Caching on the Sky: A Multiagent Federated Reinforcement Learning Approach for UAV-Assisted Edge Caching

Xuanheng Li, *Member, IEEE*, Jiahong Liu, *Student Member, IEEE*, Xianhao Chen, *Member, IEEE*, Jie Wang, *Senior Member, IEEE*, and Miao Pan, *Senior Member, IEEE*

***Abstract*—As a promising solution to alleviate network congestion, mobile edge caching based on unmanned aerial vehicles (UAVs) has emerged and received intensive research interests, where users could download their desired contents from UAVs with much lower latency. As for the UAV-assisted edge caching, to improve the users' Quality of Experience while reducing the cost on content updating, how to jointly design the trajectory and caching strategy for UAVs is critical. However, considering the dynamics and uncertainty on the traffic environment, as well as the mutual effect among different UAVs, such joint design is nontrivial. In this article, we propose a collaborative joint trajectory and caching scheme for UAV-assisted networks under the dynamic and uncertain traffic environment. Unlike most existing work relying on model-based or single-agent methods, we develop a multiagent deep reinforcement learning (MADRL) approach to obtain the solution, where the specific content demand model is not needed and each UAV would learn the best decision autonomously based on its local observations. It can achieve the adaptive cooperation among different UAVs, while optimizing the overall network performance. Moreover, standing from the perspective on swarm intelligence, we further develop a dynamic clustering federated learning framework on the MADRL algorithm. By performing parameter fusion, each UAV can improve the learning efficiency.**

***Index Terms*—Deep reinforcement learning, federated learning, mobile edge caching, unmanned aerial vehicle (UAV) networks.**

Manuscript received 11 November 2023; revised 27 March 2024; accepted 2 May 2024. Date of publication 15 May 2024; date of current version 23 August 2024. The work of Xuanheng Li and Jiahong Liu was supported in part by the National Natural Science Foundation of China under Grant 62271100; in part by the Science and Technology Program of Liaoning Province under Grant 2023JH2/101700366; in part by the Fundamental Research Funds for the Central Universities under Grant DUT24ZD127; and in part by the Xiaomi Young Talents Program. The work of Xianhao Chen was supported in part by the HKU IDS Research Seed Fund under Grant IDS-RSF2023-0012, and in part by the HKU-SCF FinTech Academy Research and Development Funding. The work of Jie Wang was supported in part by the National Natural Science Foundation of China under Grant 62071081; in part by the Science and Technology Program of Liaoning Province under Grant 2022JH1/10800100, Grant 2023JH2/101300199, and Grant 2023JH26/10300010; and in part by the Dalian Science and Technology Innovation Fund under Grant 2022JJ12WZ057. The work of Miao Pan was supported in part by the U.S. National Science Foundation under Grant CNS-2107057 and Grant CNS-2318664. *(Corresponding author: Xuanheng Li.)*

Xuanheng Li and Jiahong Liu are with the School of Information and Communication Engineering, Dalian University of Technology, Dalian 116024, China (e-mail: xhli@dlut.edu.cn; liujiahong@mail.dlut.edu.cn).

Xianhao Chen is with the Department of Electrical and Electronic Engineering, University of Hong Kong, Hong Kong (e-mail: xchen@eee.hku.hk).

Jie Wang is with the School of Information Science and Technology, Dalian Maritime University, Dalian 116026, China (e-mail: wang_jie@dlmu.edu.cn).

Miao Pan is with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA (e-mail: mpan2@uh.edu).

Digital Object Identifier 10.1109/JIOT.2024.3401219

I. Introduction

WITH the development of telecommunications networks, the global number of smart devices will surge from 7.6 to 17 billion between 2020 and 2030 [1], leading to an explosive growth on the wireless data traffic. Since around 71% of data traffic is content distribution [2] and a large number of requested contents are actually repeated, mobile edge caching was proposed and regarded as a promising technology to alleviate the network congestion under the surging traffic [3], [4], [5], [6], [7], [8]. In the edge caching network, many servers with the caching units are deployed at the edge of the network and prestore some popular contents, so that the users could acquire their desired contents from the adjacent edge servers, instead of the remote cloud servers. In this way, duplicated data transmissions could be avoided and the network backhaul bottleneck could be relieved. Moreover, users could download contents with the reduced transmission delay due to the closer connection [9].

Recently, taking the mobility and flexibility of unmanned aerial vehicles (UAVs) into account, several works have considered to employ UAVs to facilitate edge caching, also known as UAV-assisted edge caching networks [10], [11], [12], [13], [14], [15], [16], [17], [18], [19]. By deploying caching units on UAVs, these cache-enabled UAVs could swiftly fly to the hot spots and provide users with the prestored popular contents. The areas served by UAVs could be regarded as downlink small cells, where users can easily acquire their desired contents from UAVs. Compared with the static ground edge nodes, cache-enabled UAVs can act as flying containers that track the dynamic traffic demand flexibly and provide on-demand services. Furthermore, due to the reliable Line of Sight (LoS) air-to-ground (A2G) links, these flying containers can further reduce the transmission delay and improve the users' Quality of Experience (QoE).

As for an UAV-assisted edge caching network, how to well schedule the UAVs' trajectories, while accurately determining which contents to cache on each UAV and when to update the caching contents, is the key to improve the users' QoE efficiently. Unfortunately, such a joint scheme design is not an easy task.

1) Considering the limited caching capability and numerous potential contents, it is necessary to determine which contents are the most popular ones for caching. However, users' preferences and their content demand are usually time varying, making the content popularity always change as well. Such uncertainty makes the caching strategy design very challenging, especially for some practical cases without precise statistical information of the content demand. In parallel with that, the cost on content updating should be also taken into account, e.g., the spent energy and communication resources. There would be a tradeoff between the users' QoE and updating cost.
2) In general, from the perspective on the whole network performance, UAVs should be deployed in those hot spots to meet more content demand. However, as aforementioned, the content demand of each user is usually time varying. Further considering the users' mobilities, the content demand would be dynamic in both the temporal and spatial domains, making the UAV trajectory design a nontrivial task. In addition, trajectory and caching decisions are closely related with each other. Different trajectories would face different content requests, leading to different caching decisions. Hence, a sophisticated joint design would be necessary to achieve high QoE with low content updating cost.
3) With regard to the UAV group, multiple UAVs should meet diverse popular content requests and serve different hot spots cooperatively to well cover the whole network with better performance. How to achieve a collaborative intelligence is the key here.

Aiming at tackling the aforementioned challenges, several works have investigated the UAV-assisted edge caching network [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]. In [20], [21], [22], [23], [24], and [25], researchers have proposed different trajectory and caching strategies to improve the network performance from different perspectives. Most of them assumed the exact probability distribution of content demand is available, and developed the strategies relying on the model-based optimization approach. However, in some practical cases, the perfect distributional information of content demand might be hardly obtainable. To address this problem, some recent works have embraced the reinforcement learning approach to develop a model-free solution [26], [27], [28], [29], [30]. Although these works did not rely on specific content demand information, the proposed approaches either made each UAV learn its strategy independently or determined all UAVs' strategies together, rather than enable them to made decisions autonomously, where UAVs may not be able to well cooperate with each other adaptively.

In this article, aiming at maximizing the users' satisfaction on the transmission delay while reducing the cost on content updating, we propose a QoE-aware collaborative joint trajectory and caching (COACH) scheme for the UAV-assisted edge caching network, where the dynamics and uncertainty on the traffic environment caused by the varying user demand and user mobility are particularly considered. Taking the mutual effect among different UAVs and uncertain demand into account, we propose a multiagent deep reinforcement learning (MADRL) approach based on the multiagent deep deterministic policy gradient (MADDPG) algorithm. It can make each UAV learn the best decision on the trajectory and cache autonomously based on its local observations, while optimizing the overall network performance. Furthermore, standing from the perspective on swarm intelligence, to make UAVs cooperate with each other, we further develop a dynamic clustering federated learning framework on the MADDPG algorithm (DCF-MADDPG). By performing parameter fusion in an UAV cluster, each UAV could capture the content demand feature of adjacent areas and learn the experiences of other UAVs, so that the learning efficiency of the UAV group could be improved with a faster convergence rate. The main contributions of this article are summarized as follows.

1) Comprehensively considering both the users' QoE and content updating cost, we propose a COACH scheme for the UAV-assisted edge caching network to enable UAVs dynamically serve the hot spots with popular contents.
2) The uncertainty of the traffic environment is particularly considered during the design. Unlike most existing works relying on the model-based optimization approach, we develop a MADRL method built on a centralized training and decentralized executing manner. It can make each UAV learn the best trajectory and caching decision autonomously, while achieving the optimal network performance and collaborative intelligence.
3) Standing from the perspective on swarm intelligence, to facilitate each UAV to learn the experiences of other UAVs, we further develop a dynamic clustering federated learning framework on the MADRL approach. By performing parameter fusion in a cluster, each UAV can improve the learning efficiency and achieve the optimal strategy faster. Furthermore, a rigorous analysis on the convergence of the proposed approach has been presented.

The remainder of this article is organized as follows. Related work are presented in Section II. System model and problem formulation are reviewed in Section III. In Section IV, we introduce the proposed DCF-MADDPG algorithm. Then, simulation result and analysis are drawn in Section V, followed by conclusion in Section VI.

II. Related Work

Considering the superiority and great potential of UAVs, many works have been devoted to the UAV-assisted edge caching network [20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]. Wu et al. [20] developed a dual dynamic adaptive caching algorithm to jointly design the trajectory and caching strategy for serving more users with the reduced delay. In [21], service caching placement, UAV trajectory, UE-UAV association, and task offloading decisions were jointly optimized by Zhou et al. to minimize the service delay. Wang et al. proposed a dynamic pricing and capacity allocation scheme in [22] to maximize UAV service profit. In [23], aiming at minimizing total energy consumption, Gu et al. put forward an energy-aware coded caching strategy for solving the optimal content placement, power allocation, and coverage deployment. In [24], to reduce the content delivery delay, Li et al. studied flight control strategy of UAVs by the distributed optimization and robust mean field game theory. Besides, aiming at minimizing the content delivery delay, Zhang et al. jointly optimized the user association, power allocation, deployment of UAVs, and caching placement by the branch and bound-based algorithm in [25]. Although these works in [20], [21], [22], [23], [24], and [25] have designed the edge caching strategy from different perspectives to improve the network performance, they mainly relied on the optimization method, where the model-based content demand information is assumed to be available. However, in practice, the content demand model might be unavailable.

Considering the demand uncertainty, several works have employed the reinforcement learning approaches to obtain a model-free solution [26], [27], [28], [29], [30]. In [26], to reduce the average transmission delay, Luo et al. proposed a weighted K-means algorithm to deploy UAVs and adopted Q-learning algorithm to learn the content placement policy. Peng and She [27] designed a deep deterministic policy gradient (DDPG)-based solution to obtain the optimal vehicle association and resource allocation decisions. These works in [26] and [27] could make UAVs design the strategy autonomously without model-based demand information. Nevertheless, they adopted single-agent approaches, where each UAV learns its optimal decision independently. In this way, UAVs could not realize swarm intelligence and serve the network with better performance. In [28], considering the QoE satisfaction and the power consumption issue, Anokye et al. adopted the dueling deep Q-network-based algorithm to determine the caching contents and deployment of UAVs. To minimize the long-term content delivery delay, Wang et al. [29] proposed a DDPG-based algorithm to jointly optimize caching placement, user scheduling, and power allocation. In [30], to improve the global cache hit ratio and signal-to-noise ratio, Araf et al. developed a cooperative multiagent actor–critic-based reinforcement learning approach to jointly design the deployment of UAVs and caching strategy. These works in [28], [29], and [30] adopted multiagent approaches. However, it is noteworthy that all UAVs' strategies were determined together, where each UAV obtained its strategy from a controller, rather than made decisions by itself. In this way, the UAVs may not be able to well cooperate with each other adaptively. To exploit the adaptive collaboration of UAVs and fully realize swarm intelligence, we develop a DCF-MADDPG solution.

