Wireless Networks (2024) 30:1657–1670
https://doi.org/10.1007/s11276-023-03616-x

ORIGINAL PAPER

A collaborative offloading framework for multiple UAV considering service caching

Ruizhong Du[1] · Guangwen Yang[1] · Mingyue Li[1]

Accepted: 6 December 2023 / Published online: 28 December 2023

Abstract
Unmanned aerial vehicles (UAVs), with their flexible mobility, are widely used in wireless communication systems to provide high-quality auxiliary computing offloading services, especially when the original communication infrastructure is not available. Although many researches nowadays are studying the computing offloading problem of UAVs, service caching is also an important topic that should be considered in the UAV computing offloading process. The service caching serves as a database or library on which tasks pre-deployed on UAVs depend, and the computing offloading of tasks is limited by the deployment of the service caching. A single UAV has insufficient storage resources to deploy all types of service caching resources. To address the above issues, we propose a cooperative offloading strategy considering service caching based on Proximal Policy Optimization (PPO) named PPO-CO. First, we construct a system model and develop the collaborative offloading policy. The user's task can be accomplished either through collaborative offloading between UAVs or obtaining the required service caching resources through the remote service caching base station (RBS). Then, we format the policy as a complex nonlinear mixed integer programming optimization problem whose objective is to decrease the task execution delay and UAV energy consumption. Due to the challenge of solving the above problem, we solve it by PPO, a reinforcement learning (RL) algorithm. Finally, to verify the effectiveness of the PPO-CO, we compare it with the PPO-based offloading strategy without collaboration (PPO-NO), Asynchronous Advantage Actor-Critic (A3C) and Deep Q Network (DQN) based offloading strategy, and the random offloading policy in different number of user and UAV, task sizes, and service caching deployment ratios.

Keywords Offloading · Service caching · UAV · Reinforcement learning

1 Introduction

With the development of the Internet of Things (IoT) and mobile communication technologies such as 5 G and 6 G, users are demanding higher requirements for low latency, wide coverage, and mobility. As flexible mobile devices, unmanned aerial vehicles (UAVs) deployed with servers can provide ground terminals with a variety of functions such as relay communication, computational task offloading, and the distribution of service caching. As UAVs usually provide Line-of-Sight (LoS) channel links and can obtain better communication rates in relatively complex ground environments [1], UAVs will play a significant role in airborne wireless access in the future layout of a comprehensive communication for land, sea and air integration [2]. Many scenarios now require collaboration between multiple UAVs to accomplish UAV-assisted computing and communication. For example, if the original communication base station is damaged after the disaster, UAVs can provide emergency communication and computing services for rescue personnel and equipment [3].

UAVs are widely used in the computing offloading domain due to their mobility and computational capabilities [4]. However, as resource-constrained device, UAV is deficient in battery capacity, storage, and computing

✉ Guangwen Yang
ygwstudy@163.com

Ruizhong Du
durz@hbu.edu.cn

Mingyue Li
limingyue@hbu.edu.cn

[1] School of Cyber Security and Computer, Hebei University, Baoding 071000, Hebei, China

resources. Several scholars [5–8] have studied the offloading problem of UAVs to face the resource-constrained problem. The work [5] considers a single UAV aiding a single base station for wireless communication in a multi-access edge system and formulates it as a constrained optimization problem. In [6], a multi-hop task offloading scheme is proposed to achieve a more robust multi-UAV edge computing network. In the literature [7], the authors consider providing the computing services for ground vehicles on highways. Due to the mobility of vehicles, multi-UAVs are utilized to achieve adequate communication coverage. While the multi-UAV communication network in [7] may be a feasible solution to serve the ground devices, it fails to exploit the computational power of UAVs. Based on this, a UAV-supported offloading platform is considered in [8] that provides random movement and task arrival for multiple mobile ground users, and minimizes the average weighted energy consumption of all users subject to average UAV energy consumption and data queue stability constraints.

The above UAV-assisted offloading problem can be summarized as a non-convex optimization problem. The trajectory planning, offloading, and caching decisions of UAVs affect each other with specific coupling relationships, which are difficult to solve directly. Deep reinforcement learning (DRL) has been well exploited as a promising approach in UAV-assisted computing offloading. Several studies [9–11] have been practiced with DRL in UAV-assisted offloading scenarios. The optimization problem of the UAV-assisted framework proposed in [10] aims to optimize the geographical fairness among all UAVs, the fairness of each UAV load, and the overall energy consumption of the UAVs jointly. A low-complexity approach is introduced to optimize the offloading decision of UAVs for their trajectories by multi-agent reinforcement learning (MRL). A collaborative computational offloading and resource allocation scheme based on model-free DRL networks is proposed in [11], allowing continuous action space control. The scheme divides the UAV into multiple UAV clusters and runs an intelligent agent in each UAV cluster. It uses the Deep Deterministic Policy Gradient (DDPG), a DRL method, to train each intelligent agent.

