i) - V(S_i, \omega'))^2 / \partial\omega'$;
  end for
  Update the global network model parameters θ and ω using the local cumulative gradient $d\theta$ and $d\omega$ asynchronously, respectively;
until $N > N_{\max}$

Algorithm 1: A3C-based offloading of computational tasks—arbitrary single-threaded execution process.

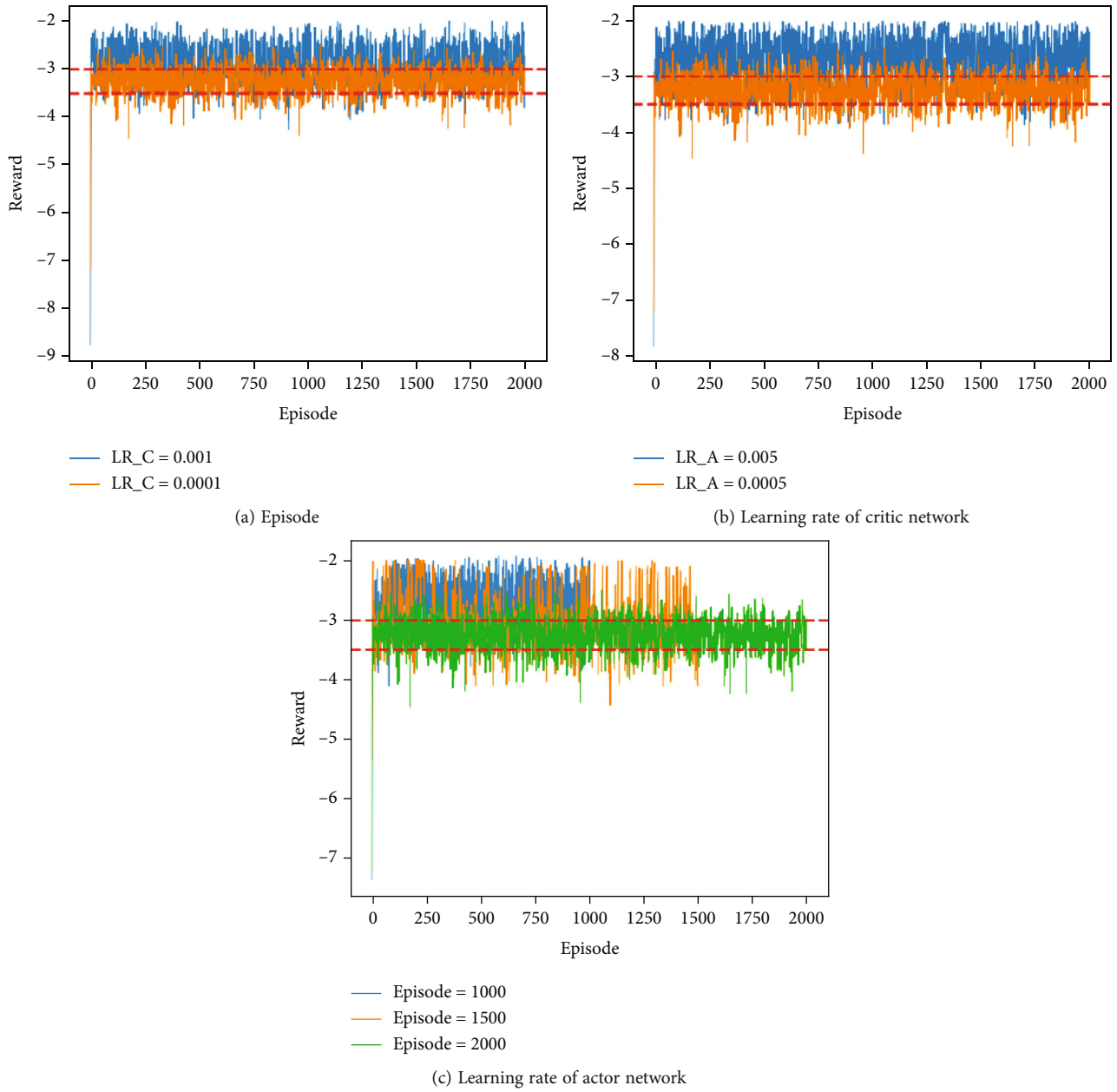
to each GME is set to $f_m^{\text{u}} = 5$ G cycles/s by reference to [30]. The specific parameter settings are shown in Table 1.