III. System Model and Problem Formulation

A. Network Model

We consider an UAV-assisted edge caching network as shown in Fig. 1, where multiple cache-enabled UAVs act as flying containers to provide users with edge caching services by carrying certain popular contents on themselves. To be specific, we assume that there are N UAVs launching from certain starting points and flying around the network in a time-slotted way. As for each time slot, it contains two parts, namely flying phase and transmission phase. At the beginning of each time slot, each UAV will determine its flight parameters and caching contents. Then, it will fly to the next location carrying the selected contents accordingly during the flying phase, and hover at that place during the transmission phase to enable users to download their desired contents. If the contents are cached on the nearby UAV, users can download them from the UAV through A2G links. Otherwise, they need to download the contents from the remote base station (BS) through the ground-to-ground (G2G) links. For such an UAV-assisted edge caching network, it is important to well schedule the UAVs' trajectories and the stored caching contents to make them serve more users, so that the overall network QoE could be enhanced. The main notations used for the system model are summarized in Table I.

B. Transmission Model

Next, we will introduce both the G2G and A2G link models, and calculate the transmission delay accordingly.

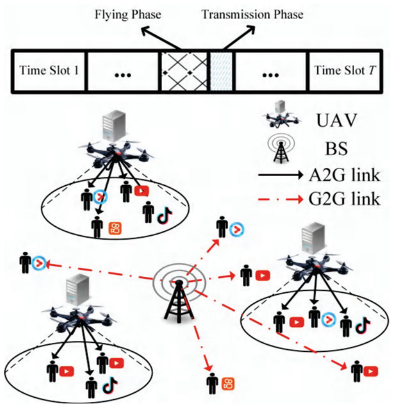


Fig. 1. UAV-assisted edge caching network.

TABLE I
Main Notations Used in This Article

Notation	Definition
N	The number of UAVs
M	The number of content categories
$d_{u,t}^{\mathrm{BS}}$	The distance between the BS and user u at time slot t
$d_{n,u,t}^{\mathrm{UAV}}$	The horizontal distance between UAV n and user u at time slot t
P^{BS}	The transmission power of the BS
P_n^{UAV}	The transmission power of UAV n
$g_{u,t}$	The channel gain from the BS to the user u at time slot t
$\Pr\left(d_{n,u,t}^{\mathrm{UAV}}\right)$	The LoS connection probability of the A2G link between UAV n and user u at time slot t
$l_{n,u,t}$	The average path loss for the A2G link between UAV n and user u at time slot t
C_n	The caching memory size of UAV n
q_m	The size of content m
$\delta_{n,m,t}$	The binary indicator to represent whether or not content m is cached by UAV n at time slot t
$\omega_{n,m,t}$	The binary indicator to represent whether or not UAV n updates the caching unit to store content m at time slot t
$\phi_{n,t}$	The azimuth of UAV n at time slot t
$v_{n,t}$	The flight velocity of UAV n at time slot t

1) Ground-to-Ground Links: For any user u, if it is located without the coverage range of UAVs or UAVs have not carried its desired contents, it needs to download the contents from BS through the G2G links. The transmission rate of the user u at the time slot t can be described as

$$r_{u,t}^{\mathrm{BS}} = W\log_2\left(1 + \frac{P^{\mathrm{BS}} g_{u,t}}{\sigma^2}\right) \tag{1}$$

in which W denotes the bandwidth allocated to each user, P^{BS} and σ^2 are the transmission power of the BS and the noise power, respectively. $g_{u,t}$ is the channel gain from the BS to the user u at the time slot t, which can be expressed based on a simplified model as [31]

$$g_{u,t} = \beta\left(d_{u,t}^{\mathrm{BS}}\right)^{-\alpha} \tag{2}$$

where α and β are the path loss factor and the antenna related parameter, respectively. $d_{u,t}^{\mathrm{BS}}$ represents the distance between the BS and user u, which can be calculated by

$$d_{u,t}^{\mathrm{BS}} = \sqrt{\left(x^{\mathrm{BS}} - x_{u,t}\right)^2 + \left(y^{\mathrm{BS}} - y_{u,t}\right)^2} \tag{3}$$

in which $(x^{\mathrm{BS}}, y^{\mathrm{BS}})$ and $(x_{u,t}, y_{u,t})$ are the coordinates of the BS and user u, respectively. Then, for each content m, we denote its size as q_m and the transmission delay downloading from the BS can be presented as

$$D_{m,u,t}^{\mathrm{BS}} = \frac{q_m}{r_{u,t}^{\mathrm{BS}}}. \tag{4}$$

2) Air-to-Ground Links: If UAVs fly around users carrying with their desired contents, users could download the contents from the nearby UAVs with the reduced transmission delay through the A2G links. Considering the chance of the LoS connectivity, we adopt the standard shadowing model of LoS links associated with the LoS connection probability to model the A2G links. To be specific, for any UAV n and user u, we denote the LoS connection probability of the A2G link between them at the time slot t as [32]

$$\Pr\left(d_{n,u,t}^{\mathrm{UAV}}\right) = \frac{1}{1 + X\exp\left(-Y\left(\arctan\left(\frac{h}{d_{n,u,t}^{\mathrm{UAV}}}\right) - X\right)\right)} \tag{5}$$

where we assume that all the UAVs fly on the same horizontal plane with the same height h. X and Y are constants related to the environment. Similar to (3), $d_{n,u,t}^{\mathrm{UAV}}$ represents the horizontal distance between the UAV n and user u, which can be described as

$$d_{n,u,t}^{\mathrm{UAV}} = \sqrt{\left(x_{n,t}^{\mathrm{UAV}} - x_{u,t}\right)^2 + \left(y_{n,t}^{\mathrm{UAV}} - y_{u,t}\right)^2} \tag{6}$$

where $(x_{n,t}^{\mathrm{UAV}}, y_{n,t}^{\mathrm{UAV}})$ is the coordinates of the UAV n at the time slot t. Then, the average path loss (in dB) for the A2G link between the UAV n and user u at the time slot t can be expressed as

$$\begin{aligned} l_{n,u,t} = {} & 20\log\left(\frac{4\pi f_c}{c}\right) + 20\log\left(\sqrt{h^2 + {d_{n,u.t}^{\mathrm{UAV}}}^2}\right) \\ & + \Pr\left(d_{n,u,t}^{\mathrm{UAV}}\right)\eta^{\mathrm{LoS}} + \left(1 - \Pr\left(d_{n,u.t}^{\mathrm{UAV}}\right)\right)\eta^{\mathrm{NLoS}} \end{aligned} \tag{7}$$

where c and f_c denote the speed of light and the carrier frequency, respectively. η^{LoS} and η^{NLoS} represent the shadowing constants corresponding to the LoS and non-LoS (NLoS) connections depending on the environment [33]. If the path loss is lower than a threshold l_0, we regard the user located within the UAV's coverage, and the transmission rate[1] can be described as

$$r_{n,u,t}^{\mathrm{UAV}} = W\log_2\left(1 + \frac{P_n^{\mathrm{UAV}}}{10^{l_{n,u,t}/10}\sigma^2}\right) \tag{8}$$

where P_n^{UAV} is the transmission power of the UAV n. Similar to (4), for each content m, if user u downloads it from the UAV n at the time slot t, the delay can be expressed as

$$D_{n,m,u,t}^{\mathrm{UAV}} = \frac{q_m}{r_{n,u,t}^{\mathrm{UAV}}}. \tag{9}$$

C. Cache Placement Model

As aforementioned, UAVs are equipped with a set of caching units to store the popular contents to serve users. Due to the limited caching memory size C_n, the amount of cache contents is restricted as

$$\sum_{m=1}^{M} q_m \delta_{n,m,t} \le C_n \ \forall n \ \forall t \tag{10}$$

where the binary indicator $\delta_{n,m,t}$ represents whether or not the content m is cached by the UAV n at the time slot t.

To cache the latest popular contents, UAVs could update their caching contents from the BS at each time slot. However, considering the cost on content updating, such as the energy consumption and resource utilization for downloading the contents, it is necessary to determine whether or not to update caching units. We adopt a binary indicator $\omega_{n,m,t}$ to represent whether or not UAV n updates the caching unit to store the content m at the time slot t. If it does, i.e., $\delta_{n,m,t} = 1$ and $\delta_{n,m,t-1} = 0$, the updating binary indicator $\omega_{n,m,t} = 1$. Otherwise, $\omega_{n,m,t} = 0$. Then, the total cost on content updating can be written as

$$K = \sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T} \hat{k}_m \omega_{n,m,t} \tag{11}$$

where $\hat{k}_m$ denotes the unit cost for updating the content m.

D. UAV Mobility Model

We use the azimuth $\phi_{n,t}$ and the flight velocity $v_{n,t}$ to describe the UAV's flight trajectory, which should satisfy

$$0 \le \phi_{n,t} \le 2\pi \ \forall n \ \forall t \tag{12}$$
$$0 \le v_{n,t} \le V^{\max} \ \forall n \ \forall t. \tag{13}$$

Then, for each UAV n, the location can be expressed as

$$x_{n,t+1}^{\mathrm{UAV}} = x_{n,t}^{\mathrm{UAV}} + v_{n,t}\cos\phi_{n,t} \ \forall n \ \forall t \tag{14}$$
$$y_{n,t+1}^{\mathrm{UAV}} = y_{n,t}^{\mathrm{UAV}} + v_{n,t}\sin\phi_{n,t} \ \forall n \ \forall t. \tag{15}$$

[1]We assume that each UAV is allocated with the orthogonal channels to serve users, and users could download their desired contents from UAVs without interference.

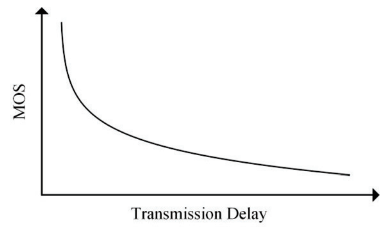


Fig. 2. Relationship between MOS and transmission delay.