Although computing offloading plays an important role in the system of UAV, service caching cannot be neglected. Unlike content caching, which is a static resource, service caching is also influenced by the computing capacity of the platform [12]. It affects task-offloading decisions. Service caching is the content on which the execution of specific computing tasks depends, which needs to be prearranged in the server. Several literature [2, 13–18] has investigated in terms of service caching and content caching.

In [13, 14], the authors provide optimization from the content placement perspective to improve the system's operational efficiency. The work [13] proposed a cooperative edge caching scheme to jointly optimize content placement and delivery and the literature [14] minimizes the average task execution time in multi-access edge computing by considering the heterogeneity of task requests, pre-storage of application data, and collaboration of base stations. In [15, 16], the authors consider the user's preference perspective to optimize the system efficiency. The work [15] takes as an entry point the events in a specific region of interest to the user and the latest data flow in that region. In contrast, the literature [16] starts with the caching problem under demand uncertainty and constructs an optimization strategy considering caching and offloading. In [17], the authors extend to the service caching problem in a multitasking scenario with joint task offloading and resource allocation.

Some literature [2, 18] has been studied in the context of UAV scenarios. In [2], a UAV-assisted wireless communication system using caching technology was designed to achieve multi-user data transmission and establish a joint optimization problem of UAV flight trajectory, cache location, and transmission power to minimize the task execution delay. In [18], the authors propose a method to apply mobile edge caching on UAVs in wireless communication systems and consider it from three user-UAV association criteria, namely the user received signal to noise ratio (SNR), user preferences and the delay.

It is worth noting that although there are many studies [5–8] on UAV-assisted offloading, they assume that the full range of types of service caching resources are deployed in a single UAV. Unfortunately, a UAV is not able to cover all types of service caching resources due to its resource-constrained nature. Since the service caching also depends on the computational power of the device, the computational offloading decision is affected by the service caching. [12] Some existing studies [13–16, 18] on service caching mainly focus on the service placement perspective to improve the operational efficiency of the system, and they do not consider the UAV offloading decision. Multi-UAVs can cover a wider range of service caching resources, and we can utilize the collaborative offloading of multi-UAVs to achieve efficient acquisition of caching resources. In addition, UAVs as resource-constrained devices, the energy consumption is directly related to the service duration. In order to ensure the quality of service (QoS) for users, the offloading decision should also consider the delay of task execution. [19] Inspired by the above two points, we propose the UAV cooperative offloading strategy based on Proximal Policy Optimization (PPO-CO) to improve the user 's QoS and decrease the energy consumption of UAVs.

The main contributions of this article are presented as follows.

- We construct an offloading model for dynamic collaboration of multiple UAVs considering service caching. It includes user task issuance, user-UAV association, UAV collaboration, and service caching. UAV trajectory planning is considered in the system. Also, to improve users' QoS and reduce UAVs' energy consumption, the above problem is represented as an optimization problem with the objectives of task execution delay and UAV energy consumption. It is formalized as a mixed integer programming problem.
- Considering the complexity and nonlinearity of the proposed optimization problem, we transform it into a Markov decision process (MDP). The state, action, and reward during the system's operation are defined using the properties of MDP. PPO-CO, a DRL algorithm, is proposed to solve the problem.
- To verify the effectiveness of PPO-CO, we compare it with PPO-based UAV no-cooperative offloading stratey (PPO-NO), random offloading strategy (Randomly), Asynchronous Advantage Actor-Critic (A3C) and Deep Q Network (DQN) based method. We compare them under the different scales of UAVs and the number of users, service caching deployment, and task scales.

The rest of the paper is structured as follows. Section 2 presents system models, which consists of offloading model, communication model, delay model, and energy model. Section 3 carries out the problem formalization. In Sect. 4, we transform the problem into an MDP and solve the problem by our proposed algorithm PPO-CO. Performance evaluation and simulation results are discussed in Sect. 5. Finally, we conclude the whole paper.