Table 1: Parameter setting.

Parameters	Values	Parameters	Values
β_0	-50 dB	ϑ	$\pi/4$
σ^2	10^{-16} W	p_k	50 mW
$\rho_k^{\text{loc}}, \rho_m^{\text{u}}$	10^{-28}	p_m^{u}	50 mW
f_k^{loc}	0.5 G cycles/sec	B	40 MHz
f_m^{u}	5 G cycles/sec	L_k	[5, 10] MB
$d_{\min}^{\text{uu}}$	4 m	C_k	[150, 200] cycles/bit

4.2. Performance Comparison. Assuming the number of GME is 20 and the number of UAVs is 3, i.e., $K = 20, M = 3$, we can clearly observe from Figure 3 that the total energy consumption of GME decreases rapidly within several iterations. The asynchronous nature of the A3C algorithm makes the reward value oscillate in an interval, and we need to reduce the oscillation interval as much as possible. When the scale of GME is large, it is acceptable for the reward value to fluctuate within 0.5. Figure 3(a) shows that as the number of episodes increases, the oscillation interval gradually decreases. At this point, we can regard it as the reward value gradually converging. Figure 3(b) shows that the oscillation interval of the reward value shrinks rapidly, indicating that the decrease of the critic network learning rate can reduce the oscillation interval of the reward value and accelerate the convergence of the reward value. Coincidentally, we reduce the learning rate of the actor network and obtain the goal of rapid convergence of the reward value in Figure 3(c). It is important that due to the characteristic that the reward value oscillates in a certain range, we use the average value of the upper and lower limits of the oscillating range as the final result of the reward value.

Figure 4 shows the minimum total energy consumption of GME as the number of GME increases. In this figure, the number of UAVs is 3, the size of task is set as 8 MB, and the number of CPU cycles to compute each bit is set as 160 cycles/bit, i.e., $M = 3$, $L_k = 8$ MB, and $C_k = 160$ cycles/bit. For different offloading strategies, the total energy consumption of GME also increases linearly with the increase of GME. When the UAVs' coverage is low and the number of GME is small, it is difficult to satisfy that all GME is within the UAV signal coverage. In the figure, the total energy consumption of GME under the four strategies is not much different at $K = 5$. But it can be seen that under the same computing task requirements, the greater the number of GME, the greater the total energy consumption of GME, and the offloading strategy based on A3C algorithm proposed by us is more advantageous.



(a) Episode

(b) Learning rate of critic network

(c) Learning rate of actor network

Figure 3: Comparison of total energy consumption with different number of GME.

Figure 5 compares the proposed offloading strategy based on A3C algorithm with other strategies in terms of all GME energy consumption versus different sizes of computation task. In this figure, the number of UAVs is 3, the number of GME is set as 20, and the number of cycles to compute each bit is set as 160 cycles/bit, i.e., $M = 3, K = 20$, and $C_k = 160$ cycles/bit. It can be seen that with the increase of the computational task size, the energy consumption gap of the four offloading strategies gradually increases. The reason is that with the increase of the data scale, due to the limitation of the calculation delay, the random strategy and the greedy strategy gradually lose their effect. By analysing the linear trend of Random algorithm, Greedy algorithm, DQN algorithm, and A3C algorithm in the graph, we can see that the larger the amount of data, the clearer the advantage of our proposed offloading strategy.