Furthermore, during the flying phase, it is necessary to keep UAVs apart from each other to avoid collisions. The distance between any two UAVs should be greater than a threshold $d^{\min}$, which can be expressed as

$$\left\|\left(x_{n,t}^{\mathrm{UAV}}, y_{n,t}^{\mathrm{UAV}}\right)-\left(x_{n',t}^{\mathrm{UAV}}, y_{n',t}^{\mathrm{UAV}}\right)\right\| \geq d^{\min} \ \forall n \neq n' \ \forall t. \quad (16)$$

E. QoE Model

We adopt the mean opinion score (MOS) to measure the users' QoE, which links the transmission delay performance indicator with the subjective user perceived quality as shown in Fig. 2. Based on (4) and (9), we construct the transmission delay-based QoE model to evaluate the satisfaction of the user u at the time slot t as [34]

$$\mathrm{MOS}_{u,t} = \hat{K}_1 \ln\left(\frac{1}{D_{u,t}}\right) + \hat{K}_2 \quad (17)$$

where $\hat{K}_1$ and $\hat{K}_2$ are both the constants. $D_{u,t}$ is the transmission delay corresponding to (4) and (9).

F. Problem Formulation

Assume that the UAVs will serve the network and return to the starting points for energy recharging every T time slots[1]. Considering both the users' QoE and content updating cost, we formulate the proposed COACH scheme as

$$\begin{aligned} \mathrm{P1}: \max_{\boldsymbol{\delta},\mathbf{v},\boldsymbol{\phi}} \ & \lambda_1 \sum_{n=1}^{N}\sum_{u=1}^{U}\sum_{t=1}^{T} \xi_{n,u,t}\mathrm{MOS}_{u,t} - \lambda_2 K \\ \text{s.t.} \ & (10),\ (12)-(16) \end{aligned} \quad (18)$$

where λ_1 and λ_2 are the bias parameters. $\xi_{n,u,t}$ is an indicator to represent whether or not user u is located within the coverage range of the UAV n at the time slot t.

Note that, the formulated problem P1 is an NP-hard problem, which involves coupling continuous decision variables $\mathbf{v}, \boldsymbol{\phi}$ and binary decision variables $\boldsymbol{\delta}$. Furthermore, the exact information of the content demand during the T time slots is required to obtain the solution, which, unfortunately, might be hardly obtainable precisely in advance. Thus, we develop an MADRL solution, which enables each UAV to design the best trajectory and caching strategy autonomously that will optimize the overall network performance. Then, we further develop a DCF-MADDPG algorithm to fully exploit the collaboration of UAVs, so that the learning efficiency of the UAV group could be improved with a faster convergence rate.

[1]Note that energy efficiency is an important issue for the UAV assistance paradigm [35], [36], [37], [38]. Since, the key point of this work is to develop an adaptive joint trajectory and caching scheme for the UAV-assisted edge caching network under the dynamic and uncertain traffic environment, we have not particularly addressed this issue, but considered a classical recharging method as in many existing works.

IV. Multiagent Federated Deep Reinforcement Learning Solution for the COACH Scheme

In this section, we first propose an MADDPG-based deep reinforcement learning solution to make each UAV learn the best decision autonomously based on its local observations to achieve an optimal network overall utility. Then, for fully exploiting the collaboration of UAVs, we further develop a DCF-MADDPG solution [39]. UAVs perform parameter fusion in divided clusters to learn the strategies of adjacent UAVs. In this way, the learning efficiency of the UAV group could be improved, and the convergence rate could be accelerated.

A. Reinforcement Learning Framework

At each time slot t, each UAV n will observe the state $\boldsymbol{s}_{n,t}$ and take an action $\boldsymbol{a}_{n,t}$ according to certain policy $\pi_{n,t}$. Then, it will obtain an immediate reward $r_{n,t}$ by interacting with the environment and traverse to a new state $\boldsymbol{s}_{n,t+1}$. Each UAV n is committed to finding the optimal policy that maximizes the accumulated reward

$$R_{n,t} = \sum_{i=t}^{T} \gamma^{i-t} r_{n,i} \quad (19)$$

where $\gamma \in (0,1)$ is a discount factor. Next, along with the formulated problem P1, we will define the state, action, and reward for the COACH scheme.

1) State: We denote the amount of demand within the coverage range of UAV n on all the M contents at the time slot t as $\boldsymbol{\rho}_{n,t} = [\rho_{1,n,t}, \ldots, \rho_{M,n,t}]^T$. Since, the location of each UAV and the content demand at that place will directly influence the users' QoE and content updating cost, we define the state $\boldsymbol{s}_{n,t}$ observed by each UAV n at the time slot t as

$$\boldsymbol{s}_{n,t} = \left\{x_{n,t}^{\mathrm{UAV}}; y_{n,t}^{\mathrm{UAV}}; \boldsymbol{\rho}_{n,t}\right\}. \quad (20)$$

2) Action: As the decisions to make, we define the action of each UAV n at the time slot t as

$$\boldsymbol{a}_{n,t} = \left\{\boldsymbol{\delta}_{n,t}, v_{n,t}, \phi_{n,t}\right\} \quad (21)$$

in which $\boldsymbol{\delta}_{n,t} = [\delta_{1,n,t}, \ldots, \delta_{M,n,t}]^T$ is the binary indicator associated with the caching decisions satisfying the constraint (10). $v_{n,t}$ and $\phi_{n,t}$ denote the flight velocity and azimuth, respectively, for the trajectory decision.

3) Reward: The reward $r_{n,t}$ is the feedback to evaluate the action $\boldsymbol{a}_{n,t}$ choosing in state $\boldsymbol{s}_{n,t}$. Taking the MOS, cost on content updating and the collision avoidance into account, we define the reward for the UAV n at the time slot t as

$$\begin{aligned} r_{n,t} = \ & \lambda_1' \sum_{u=1}^{U} \xi_{n,u,t}\mathrm{MOS}_{u,t} - \lambda_2' \sum_{m=1}^{M} \hat{k}_m \omega_{n,m,t} \\ & - \lambda_3' \frac{\mu}{\left\|\left(x_{n,t}^{\mathrm{UAV}}, y_{n,t}^{\mathrm{UAV}}\right)-\left(x_{n',t}^{\mathrm{UAV}}, y_{n',t}^{\mathrm{UAV}}\right)\right\|} \end{aligned} \quad (22)$$

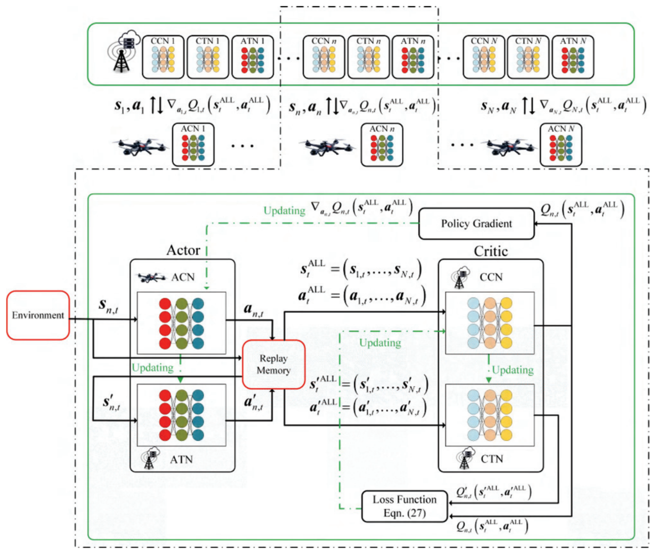


Fig. 3. MADDPG algorithm for the COACH scheme.

where λ'_1, λ'_2, and λ'_3 are the bias parameters. μ is the penalty coefficient. The first penalty term $\sum_{m=1}^{M}\hat{k}_m\omega_{n,m,t}$ can avoid frequent content updating to reduce the updating cost. The second penalty term $(\mu/[\|(x_{n,t}^{\text{UAV}}, y_{n,t}^{\text{UAV}})-(x_{n',t}^{\text{UAV}}, y_{n',t}^{\text{UAV}})\|])$ can avoid collisions among different UAVs because a small distance on the denominator will make the reward a very small negative value.

B. MADDPG Algorithm

The proposed MADDPG-based deep reinforcement learning method [40] is shown in Fig. 3. It contains four neural networks, namely actor current network (ACN), actor target network (ATN), critic current network (CCN), and critic target network (CTN). The first ACN is placed on each UAV for the action decision making and the other three neural networks are placed at the central server to evaluate the strategy that each UAV makes from the perspective on the whole network, which will be adopted for training the ACN. The MADDPG method will be implemented in a centralized training and decentralized executing manner.

1) Actor Network: As aforementioned, the ACN is deployed on the UAVs, so that they could determine the actions by themselves based on the local observations. Specifically, for each UAV n, the ACN $\pi_{n,t}(\boldsymbol{s}_{n,t}|\boldsymbol{\varphi}_n.)$ is used to determine which action is chosen. The input of it is the current observed state $\boldsymbol{s}_{n,t}$ and the output is the $\pi_{n,t}(\boldsymbol{s}_{n,t})$. Note that, the output of the ACN $\pi_{n,t}(\boldsymbol{s}_{n,t})$ is a continuous vector, while the caching decision $\boldsymbol{\delta}_{n,t}$ should be a discrete one. Hence, considering the constraint (10), we discretize the continuous output into $\mathbf{C}_M^{C_n}$ levels, corresponding to the $\mathbf{C}_M^{C_n}$ possible cases, and obtain the discrete action $\boldsymbol{a}_{n,t}$ by determining which level $\pi_{n,t}(\boldsymbol{s}_{n,t})+\mathcal{N}(0,\hat{\sigma}^2)$ belongs to, where $\mathcal{N}(0,\hat{\sigma}^2)$ is a Gaussian noise with the variance $\hat{\sigma}^2$ added on the output for the environment exploration. Then, it will transit to the next state $\boldsymbol{s}_{n,t+1}$ and the variance $\hat{\sigma}^2$ decreases. The UAV will obtain a reward $r_{n,t}$ according to the feedback from the environment. The experience tuple $\{\boldsymbol{s}_{n,t}, \boldsymbol{a}_{n,t}, r_{n,t}, \boldsymbol{s}_{n,t+1}\}$ will be uploaded to the central server and stored in a replay memory for training the neural networks. When the surrounding environment changes dramatically, UAVs will explore the environment to capture the new demand feature. To be specific, if $\sum_{i=0}^{I-1} r_{n,t-i} \le r_{n,t-I}$, $\hat{\sigma}^2$ will be initialized.

The ATN $\pi'_{n,t}(\boldsymbol{s}'_{n,t}|\boldsymbol{\varphi}'_n)$ has the same network structure as the ACN, which is deployed on the central sever and employed for training the critic networks. The parameters of it $\boldsymbol{\varphi}'_n$ are copied from the ACN every T^a time slots in a soft updating way as

$$\boldsymbol{\varphi}'_n \leftarrow \tau^a\boldsymbol{\varphi}_n + \left(1-\tau^a\right)\boldsymbol{\varphi}'_n \tag{23}$$

where $\tau^a \ll 1$ is the forgetting factor of the ATN.

The parameters of the ACN $\boldsymbol{\varphi}_n$ are updated based on the gradient ascent method, i.e.,

$$\boldsymbol{\varphi}_{n,t+1} = \boldsymbol{\varphi}_{n,t} + \lambda^a\nabla_{\boldsymbol{\varphi}_n}J\left(\boldsymbol{\varphi}_n\right) \tag{24}$$

where λ^a denotes learning rate of the ACN. $\nabla_{\boldsymbol{\varphi}_n}J(\boldsymbol{\varphi}_n)$ is the policy gradient in terms of $\boldsymbol{\varphi}_n$ expressed as

$$\nabla_{\boldsymbol{\varphi}_n}J\left(\boldsymbol{\varphi}_n\right) = \mathbb{E}\Big[\nabla_{\boldsymbol{a}_{n,t}}Q_{n,t}\Big(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}}\Big)\nabla_{\boldsymbol{\varphi}_n}\pi_{n,t}\left(\boldsymbol{s}_{n,t}\right)\Big] \tag{25}$$

where $\nabla_{\boldsymbol{a}_{n,t}}Q_{n,t}(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}})$ is the policy gradient of the CCN in terms of the action $\boldsymbol{a}_{n,t}$ that can be downloaded from the central server.