2 Model

2.1 System model

As shown in Fig. 1, we consider a system for UAV-assisted computational offloading, and our system consists of UAVs, mobile users on the ground, and remote service caching base station (RBS). The set of UAVs and users are represented by $\mathcal{N} = \{1, 2, \ldots, N\}$ and $\mathcal{M} = \{1, 2, \ldots, M\}$ respectively. Each UAV is equipped with a small server for offloading computational tasks. The UAVs also have a certain amount of storage space for storing service caching resources. The ground user randomly generates a computational task that depends on a specific service caching type. When the ground base station fails due to natural disasters, etc., the user's computation task needs to be transmitted to and executed by the UAV over a wireless channel. The maximum computing power of the UAV can be expressed as $F_{UAV} = \{f_1, f_2, ..., f_N\}$. The bandwidth of UAV can be denoted as $B = \{B_1, B_2, ..., B_N\}$.

There are S types of service caching, which is denoted as $s \in \mathcal{S}, 0 \leq s \leq S$. D_s denotes the size of the storage resource occupied by service caching s [20]. Not all UAVs have caching instances for each service due to data security and privacy concerns, load balancing, and limited storage at each base station [21]. The caching deployment in the UAV n is represented by

$$\begin{aligned} &\mathbf{SC}_n = [sc_n^0, ..., sc_n^s, ..., sc_n^S]^T, \\ &sc_n^s \in \{0, 1\}, \forall n \in \mathcal{N}, s \in \mathcal{S}. \end{aligned} \tag{1}$$

When the service caching s is deployed in the UAV n, it can be expressed as $sc_n^s = 1$, otherwise $sc_n^s = 0$. A mobile user generates a random task, which relies on a specific service caching. Only the UAV deployed with the corresponding service caching can execute the task. RBS is a server with powerful storage resources deployed with all service caching types. The location of RBS is deployed far away from the UAVs and users, which causes high transmission latency despite the good resource advantages of RBS. In order to accomplish the offloading of user tasks, we divided the offloading of user tasks into three parts: (1) UAV trajectory planning and task decision making, (2) task uploading, and (3) collaborative UAV offloading and task execution. UAVs that do not deploy the specific service caching required for task execution are not capable of executing the task, and need to transfer the task to other UAVs that have deployed the cache to execute it. The content of the required service caching can also be obtained through RBS.

2.2 Offloading model

An offloading process divides a period T into several small time slots. In each time slot $t \in \mathcal{T} = \{1, \ldots, T\}$, the user's position is assumed not to change. Assume that the length of one slot is τ_{total}. τ is used for the trajectory planning and offloading decision, and $\tau_{total} - \tau$ is used for the offloading and task computing.

Let Q_t denote the task request information for user m at time slot t, and it can be denoted as

$$Q_t = \langle F_t, S_s, D_t, m \rangle, \tag{2}$$

where F_t represents the number of CPU cycles required for computing the task, S_s denotes the type of service caching that the task needs, D_t denotes the amount of data the user needs to upload, and m is the user who generated the task.

To simplify the complexity of the model, we assume that the user's task is non-separable and the task can only