Figure 6 describes the sum energy consumption of all GME corresponding to the number of CPU cycles required for different calculations per bit of task data. In this figure, the size of task is set as 8 MB, the number of UAVs is 3 and the number of GME is set as 20, i.e., $L_k = 8$ MB, $M = 3$, and $K = 20$. As shown in Figure 6, it is interesting to note that there is a significant gap between the offloading strategy based on the DRL algorithm and the offloading strategy based on the random algorithm and the greedy algorithm. The reason is that the larger the number of cycles for calculating each bit, the higher the calculation delay

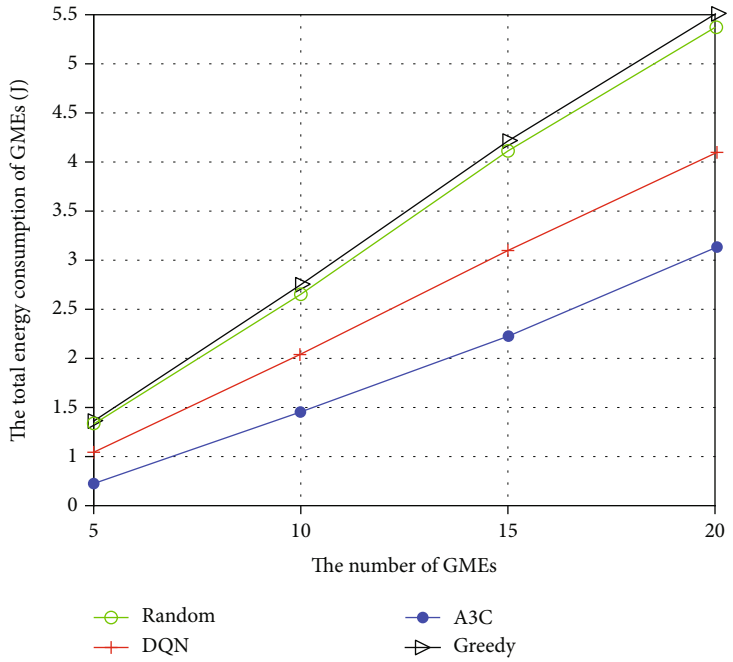


Figure 4: Comparison of total energy consumption with different number of GME.

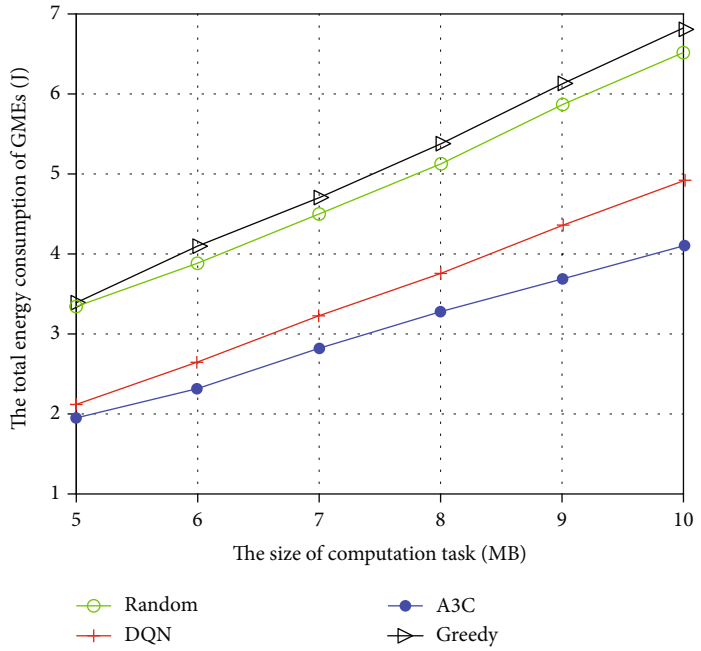


Figure 5: The total energy consumption of GME with different sizes of computation task.