2) Critic Network: The two critic networks, namely CCN $Q_{n,t}(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}}|\boldsymbol{\theta}_n)$ and CTN $Q'_{n,t}(\boldsymbol{s}_t'^{\text{ALL}}, \boldsymbol{a}_t'^{\text{ALL}}|\boldsymbol{\theta}'_n)$, are placed at the central server. The former one is used to evaluate how good all the UAVs' actions are in the current states, and the latter one is employed for updating the CCN's parameters. Both two networks have the same structure. For CCN, the input is the state vector, including all the UAVs' submitted states, i.e., $\boldsymbol{s}_t^{\text{ALL}} = (\boldsymbol{s}_{1,t}, \ldots, \boldsymbol{s}_{N,t})$, and the action vector, including all the UAVs' submitted actions, i.e., $\boldsymbol{a}_t^{\text{ALL}} = (\boldsymbol{a}_{1,t}, \ldots, \boldsymbol{a}_{N,t})$. The output is the Q-value of the state-action pair for all the UAVs. For each UAV n, the policy gradient of the corresponding CCN $\nabla_{\boldsymbol{a}_{n,t}}Q_{n,t}(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}})$ will return to the UAV for training its local ACN as in (24).

The parameters of CCN are updated as

$$\boldsymbol{\theta}_{n,t+1} = \boldsymbol{\theta}_{n,t} - \lambda^c\nabla_{\boldsymbol{\theta}_n}J(\boldsymbol{\theta}_n) \tag{26}$$

where λ^c is the learning rate of the CCN. $J(\boldsymbol{\theta}_n)$ is the loss function defined as

$$J(\boldsymbol{\theta}_n) = \mathbb{E}\bigg[\Big(\hat{y}_{n,t} - Q_{n,t}\Big(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}}\Big)\Big)^2\bigg]. \tag{27}$$

$\hat{y}_{n,t}$ is the target value calculated as

$$\hat{y}_{n,t} = r_{n,t} + \gamma Q'_{n,t}\Big(\boldsymbol{s}_t'^{\text{ALL}}, \boldsymbol{a}_t'^{\text{ALL}}\Big) \tag{28}$$

where $Q'_{n,t}(\boldsymbol{s}_t'^{\text{ALL}}, \boldsymbol{a}_t'^{\text{ALL}})$ is the output of the CTN. $\boldsymbol{a}_t'^{\text{ALL}}$ is the output of the ATN under the input $\boldsymbol{s}_t'^{\text{ALL}}$, i.e., $\boldsymbol{a}_t'^{\text{ALL}} = (\pi'_{1,t}(\boldsymbol{s}'_{1,t}), \ldots, \pi'_{N,t}(\boldsymbol{s}'_{N,t}))$. The parameter updating can be achieved by sampling a mini-batch experience data from the replay memory. Similar to the relationship between the ACN and ATN, the parameters of CTN $\boldsymbol{\theta}'_n$ are copied from the CCN every T^c time slots as

$$\boldsymbol{\theta}'_n \leftarrow \tau^c\boldsymbol{\theta}_n + \left(1-\tau^c\right)\boldsymbol{\theta}'_n \tag{29}$$

where $\tau^c \ll 1$ is the forgetting factor of the CTN.

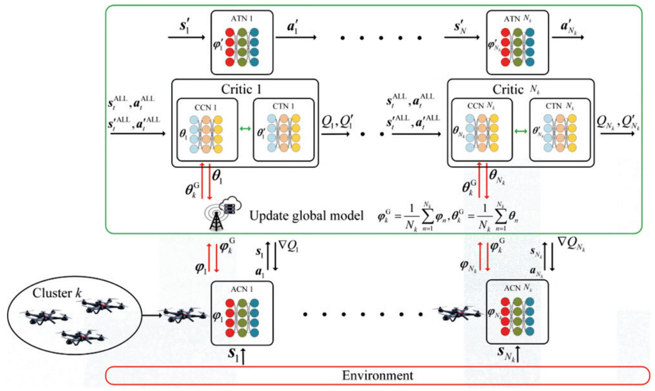


Fig. 4. Dynamic clustering federated MADDPG algorithm for the COACH scheme.

C. Dynamic Clustering Federated MADDPG Algorithm

Since all UAVs need to learn the optimal strategy in the same network environment, to further exploit the collaboration of UAVs, we incorporate the concept of federated learning with dynamic clustering and develop a DCF-MADDPG-based solution standing from the perspective on swarm intelligence as shown in Fig. 4, where each UAV could learn the experiences of others. Specifically, to facilitate adjacent UAVs to share the surrounding environment information, the central server will first classify UAVs into different clusters based on the UAVs' locations. Then, UAVs perform parameter fusion in divided clusters to update network parameters, which could learn the strategies of other UAVs. In this way, each UAV can improve the learning efficiency.

Each UAV n will upload location $(x_{n,t}^{\text{UAV}}, y_{n,t}^{\text{UAV}})$ to the central server every $\tilde{T}$ time slots for determining the dynamic clusters. After obtaining all UAVs' locations, the central server is committed to categorizing neighbor UAVs into the same cluster to make them learn strategies from each other. If the distance between the UAV n and n' is less than $\hat{d}$, they will be divided into the same cluster to learn the action and evaluation strategies of neighbor UAVs.

When UAVs are classified into different clusters, they will perform parameter fusion. As for the ACN equipped on the UAVs, each UAV n will upload the local model $\boldsymbol{\varphi}_n$ to the central server every $\tilde{T}$ time slots. For the cluster k, the central server performs parameter fusion as

$$\boldsymbol{\varphi}_k^G = \frac{1}{N_k}\sum_{n=1}^{N_k}\boldsymbol{\varphi}_n \tag{30}$$

where N_k is the number of UAVs in the cluster k. Then, the central server sends back the global model $\boldsymbol{\varphi}_k^G$ to UAVs belonging to the cluster k for updating their ACN parameters accordingly. Each UAV will keep learning the environment starting from the new global model parameters $\boldsymbol{\varphi}_k^G$ and make actions accordingly.

Similarly, as for the CCNs at the central server, they fuse the global model every $\tilde{T}$ time slots as

$$\boldsymbol{\theta}_k^G = \frac{1}{N_k}\sum_{n=1}^{N_k}\boldsymbol{\theta}_n. \tag{31}$$

Then, each CCN belonging to the cluster k updates its parameters to $\boldsymbol{\theta}_k^G$ and keep running the MADDPG algorithm.

D. Implementation Based on Centralized Training and Decentralized Executing

Next, we will summarize the whole implementation process of the proposed DCF-MADDPG algorithm. The ACN is placed on each UAV for the action decision making, and the other three neural networks are placed at the central server to evaluate the strategy that each UAV makes from the perspective on the whole network, which will be adopted for training the ACN. The four neural networks would be trained offline with the historical data at first, and then executed online. During the online execution, the networks will be trained as well according to the new observations, and the network parameters of ACN and CCN will be updated every $\hat{T}^a$ and $\hat{T}^c$ time slots, which will be copied to ATN and CTN every T^a and T^c time slots. Besides, every $\tilde{T}$ time slots, the parameters of both the ACN and CCN will be fused at the central server for federated learning.

To be specific, at each time slot t, each UAV n will first observe the current state $\boldsymbol{s}_{n,t}$, including its location $(x_{n,t}^{\text{UAV}}, y_{n,t}^{\text{UAV}})$ and the content demand at that place $\boldsymbol{\rho}_{n,t}$. Then, it will choose the action based on the output of the ACN $\pi_{n,t}(\boldsymbol{s}_{n,t}) + \mathcal{N}(0, \hat{\sigma}^2)$. After performing the action, it will fly to a new location carrying certain determined contents, and get a reward $r_{n,t}$ according to the users' QoE and updating cost. Each UAV n will submit its experience tuple $\{\boldsymbol{s}_{n,t}, \boldsymbol{a}_{n,t}, r_{n,t}, \boldsymbol{s}_{n,t+1}\}$ to the central server. Note that, the proposed DCF-MADDPG algorithm is implemented in a distributed manner, where UAVs execute the action without the control of the central server.

After obtaining all UAVs' information, the central server will send back the policy gradient of CCN $\nabla_{\boldsymbol{a}_{n,t}} Q_{n,t}(\boldsymbol{s}_t^{\text{ALL}}, \boldsymbol{a}_t^{\text{ALL}})$ to each UAV n every $\hat{T}^a$ time slots, which will be adopted for updating the parameters of ACN as (24), and the CCNs will be trained as well based on (26) every $\hat{T}^c$ time slots. The central server determines the dynamic clusters based on the UAVs' locations every $\tilde{T}$ time slots. Then, the parameters of both the ACN and CCN will be fused at the central server, and the parameters of ACN and CCN will be copied to ATN and CTN every T^a and T^c time slots, respectively. The aforementioned training process is centralized at the central server. The whole process is summarized in Algorithm 1.

E. Convergence Analysis

Next, according to the analysis in [41], we will prove the convergence of the proposed DCF-MADDPG algorithm. We take the actor network as an example. The convergence of the critic network could be proved in similar way. We first introduce a single-agent approach, where the strategies of all UAVs are trained together in the centralized controller. The parameter vector of the single-agent approach is denoted as $\boldsymbol{\mu}_{\hat{l}}(t)$, $t \in [(\hat{l}-1)\tilde{T}, \hat{l}\tilde{T}]$. It updates from the $(\hat{l}-1)$-th global

Algorithm 1 DCF-MADDPG Algorithm

```
1: Initialize: $\varphi$, $\varphi'$, $\theta$, $\theta'$, $\tau^{a}$, $\tau^{c}$, $\lambda^{a}$, $\lambda^{c}$, $\gamma$, $\hat{\sigma}^2$.
2: for UAV $n$ in $N$ do
3:   Initialize state $s_{n,t}$.
4: end for
5: for episode $= 1, 2, \ldots$ do
6:   for $t = 1, 2, \ldots, T$ do
7:     for UAV $n$ in $N$ do
8:       $\hat{\sigma}^2 \leftarrow 0.9999 \times \hat{\sigma}^2$.
9:       Execute action $a_{n,t}$ according to $\pi_{n,t}(s_{n,t}) + \mathcal{N}(0, \hat{\sigma}^2)$.
10:      Obtain reward $r_{n,t}$ according to (22).
11:      if $\sum_{i=0}^{I-1} r_{n,t-i} \leq r_{n,t-I}$ then
12:        Initialize $\hat{\sigma}^2$.
13:      end if
14:      Transit to the next state $s_{n,t+1}$.
15:      Store transition $\{s_{n,t}, a_{n,t}, r_{n,t}, s_{n,t+1}\}$ in the replay memory unit.
16:      Sample a mini-bach from the memory unit.
17:      if $t$ mod $\hat{T}^{a} == 0$ then
18:        Update ACN according to (24).
19:      end if
20:      if $t$ mod $\hat{T}^{c} == 0$ then
21:        Update CCN according to (26).
22:      end if
23:      if $t$ mod $T^{a} == 0$ then
24:        Update ATN according to (23).
25:      end if
26:      if $t$ mod $T^{c} == 0$ then
27:        Update CTN according to (29).
28:      end if
29:      if $t$ mod $\tilde{T} == 0$ then
30:        Upload location $(x_{n,t}^{\mathrm{UAV}}, y_{n,t}^{\mathrm{UAV}})$ to central server.
31:        Upload local model $\varphi_n$ to central server.
32:        Perform parameter fusion according to (30) and (31).
33:      end if
34:    end for
35:  end for
36: end for
```

model parameter $\varphi^G((\hat{l}-1)\tilde{T})$ and follows a centralized gradient descent as:

$$\mu_{\hat{l}}(t) = \mu_{\hat{l}}(t-1) - \hat{\lambda} J\big(\mu_{\hat{l}}(t-1)\big) \tag{32}$$

where $\hat{\lambda}$ is the learning rate. We denote $J(\mu_{\hat{l}}(t)) - J(\varphi^*)$ as $\hat{\theta}_{\hat{l}}(t)$, where φ^* is the optimal actor network parameter that minimizes the loss function $J(\cdot)$. Based on the parameter vector $\mu_{\hat{l}}(t)$, we will analyse the gap between $J(\varphi^G(\hat{L}\tilde{T}))$ and $J(\varphi^*)$ to show the convergence of the global parameter after $\hat{L}\tilde{T}$ time slots.

Then, we present some definitions as follows.

Definition 1: For any actor network parameter $\varphi_n(t)$ and global model parameter $\varphi^G(t)$, the gradient divergence ψ_n is defined as

$$\left\|\nabla J\big(\varphi_n(t)\big) - \nabla J\Big(\varphi^G(t)\Big)\right\| \leq \psi_n. \tag{33}$$

Based on ψ_n, the global gradient divergence ψ^G is defined as

$$\psi^G = \frac{1}{N}\sum_{n=1}^{N}\psi_n. \tag{34}$$

Definition 2: We define the learning rate of centralized gradient descent $\hat{\lambda}$ in (32) satisfying

$$\hat{\lambda} \leq \frac{1}{\hat{\beta}} \tag{35}$$

and

$$\hat{\lambda}\tau - \frac{\hat{\rho} g(\tilde{T})}{\tilde{T}\varepsilon^2} > 0 \tag{36}$$

where $\hat{\beta}$ is a constant making the loss function $J(\cdot)$ $\hat{\beta}$-smooth as

$$\left\|\nabla J\big(\bar{\varphi}_n(t)\big) - \nabla J\big(\hat{\varphi}_n(t)\big)\right\| \leq \hat{\beta}\left\|\bar{\varphi}_n(t) - \hat{\varphi}_n(t)\right\| \tag{37}$$

for any $\bar{\varphi}_n(t)$ and $\hat{\varphi}_n(t)$ [42]. $\hat{\rho}$ is a constant making $J(\cdot)$ $\hat{\rho}$-Lipschitz as

$$\left\|J\big(\bar{\varphi}_n(t)\big) - J\big(\hat{\varphi}_n(t)\big)\right\| \leq \hat{\rho}\left\|\bar{\varphi}_n(t) - \hat{\varphi}_n(t)\right\| \tag{38}$$

for any $\bar{\varphi}_n(t)$ and $\hat{\varphi}_n(t)$. $\tau = \hat{\omega}(1 - ([\hat{\lambda}\hat{\beta}]/2))$ and $\hat{\omega} = \min_t(1[\|\mu_{\hat{l}}(t) - \varphi^*\|^2])$. The function $g(x)$ is presented as

$$g(x) = \frac{\psi^G}{\hat{\beta}}\left(\left(\hat{\lambda}\hat{\beta}+1\right)^x - 1\right) - \hat{\lambda}\psi^G x. \tag{39}$$

ε is a constant satisfying

$$J\Big(\mu_{\hat{l}}\Big(\hat{l}\tilde{T}\Big)\Big) - J(\varphi^*) \geq \varepsilon \;\forall \hat{l} \tag{40}$$

and

$$J\Big(\varphi^G\Big(\hat{L}\tilde{T}\Big)\Big) - J(\varphi^*) \geq \varepsilon. \tag{41}$$

Next, we will first analyse the gap between $\hat{\theta}_{\hat{l}}(t+1)$ and $\hat{\theta}_{\hat{l}}(t)$, and then prove the convergence of the global parameter. According to (37), since $J(\cdot)$ is $\hat{\beta}$-smooth, we have (42), shown at the bottom of the page. Recalling $J(\mu_{\hat{l}}(t)) - J(\varphi^*) = \hat{\theta}_{\hat{l}}(t)$, (42) is equivalent to

$$\hat{\theta}_{\hat{l}}(t+1) \leq \hat{\theta}_{\hat{l}}(t) - \hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right)\left\|\nabla J\big(\mu_{\hat{l}}(t)\big)\right\|^2. \tag{43}$$

$$\begin{aligned} J\big(\mu_{\hat{l}}(t+1)\big) - J\big(\mu_{\hat{l}}(t)\big) &\leq \nabla J\big(\mu_{\hat{l}}(t)\big)^T\big(\mu_{\hat{l}}(t+1) - \mu_{\hat{l}}(t)\big) + \frac{\hat{\beta}}{2}\left\|\mu_{\hat{l}}(t+1) - \mu_{\hat{l}}(t)\right\|^2 \\ &\leq -\hat{\lambda}\nabla J\big(\mu_{\hat{l}}(t)\big)^T\nabla J\big(\mu_{\hat{l}}(t)\big) + \frac{\hat{\beta}\hat{\lambda}^2}{2}\left\|\nabla J\big(\mu_{\hat{l}}(t)\big)\right\|^2. \end{aligned} \tag{42}$$

Since $J(\cdot)$ is a convex function, we have

$$\hat{\theta}_{\hat{l}}(t) \le \nabla J\big(\boldsymbol{\mu}_{\hat{l}}(t)\big)^T\big(\boldsymbol{\mu}_{\hat{l}}(t) - \boldsymbol{\varphi}^*\big). \tag{44}$$

By Cauchy–Schwarz Inequality, (33) could be transformed into

$$\frac{\hat{\theta}_{\hat{l}}(t)}{\big\|\boldsymbol{\mu}_{\hat{l}}(t) - \boldsymbol{\varphi}^*\big\|} \le \big\|\nabla J\big(\boldsymbol{\mu}_{\hat{l}}(t)\big)\big\|. \tag{45}$$

Based on (32) and (34), we could obtain

$$\hat{\theta}_{\hat{l}}(t+1) \le \hat{\theta}_{\hat{l}}(t) - \frac{\hat{\lambda}\Big(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\Big)\hat{\theta}_{\hat{l}}(t)^2}{\big\|\boldsymbol{\mu}_{\hat{l}}(t) - \boldsymbol{\varphi}^*\big\|^2}. \tag{46}$$

Since, $-\hat{\omega} \ge -(1/[\|\boldsymbol{\mu}_{\hat{l}}(t) - \boldsymbol{\varphi}^*\|^2])$, we could reformulate (46) into

$$\hat{\theta}_{\hat{l}}(t+1) \le \hat{\theta}_{\hat{l}}(t) - \hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right)\hat{\theta}_{\hat{l}}(t)^2. \tag{47}$$

It could be rewritten as

$$\frac{1}{\hat{\theta}_{\hat{l}}(t+1)} - \frac{1}{\hat{\theta}_{\hat{l}}(t)} \ge \frac{\hat{\omega}\hat{\lambda}\Big(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\Big)\hat{\theta}_{\hat{l}}(t)}{\hat{\theta}_{\hat{l}}(t+1)}. \tag{48}$$

Since, $\hat{\theta}_{\hat{l}}(t+1) \le \hat{\theta}_{\hat{l}}(t)$ from (32), (48) could be further reformulated into

$$\frac{1}{\hat{\theta}_{\hat{l}}(t+1)} - \frac{1}{\hat{\theta}_{\hat{l}}(t)} \ge \hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right). \tag{49}$$

So far, we have presented the gap between $\hat{\theta}_{\hat{l}}(t+1)$ and $\hat{\theta}_{\hat{l}}(t)$. Based on the gap, we will prove that $J(\boldsymbol{\varphi}^G(\hat{L}\tilde{T})) - J(\boldsymbol{\varphi}^*)$ converges to a constant. For (49), summing up t from $(\hat{l}-1)\tilde{T}$ to $\hat{l}\tilde{T} - 1$, we have

$$\frac{1}{\hat{\theta}_{\hat{l}}\big(\hat{l}\tilde{T}\big)} - \frac{1}{\hat{\theta}_{\hat{l}}\Big(\big(\hat{l}-1\big)\tilde{T}\Big)} \ge \tilde{T}\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right). \tag{50}$$

For (50), summing up all $\hat{l} = 1, 2, \ldots, \hat{L}$, we achieve that

$$\sum_{\hat{l}=1}^{\hat{L}}\left(\frac{1}{\hat{\theta}_{\hat{l}}\big(\hat{l}\tilde{T}\big)} - \frac{1}{\hat{\theta}_{\hat{l}}\Big(\big(\hat{l}-1\big)\tilde{T}\Big)}\right) \ge \hat{L}\tilde{T}\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right) \tag{51}$$

which can be further rewritten as (52), shown at the bottom of the page. Then, since $\boldsymbol{\mu}_{\hat{l}+1}(\hat{l}\tilde{T}) = \boldsymbol{\varphi}^G(\hat{l}\tilde{T})$, referring to [41, Th. 1], the conclusion could be presented as

$$\Big\|\boldsymbol{\mu}_{\hat{l}}\big(\hat{l}\tilde{T}\big) - \boldsymbol{\mu}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big)\Big\| \le g\big(\tilde{T}\big). \tag{53}$$

Since, the loss function $J(\cdot)$ is $\hat{\rho}$-Lipschitz, we could obtain (54), shown at the bottom of the page. We combine (53) with (54) to obtain that

$$\hat{\theta}_{\hat{l}}\big(\hat{l}\tilde{T}\big) - \hat{\theta}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big) \ge -\hat{\rho}g\big(\tilde{T}\big). \tag{55}$$

Considering (40) and $\hat{\theta}_{\hat{l}}(t+1) \le \hat{\theta}_{\hat{l}}(t)$ from (32), we draw the conclusion as

$$\theta_{\hat{l}}\big(\hat{l}\tilde{T}\big)\theta_{\hat{l}+1}\big(\hat{l}\tilde{T}\big) \ge \varepsilon^2. \tag{56}$$

Based on (55) and (56), (52) could be transformed into

$$\frac{1}{\hat{\theta}_L\big(\hat{L}\tilde{T}\big)} - \frac{1}{\hat{\theta}_1(0)} \ge \hat{L}\tilde{T}\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right) - \big(\hat{L}-1\big)\frac{\hat{\rho}g\big(\tilde{T}\big)}{\varepsilon^2}. \tag{57}$$