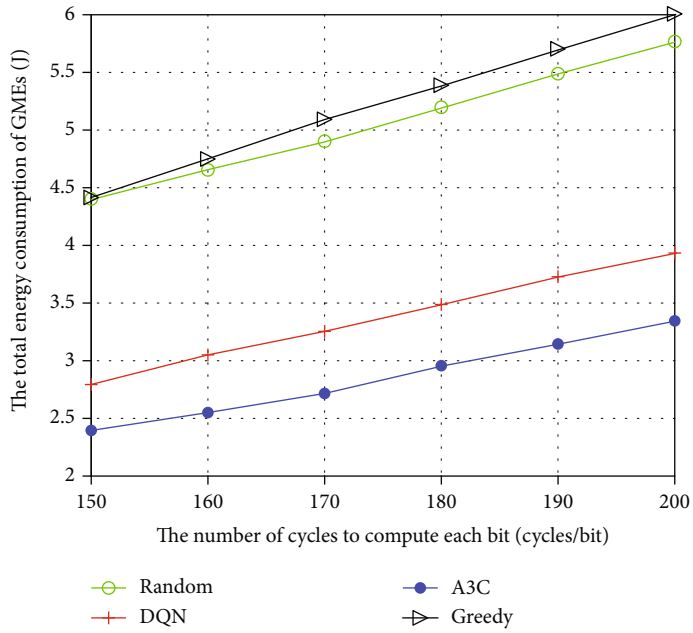


Figure 6: Performance comparison of different strategies versus the number of cycles to compute each bit.

requirements, and the traditional algorithms are difficult to meet such task offloading requirements.

5. Conclusions

In this paper, we researched the computational task offloading problem in an MUEMN and formulated a constrained optimization problem with the objective of minimizing the total energy consumption of all GME. We proposed a model-free DRL scheme with an asynchronous A3C algorithm to effectively generate offloading decisions. A large number of numerical results showed that the proposed A3C algorithm can accelerate the convergence speed of the algorithm and effectively reduce the total energy consumption of GME. In theory, the greater the number of UAVs, the task calculation delay can be greatly reduced, and the energy consumption of GME can also be reduced. However, too many UAVs can be a waste of resources when the limited space of application scenarios. In future work, we plan to study the optimal number of UAVs deployed in the MUEMN with limited space.

Data Availability

All the data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 62101277), by the Natural Science Foundation of Jiangsu Province (no. BK20200822), by the Natural Science Foundation of Jiangsu Higher Education Institutions of China (no. 20KJB510036), and by the open research fund of Key Lab of Broadband Wireless Communication and Sensor Network Technology (Nanjing University of Posts and Telecommunications) under grant JZNY202103, Ministry of Education.

References

[1] L. Zhao, G. Han, Z. Li, and L. Shu, "Intelligent digital twin-based software-defined vehicular networks," *IEEE Network*, vol. 34, no. 5, pp. 178–184, 2020.

[2] L. Zhao, W. Zhao, A. Y. Al-Dubai, G. Min, A. Y. Zomaya, and C. Gong, "Novel online sequential learning-based adaptive routing for edge software-defined vehicular networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 5, pp. 2991–3004, 2021.

[3] Z. Chang, L. Liu, X. Guo, and Q. Sheng, "Dynamic resource allocation and computation offloading for IoT fog computing system," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 5, pp. 3348–3357, 2021.

[4] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," *ETSI White Paper*, vol. 11, no. 11, pp. 1–16, 2015.

[5] H. Yang, Y. Ye, X. Chu, and S. Sun, "Energy efficiency maximization for UAV-enabled hybrid backscatter-harvest-then-transmit communications," *IEEE Transactions on Wireless Communications*1.

[6] Z. Yang, C. Pan, K. Wang, and M. Shikh-Bahaei, "Energy efficient resource allocation in UAV-enabled mobile edge

computing networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 9, pp. 4576–4589, 2019.

[7] L. Zhao, K. Yang, Z. Tan, X. Li, S. Sharma, and Z. Liu, "A novel cost optimization strategy for SDN-enabled UAV-assisted vehicular computation offloading," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 6, pp. 3664–3674, 2021.