Then, based on (40) and (41), we have

$$-\frac{1}{\Big(J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big) - J(\boldsymbol{\varphi}^*)\Big)\hat{\theta}_{\hat{L}}\big(\hat{L}\tilde{T}\big)} \ge -\frac{1}{\varepsilon^2}. \tag{58}$$

Based on (55) and (58), we could obtain (59), shown at the bottom of the page. Summing up (57) and (59), we get (60), shown at the bottom of the page. Since, $\hat{\theta}_1(0) > 0$, (60) could be transformed into

$$\frac{1}{J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big) - J(\boldsymbol{\varphi}^*)} \ge \hat{L}\tilde{T}\left(\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right) - \frac{\hat{\rho}g\big(\tilde{T}\big)}{\tilde{T}\varepsilon^2}\right). \tag{61}$$

$$\frac{1}{\hat{\theta}_{\hat{L}}\big(\hat{L}\tilde{T}\big)} - \frac{1}{\hat{\theta}_1(0)} - \hat{L}\tilde{T}\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right) \ge \sum_{\hat{l}=1}^{\hat{L}-1}\left(\frac{\hat{\theta}_{\hat{l}}\big(\hat{l}\tilde{T}\big) - \hat{\theta}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big)}{\hat{\theta}_{\hat{l}}\big(\hat{l}\tilde{T}\big)\hat{\theta}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big)}\right) \tag{52}$$

$$\Big\|J\Big(\boldsymbol{\mu}_{\hat{l}}\big(\hat{l}\tilde{T}\big)\Big) - J\Big(\boldsymbol{\mu}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big)\Big)\Big\| \le \hat{\rho}\Big\|\boldsymbol{\mu}_{\hat{l}}\big(\hat{l}\tilde{T}\big) - \boldsymbol{\mu}_{\hat{l}+1}\big(\hat{l}\tilde{T}\big)\Big\| \tag{54}$$

$$\frac{1}{J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big) - J(\boldsymbol{\varphi}^*)} - \frac{1}{\hat{\theta}_{\hat{L}}\big(\hat{L}\tilde{T}\big)} = \frac{J\Big(\boldsymbol{\mu}_{\hat{L}}\big(\hat{L}\tilde{T}\big)\Big) - J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big)}{\Big(J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big) - J(\boldsymbol{\varphi}^*)\Big)\hat{\theta}_{\hat{L}}\big(\hat{L}\tilde{T}\big)} \ge -\frac{\hat{\rho}g\big(\tilde{T}\big)}{\varepsilon^2} \tag{59}$$

$$\frac{1}{J\Big(\boldsymbol{\varphi}^G\big(\hat{L}\tilde{T}\big)\Big) - J(\boldsymbol{\varphi}^*)} - \frac{1}{\hat{\theta}_1(0)} \ge \hat{L}\tilde{T}\hat{\omega}\hat{\lambda}\left(1 - \frac{\hat{\beta}\hat{\lambda}}{2}\right) - \hat{L}\frac{\hat{\rho}g\big(\tilde{T}\big)}{\varepsilon^2} \tag{60}$$

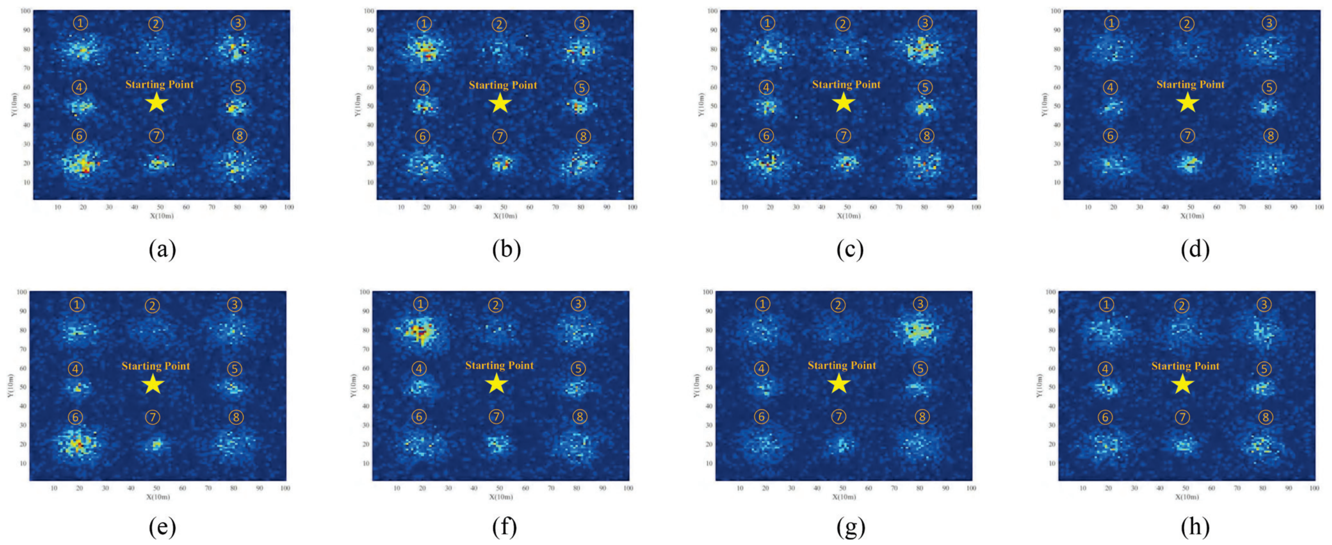


Fig. 5. Different demand distributions of different contents. (a) Demand distribution on content 1. (b) Demand distribution on content 2. (c) Demand distribution on content 3. (d) Demand distribution on content 4. (e) Demand distribution on content 5. (f) Demand distribution on content 6. (g) Demand distribution on content 7. (h) Demand distribution on content 8.

From (61), we could present the gap between $J(\boldsymbol{\varphi}^G(\hat{L}\tilde{T}))$ and $J(\boldsymbol{\varphi}^*)$ as

$$J\left(\boldsymbol{\varphi}^G\left(\hat{L}\tilde{T}\right)\right) - J\left(\boldsymbol{\varphi}^*\right) \leq \frac{1}{\hat{L}\tilde{T}\left(\hat{\omega}\hat{\lambda}\left(1-\frac{\hat{\beta}\hat{\lambda}}{2}\right)-\frac{\hat{\rho}g(\tilde{T})}{\tilde{T}\varepsilon^2}\right)}. \quad (62)$$

So far, we have proved that $J(\boldsymbol{\varphi}^G(\hat{L}\tilde{T}))-J(\boldsymbol{\varphi}^*)$ converges to a constant. Since, the global model parameter of each dynamic cluster could converge, the convergence of the proposed DCF-MADDPG algorithm could be guaranteed.

V. Simulation Result and Discussion

We consider a 1000×1000 m^2 area as the simulation scenario, where $N = 3$ UAVs are deployed to serve the entire network and eight hot spots exist with huge traffic demand. In each hot spot, we randomly deploy $[200, 400]$ users based on a normal distribution, and let them prefer to download two specific contents from all the $M = 8$ contents. At each time slot, each user has one content request, and it either stays in its current position or move to another hot spot at a speed of 1 m/s. Thousand users are randomly deployed in other areas with a random movement with a speed of 1 m/s, and each one has one content request that is randomly chosen from all the $M = 8$ contents. At each time slot, all users in the network have a 5% probability of changing their content requests. Fig. 5 presents the demand distribution on different contents at one time slot.

One time slot is set as 1 s and 500 time slots are defined as one episode. Each UAV returns to the starting point for power charging at the end of every episode. The BS is located in the corner of this area and the number of orthogonal channels is set as 300. The key simulation parameters of the communication and caching models are set following the common rules as in [13], [33], [43], [44], and [45], which are shown in Table II. As for the hyper-parameters settings, we let bias parameter $\lambda_1' = 1$, $\lambda_2' = 1$, $\lambda_3' = 1$, discount factor $\gamma = 0.9$, exploration noise $\hat{\sigma}^2 = 1$, exploration related parameter $I = 5$, and model fusion cycle $\tilde{T} = 25$. The memory unit size is 20 000 and mini-batch size is 256. Other detailed parameters are described in Tables III and IV. Under the parameter settings, we compare our proposed DCF-MADPPG algorithm with the following benchmarks to evaluate the performance.

TABLE II
Parameter Settings

Parameters	Values
size of candidate contents, q	10 Mb [43]
memory size, C	20 Mb [43]
maximum flight velocity, $V^{\max}$	5 m/s
path loss factor, α	4 [13]
communications related parameter, β	4 [13]
environment constant, X	9.61 [33]
environment constant, Y	0.16 [33]
cluster distance, $\hat{d}$	500 m
unit cost for updating contents, $\hat{k}$	200
penalty coefficient, μ	20
bandwidth, W	10 MHz
height, h	50 m
MOS constant, $\hat{K}_1$	1 [44]
MOS constant, $\hat{K}_2$	3 [44]
carrier frequency, f_c	2.5 GHz
path loss threshold, l_0	80 dB
noise power, σ^2	-95 dBm
transmission power of BS, P^{BS}	43 dBm [43]
transmission power of UAVs, P^{UAV}	40 dBm [45]
shadowing parameter of LOS links, η^{LoS}	1 dB [33]
shadowing parameter of NLOS links, η^{NLoS}	20 dB [33]

1) *MADDPG:* In this algorithm, the actor network and critic network are trained according to (23), (24), (26),

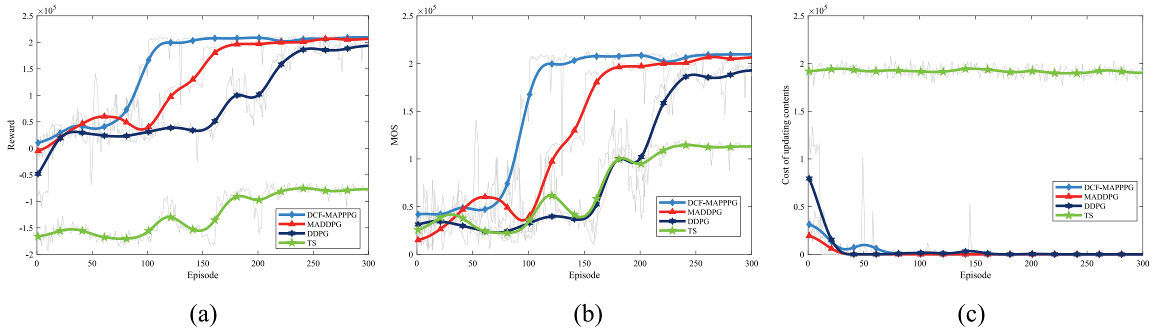


Fig. 6. Reward, MOS, and content updating cost under different algorithms. (a) Reward under different algorithms. (b) MOS under different algorithms. (c) Content updating cost under different algorithms.

TABLE III
HYPER-PARAMETERS OF ACTOR NETWORK

Parameters	Values
Number of hidden layers	3
Number of neurons in hidden layers	30,30,30
Activation function	ReLu,ReLu,Tanh
ATN updating cycle T^{a}	1
ACN updating cycle $\hat{T}^{\mathrm{a}}$	1
Forgetting factor τ^{a}	0.01
Learning rate λ^{a}	5×10^{-4}

TABLE IV
HYPER-PARAMETERS OF CRITIC NETWORK

Parameters	Values
Number of hidden layers	2
Number of neurons in hidden layers	60,30
Activation function	ReLu,ReLu
CTN updating cycle T^{c}	1
CCN updating cycle $\hat{T}^{\mathrm{c}}$	1
Forgetting factor τ^{c}	0.01
Learning rate λ^{c}	7×10^{-4}

and (29) without federated learning to develop the trajectory and caching strategy.

2) *DDPG:* According to the DDPG, each UAV independently learns the trajectory and caching strategy without collaboration.

3) *Trajectory Strategy Without Caching Decision Making (TS):* Based on the MADDPG solution, UAVs cache contents randomly and learn the optimal trajectory strategy.

In Fig. 6, we present the average reward, MOS and content updating cost under different algorithms. Compared with MADDPG, the proposed DCF-MADPPG accelerates the convergence rate with the aid of dynamic clustering and global model aggregation, which represents that the proposed solution could make each UAV learn the experiences of others effectively. Then, it can be seen that the rewards and MOS of DCF-MADPPG and MADDPG significantly outperform than those of DDPG. For achieving an optimal network overall performance, DCF-MADPPG and MADDPG fully exploit the swarm intelligence to make each UAV learn the best decision on the trajectory and cache autonomously. Moreover, it is evident that DCF-MADPPG, MADDPG, and DDPG have much better performance than TS, indicating that it is essential to jointly determine both the trajectory and caching in the UAV-assisted network. The reason is that the trajectory and caching decisions are closely related with each other. Different trajectories would face different content requests, leading to different caching decisions.

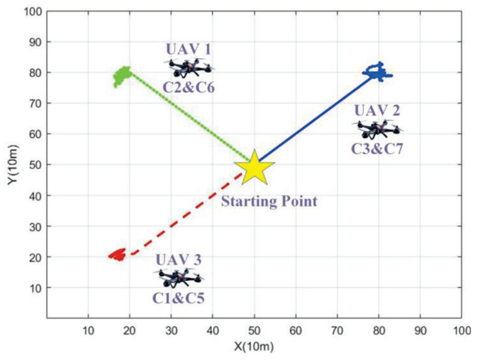


Fig. 7. Trajectory and caching decisions under the DCF-MADDPG solution.

In Fig. 7, we show trajectory and caching decisions under the DCF-MADPPG solution. We can observe that UAVs fly around different hot spots carrying on the cache popular contents. Taking the UAV 1 as an example, it flies around the hot spot 1, and caches the contents 2 and 6. Correspondingly, users in hot spot 1 prefer to download the contents 2 and 6, indicating UAVs could track dynamic traffic demand autonomously.

Next, we change the demand distributions of different contents in the 300th episode, regarding as a dynamic traffic environment, and evaluate the performance under different algorithms. The changed demand distribution of each content is shown in Fig. 8. In Fig. 9, we present the reward, MOS, and content updating cost under different algorithms in this dynamic traffic environment. From Fig. 9, it can be observed that the MADDPG and DCF-MADPPG make each UAV learn the optimal decision on the trajectory and cache adaptively in the new environment, and obtain a better reward than the DDPG and TS. Furthermore, compared with MADDPG, the

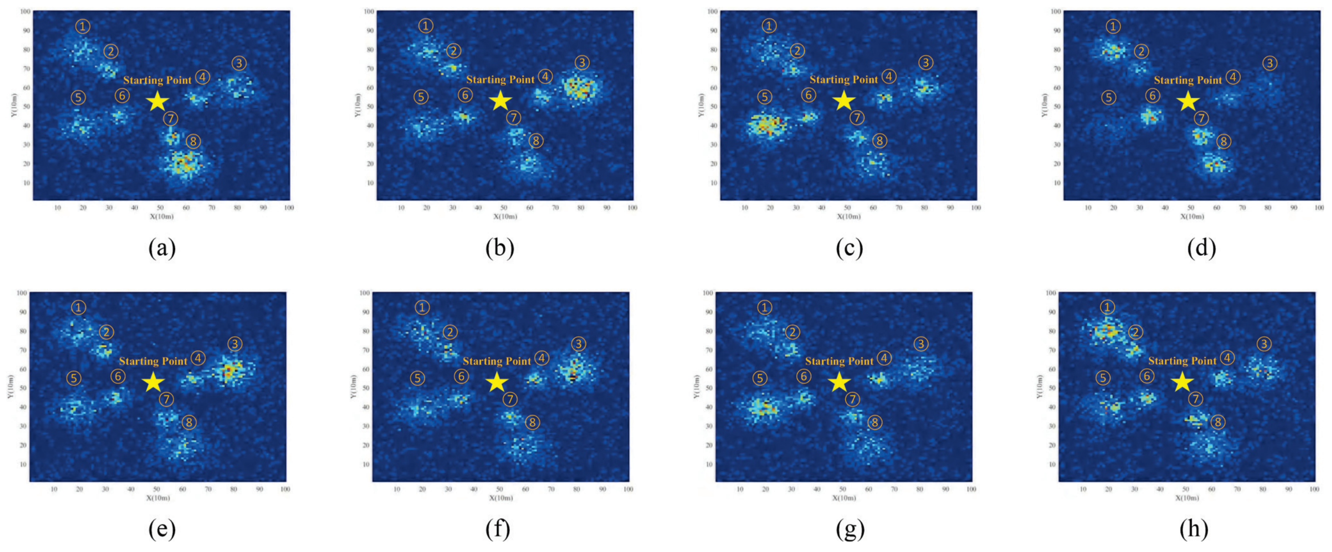


Fig. 8. Different demand distributions of different contents when the traffic environment changes. (a) Demand distribution on content 1. (b) Demand distribution on content 2. (c) Demand distribution on content 3. (d) Demand distribution on content 4. (e) Demand distribution on content 5. (f) Demand distribution on content 6. (g) Demand distribution on content 7. (h) Demand distribution on content 8.

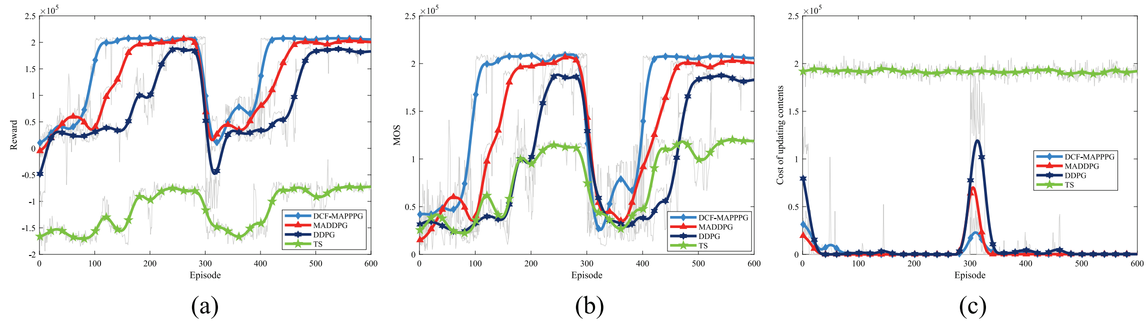


Fig. 9. Reward, MOS, and content updating cost under different algorithms when the traffic environment changes. (a) Reward under different algorithms. (b) MOS under different algorithms. (c) Content updating cost under different algorithms.

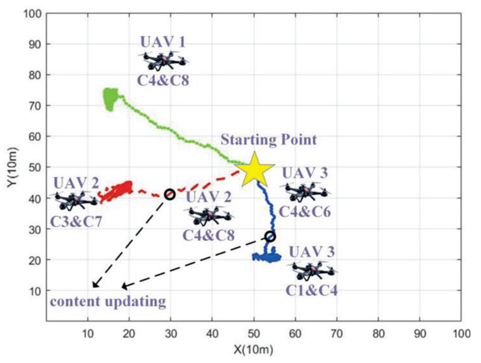


Fig. 10. Trajectory and caching decisions under the DCF-MADDPG solution when the traffic environment changes.

proposed DCF-MADPPG fully exploits the collaboration of UAVs to make each UAV learn the new environment feature faster. In summary, the proposed DCF-MADPPG algorithm can enable UAVs to track dynamic hot spots and cache popular contents adaptively, so that the users' QoE could be improved with low content updating cost.

In Fig. 10, we present trajectory and caching decisions under the DCF-MADDPG solution in this dynamic traffic environment. It can be observed that as the time goes on, UAVs can learn the new environment feature and adaptively adjust their trajectory and caching decisions. Specifically, taking the UAV 3 as an example, it ceases serving the original area and instead flies to the new hot spot adaptively. When UAV 3 flies over the hot spot 7, it caches the contents 4 and 6. From Fig. 8, we can see that users in hot spot 7 prefer to download the contents 4 and 6. Subsequently, UAV 3 leaves the hot spot 7 and heads toward the hot spot 8. During the flight, due to the cost on content updating, UAV 3 does not update cache units to serve scattered users. When UAV 3 arrives at the hot spot 8, it updates cache units to cache the contents 1 and 4. Correspondingly, users in hot spot 8 prefer to download the contents 1 and 4. The scheduling results represent that the UAVs could adapt to changing surroundings autonomously to make network overall performance optimal.

VI. Conclusion and Future Work

In this article, taking the long-term overall users' QoE as the objective, also with the consideration on the content updating cost, we proposed a QoE-aware COACH scheme for the UAV-assisted edge caching network, where the dynamics and uncertainty on the traffic environment caused by the varying user demand and user mobility were particularly considered. To address the content demand uncertainty, we developed an MADRL approach. It could make each UAV learn the best decision on the trajectory and caching autonomously based on its local observations to achieve an optimal network overall performance. Moreover, to fully exploit the swarm intelligence and make each UAV learn the experiences of others, we further developed a DCF-MADDPG algorithm. By performing parameter fusion in an UAV cluster, each UAV could improve the learning efficiency and achieve the optimal strategy faster. Simulation results have verified that the proposed DCF-MADDPG algorithm could make UAVs track dynamic traffic demand to improve users' QoE with low content updating cost. The key limitation of the proposed solution is the complexity on training the neural networks and the cost on the parameter fusion. Fortunately, the proposed COACH scheme can be implemented in a centralized training and decentralized executing manner, which would be practical for the UAV network. Specifically, the complex training process would be handled by the power central server, while each UAV only needs to download the updated parameters and perform the actions according to the output of the actor network based on its local observations. Furthermore, the parameter fusion could be implemented once a period of time, not very frequently. As for our future work, we will further jointly investigate the edge computing and caching, and develop more efficient multiagent decision making approaches.

References

[1] M. Shafi et al., "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 6, pp. 1201–1221, Jun. 2017.

[2] U. Cisco, "Cisco annual Internet report (2018–2023) white paper," Cisco Technol. Co., San Jose, CA, USA, White Paper, 2020.

[3] Y. Liu, Y. Mao, X. Shang, Z. Liu, and Y. Yang, "Distributed cooperative caching in unreliable edge environments," in *Proc. IEEE INFOCOM*, 2022, pp. 1049–1058.

[4] G. Qiao, S. Leng, S. Maharjan, Y. Zhang, and N. Ansari, "Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 7, no. 1, pp. 247–257, Jan. 2020.