[8] Y. Ye, R. Q. Hu, G. Lu, and L. Shi, "Enhance latency-constrained computation in MEC networks using uplink NOMA," *IEEE Transactions on Communications*, vol. 68, no. 4, pp. 2409–2425, 2020.

[9] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1927–1941, 2018.

[10] B. Li, Z. Fei, Y. Zhang, and M. Guizani, "Secure UAV communication networks over 5G," *IEEE Wireless Communications*, vol. 26, no. 5, pp. 114–120, 2019.

[11] Y. Kawamoto, H. Nishiyama, N. Kato, F. Ono, and R. Miura, "Toward future unmanned aerial vehicle networks: architecture, resource allocation and field experiments," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 94–99, 2019.

[12] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: recent advances and future trends," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2241–2263, 2019.

[13] Q. Tang, Z. Fei, B. Li, and Z. Han, "Computation offloading in LEO satellite networks with hybrid cloud and edge computing," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 9164–9176, 2021.

[14] S. Shakoor, Z. Kaleem, D.-T. Do, O. A. Dobre, and A. Jamalipour, "Joint optimization of UAV 3-D placement and path-loss factor for energy-efficient maximal coverage," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9776–9786, 2021.

[15] L. Qian, Y. Wu, J. Ouyang, Z. Shi, B. Lin, and W. Jia, "Latency optimization for cellular assisted mobile edge computing via non-orthogonal multiple access," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5494–5507, 2020.

[16] Z. Wu, B. Li, Z. Fei, Z. Zheng, and Z. Han, "Energy-efficient robust computation offloading for fog-IoT systems," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4417–4425, 2020.

[17] H. Guo and J. Liu, "UAV-enhanced intelligent offloading for internet of things at the edge," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2737–2746, 2020.

[18] 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 Transactions on Vehicular Technology*, vol. 69, no. 5, pp. 5723–5728, 2020.

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

[20] B. Shang and L. Liu, "Mobile-edge computing in the sky: energy optimization for air-ground integrated networks," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7443–7456, 2020.

[21] F. Guo, H. Zhang, H. Ji, X. Li, and V. C. Leung, "Joint trajectory and computation offloading optimization for UAV-assisted MEC with NOMA," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1–6, Paris, France, Apr-May 2019.

[22] S. Batabyal and P. Bhaumik, "Mobility models, traces and impact of mobility on opportunistic routing algorithms: a survey," *IEEE Communication Surveys and Tutorials*, vol. 17, no. 3, pp. 1679–1707, 2015.

[23] L. Shi, Z. Jiang, and S. Xu, "Throughput-aware path planning for UAVs in D2D 5G networks," *Ad Hoc Networks*, vol. 116, p. 102427, 2021.

[24] P. M. Pardalos and S. A. Vavasis, "Quadratic programming with one negative eigenvalue is NP-hard," *Journal of Global Optimization*, vol. 1, no. 1, pp. 15–22, 1991.

[25] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, "Deep reinforcement learning for internet of things: a comprehensive survey," *IEEE Communication Surveys and Tutorials*, vol. 23, no. 3, pp. 1659–1692, 2021.

[26] N. Cheng, F. Lyu, W. Quan et al., "Space/aerial-assisted computing offloading for IoT applications: a learning-based approach," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 5, pp. 1117–1129, 2019.

[27] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4005–4018, 2019.

[28] Z. Chen and X. Wang, "Decentralized computation offloading for multi-user mobile edge computing: a deep reinforcement learning approach," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, 21 pages, 2020.

[29] C. Di Franco and G. Buttazzo, "Energy-aware coverage path planning of UAVs," in *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*, pp. 111–117, Vila Real, Portugal, Apr 2015.

[30] S. Kuutti, S. Fallah, K. Katsaros, M. Dianati, F. Mccullough, and A. Mouzakitis, "A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 829–846, 2018.