[5] J. Xu, L. Chen, and P. Zhou, "Joint service caching and task offloading for mobile edge computing in dense networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2018, pp. 207–215.

[6] Y. Fu, Q. Yu, A. K. Y. Wong, Z. Shi, H. Wang, and T. Q. S. Quek, "Exploiting coding and recommendation to improve cache efficiency of reliability-aware wireless edge caching networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 11, pp. 7243–7256, Nov. 2021.

[7] Y. Guan, X. Zhang, and Z. Guo, "PrefCache: Edge cache admission with user preference learning for video content distribution," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 4, pp. 1618–1631, Apr. 2021.

[8] Y. Han, L. Ai, R. Wang, J. Wu, D. Liu, and H. Ren, "Cache placement optimization in mobile edge computing networks with unaware environment-an extended multi-armed bandit approach," *IEEE Trans. Wireless Commun.*, vol. 20, no. 12, pp. 8119–8133, Dec. 2021.

[9] O. Serhane, K. Yahyaoui, B. Nour, and H. Moungla, "A survey of ICN content naming and in-network caching in 5G and beyond networks," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4081–4104, Mar. 2021.

[10] F. Zhou, N. Wang, G. Luo, L. Fan, and W. Chen, "Edge caching in multi-UAV-enabled radio access networks: 3D modeling and spectral efficiency optimization," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 329–341, Apr. 2020.

[11] A. Masood, T.-V. Nguyen, T. P. Truong, and S. Cho, "Content caching in HAP-assisted multi-UAV networks using hierarchical federated learning," in *Proc. IEEE ICTC*, 2021, pp. 1160–1162.

[12] J. Ji, K. Zhu, and L. Cai, "Trajectory and communication design for cache-enabled UAVs in cellular networks: A deep reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 22, no. 10, pp. 6190–6204, Oct. 2023.

[13] X. Li, J. Liu, N. Zhao, and X. Wang, "UAV-assisted edge caching under uncertain demand: A data-driven distributionally robust joint strategy," *IEEE Trans. Commun.*, vol. 70, no. 5, pp. 3499–3511, May 2022.

[14] F. Cheng, G. Gui, N. Zhao, Y. Chen, J. Tang, and H. Sari, "UAV-relaying-assisted secure transmission with caching," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3140–3153, May 2019.

[15] J. Zhang, F. Liang, B. Li, Z. Yang, Y. Wu, and H. Zhu, "Placement optimization of caching UAV-assisted mobile relay maritime communication," *China Commun.*, vol. 17, no. 8, pp. 209–219, Aug. 2020.

[16] V. Sharma, I. You, D. N. K. Jayakody, D. G. Reina, and K.-K. R. Choo, "Neural-blockchain-based ultrareliable caching for edge-enabled UAV networks," *IEEE Trans. Ind. Informat.*, vol. 15, no. 10, pp. 5723–5736, Oct. 2019.

[17] X. Li, J. Liu, N. Zhao, and N. Yao, "A proactive joint strategy on trajectory and caching for UAV-assisted networks: A data-driven distributionally robust approach," in *Proc. IEEE ICCC*, 2021, pp. 752–757.

[18] X. Li, S. Cheng, H. Ding, M. Pan, and N. Zhao, "When UAVs meet cognitive radio: Offloading traffic under uncertain spectrum environment via deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 22, no. 2, pp. 824–838, Feb. 2023.

[19] J. Wang, C. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 73–82, Sep. 2017.

[20] G. Wu, Y. Miao, B. Alzahrani, A. Barnawi, A. Alhindi, and M. Chen, "Adaptive edge caching in UAV-assisted 5G network," in *Proc. IEEE GLOBECOM*, 2021, pp. 1–6.

[21] R. Zhou, X. Wu, H. Tan, and R. Zhang, "Two time-scale joint service caching and task offloading for UAV-assisted mobile edge computing," in *Proc. IEEE INFOCOM*, 2022, pp. 1189–1198.

[22] X. Wang and L. Duan, "Dynamic pricing and capacity allocation of UAV-provided mobile services," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, 2019, pp. 1855–1863.

[23] S. Gu, X. Sun, Z. Yang, T. Huang, W. Xiang, and K. Yu, "Energy-aware coded caching strategy design with resource optimization for satellite-UAV-vehicle-integrated networks," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5799–5811, Apr. 2022.

[24] L. Li et al., "Delay optimization in multi-UAV edge caching networks: A robust mean field game," *IEEE Trans. Veh. Technol.*, vol. 70, no. 1, pp. 808–819, Jan. 2021.

[25] T. Zhang, Z. Wang, Y. Liu, W. Xu, and A. Nallanathan, "Joint resource, deployment, and caching optimization for AR applications in dynamic UAV NOMA networks," *IEEE Trans. Wireless Commun.*, vol. 21, no. 5, pp. 3409–3422, May 2022.

[26] J. Luo, J. Song, F.-C. 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, pp. 5656–5660, May 2022.

[27] H. Peng and X. S. She, "DDPG-based resource management for MEC/UAV-assisted vehicular networks," in *Proc. 92nd IEEE VTC*, 2020, pp. 1–6.

[28] S. Anokye, D. Ayepah-Mensah, A. M. Seid, G. O. Boateng, and G. Sun, "Deep reinforcement learning-based mobility-aware UAV content caching and placement in mobile edge networks," *IEEE Syst. J.*, vol. 16, no. 1, pp. 275–286, Mar. 2022.

[29] Z. Wang, T. Zhang, Y. Liu, and W. Xu, "Deep reinforcement learning for caching placement and content delivery in UAV NOMA networks," in *Proc. IEEE WCSP*, 2020, pp. 406–411.

[30] S. Araf, A. S. Saha, S. H. Kazi, N. H. Tran, and M. G. R. Alam, "UAV assisted cooperative caching on network edge using multi-agent actor-critic reinforcement learning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 2, pp. 2322–2337, Feb. 2023.

[31] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[32] A. Al-Hourani, S. Kandeepan, and A. Jamalipour, "Modeling air-to-ground path loss for low altitude platforms in urban environments," in *Proc. IEEE GLOBECOM*, 2014, pp. 2898–2904.

[33] 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.

[34] M. Rugelj, U. Sedlar, M. Volk, J. Sterle, M. Hajdinjak, and A. Kos, "Novel cross-layer QoE-aware radio resource allocation algorithms in multiuser OFDMA systems," *IEEE Trans. Commun.*, vol. 62, no. 9, pp. 3196–3208, Sep. 2014.
[35] M. Li, X. Tao, N. Li, H. Wu, and J. Xu, "Secrecy energy efficiency Maximization in UAV-enabled wireless sensor networks without eavesdroppers CSI," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3346–3358, Mar. 2022.
[36] X. Dai, B. Duo, X. Yuan, and M. D. Renzo, "Energy-efficient UAV communications in the presence of wind: 3D modeling and trajectory design," *IEEE Trans. Wireless Commun.*, vol. 23, no. 3, pp. 1840–1854, Mar. 2024.
[37] 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, Sep. 2019.
[38] Y. Cai, Z. Wei, R. Li, D. W. K. Ng, and J. Yuan, "Joint trajectory and resource allocation design for energy-efficient secure UAV communication systems," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 4536–4553, Jul. 2020.
[39] X. Hou, J. Wang, C. Jiang, X. Zhang, Y. Ren, and M. Debbah, "UAV-enabled covert federated learning," *IEEE Trans. Wireless Commun.*, vol. 22, no. 10, pp. 6793–6809, Oct. 2023.
[40] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst*, 2017, pp. 6382–6393.
[41] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.
[42] S. Bubeck, "Convex optimization: Algorithms and complexity," *Found. Trends Mach. Learn.*, vol. 8, nos. 3–4, pp. 231–357, 2015.
[43] T. Zhang, Y. Wang, Y. Liu, W. Xu, and A. Nallanathan, "Cache-enabling UAV communications: Network deployment and resource allocation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7470–7483, Nov. 2020.
[44] Y. Wang, C. Feng, T. Zhang, Y. Liu, and A. Nallanathan, "QoE based network deployment and caching placement for cache-enabling UAV networks," in *Proc. IEEE ICC*, 2020, pp. 1–6.
[45] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for Optimized quality-of-experience," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1046–1061, May 2017.

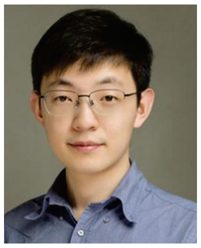
Xuanheng Li (Member, IEEE) received the Ph.D. degree in communication and information system from Dalian University of Technology, Dalian, China, in 2018.

From 2015 to 2017, he was with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA, as a Visiting Scholar. He is currently an Associate Professor with Dalian University of Technology. His current research interests include intelligent communications, wireless sensing, ISAC, and UAV-assisted networks.

Dr. Li was a recipient of the Best Paper Award at the IEEE Globecom 2015.

Jiahong Liu (Student Member, IEEE) received the B.S. and M.S. degree from Dalian University of Technology, Dalian, China, in 2020 and 2023, respectively, where he is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering.

His current research interests include UAV-assisted networks, mobile edge computing, and caching.



Xianhao Chen (Member, IEEE) received the B.Eng. degree in electronic information from Southwest Jiaotong University, Chengdu, China, in 2017, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2022.

He is currently an Assistant Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong. His research interests include wireless networking, edge intelligence, and machine learning.

Dr. Chen received the 2022 ECE Graduate Excellence Award for Research from the University of Florida. He serves as an Associate Editor for *ACM Computing Surveys*.

Jie Wang (Senior Member, IEEE) received the B.S. degree in electronic engineering from Dalian University of Technology, Dalian, China, in 2003, the M.S. degree in electronic engineering from Beihang University, Beijing, China, in 2006, and the Ph.D. degree in electronic engineering from Dalian University of Technology, in 2011.

He is currently a Professor with the School of Information Science and Technology, Dalian Maritime University, Dalian. He was a Visiting Researcher with University of Florida, Gainesville, FL, USA, from 2013 to 2014. His research interests include intelligent wireless sensing, machine learning, wireless localization and tracking, ISAC, and cognitive radio networks.

Prof. Wang is an Editor of *ACM Computing Surveys* and was an Associate Editor of IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

Miao Pan (Senior Member, IEEE) received the B.Sc. degree in electrical engineering from Dalian University of Technology, Dalian, China, in 2004, the M.A.Sc. degree in electrical and computer engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2007, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2012.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA. His research interests include wireless for AI, deep learning privacy, cybersecurity, and underwater communications and networking.

Dr. Pan work won the IEEE Technical Committee on Green Communications and Computing Best Conference Paper Awards in 2019 and the Best Paper Awards in ICC 2019, VTC 2018, Globecom 2017, and Globecom 2015. He was a recipient of the NSF CAREER Award in 2014. He is an Editor of IEEE OPEN JOURNAL OF VEHICULAR TECHNOLOGY and an Associate Editor of *ACM Computing Surveys*, IEEE INTERNET OF THINGS JOURNAL (Area 5: Artificial Intelligence for IoT), and IEEE JOURNAL OF OCEANIC ENGINEERING. He has also been serving as a Technical Organizing Committee for several conferences, such as the TPC Co-Chair for Mobiquitous 2019 and ACM WUWNet 2019. He is a member of AAAI and ACM.