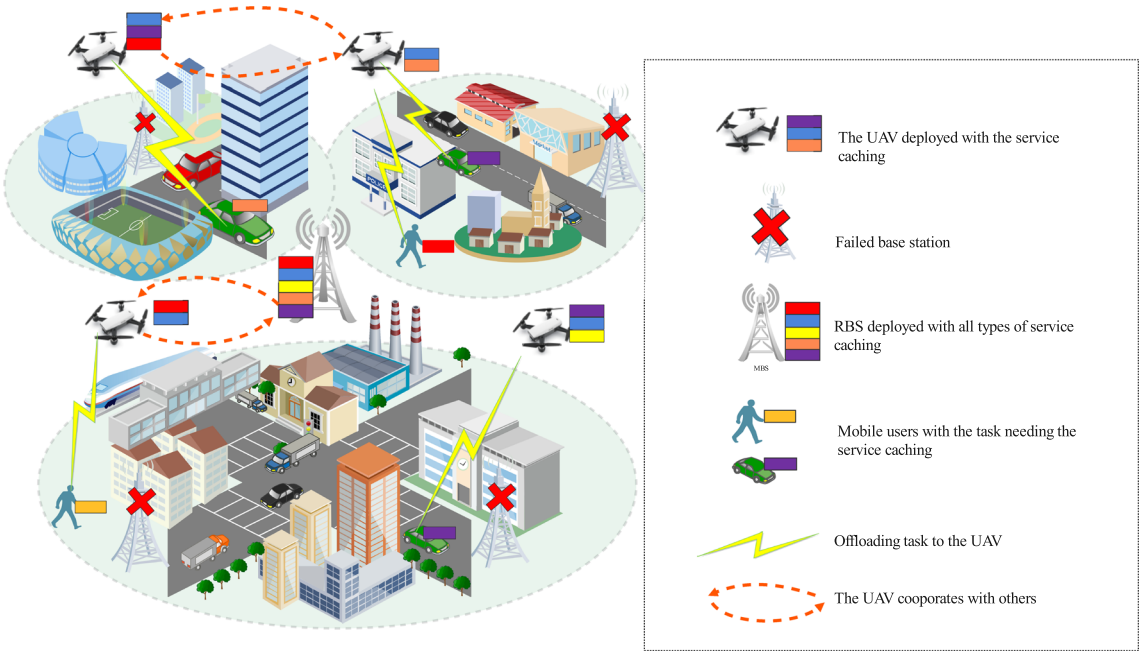

Fig. 1 System model figure

be computed by the UAV. Only one user will generate one task at the beginning of each time slot. The user m uploads the generated task to the associated UAV. The association variable between the UAV n and the user m can be expressed as

$$\boldsymbol{\alpha} = [a_1, ..., a_N]^T, a_n \in \{0, 1\}, 1 \leq n \leq N, \tag{3}$$

where n represents the serial number of the UAV. When $a_n = 1$, it means that the task is uploaded to the UAV n. The user's task can only be uploaded to one UAV, so the association variable should satisfy the constraint $\sum_{n=1}^{N} a_n = 1$. Users may not directly connect to the UAV which has deployed the service caching because the environment, changing channel states and distance affect the communication between UAVs and users. So after the task has been sent to the associated UAV, the task collaboration between the UAVs needs to be performed, and the collaboration variable between the UAVs is given by

$$\boldsymbol{\beta} = [b_1, ..., b_N]^T, b_n \in \{0, 1\}, 1 \leq n \leq N. \tag{4}$$

When $b_n = 1$, it means that the cooperative UAV is UAV n. Also, we assume that only one cooperative UAV can be selected to cooperate with the association UAV, so it satisfies $\sum_{n=1}^{N} b_n = 1$.

The offloading policy is determined based on $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and **SC**. To facilitate the representation of the offloading process, we assume that the user m generates the task witch relies on the service caching s, the associated UAV is n, and the cooperative UAV is n_1. When the variables satisfy the conditions of **P-1**, the **P-1** offloading decision is selected. When **P-1** is not satisfied, but the condition of **P-2** is satisfied, **P-2** is selected. Otherwise, **P-3** is executed. The detail of each offloading policy is as follows.

- **P-1** ($a_n = 1, sc_n^s = 1$): The user task is offloaded to the associated UAV n with the service caching. The UAV can execute the user's task directly because it has deployed the service caching resources required for task completion.
- **P-2** ($a_n = 1, sc_n^s = 0, b_{n_1} = 1, sc_{n_1}^s = 1$): The user offloads the task to the associated UAV n and UAV n relay it to the specified cooperative UAV n_1 through inter-UAV collaboration which executed the task indeed. The associated UAV does not have the service caching data required to perform the task. It needs to collaborate to forward the task information to the cooperative UAV and have it done.
- **P-3** ($a_n = 1, sc_n^s = 0, b_{n_1} = 1, sc_{n_1}^s = 0$): If neither the associated UAV nor the collaborative UAV has the service caching, the service caching is downloaded to the associated UAV via the RBS.

2.3 Communication model

Based on the modeling of the flight of UAVs in [21–23], and our proposed model, the following assumptions are made in this paper.

We assume that the flight altitude of the UAV is kept constant, the flight altitude of the UAV is H, and the position of the n-th UAV at the beginning of time slot t is

denoted as $\mathbf{u}_n[t] = [x_n, y_n, H]^t$. $\overline{\mathbf{X}}_m[t] = [\bar{x}_m, \bar{y}_m, 0]^t$ denotes the position of user m. The RBS's location is fixed and the location of it is denoted as $\mathbf{u}_{RBS}$. The distance between the user m and the UAV n can be given by

$$d_{n,m} = \left\|\mathbf{u}_n[t] - \overline{\mathbf{X}}_m[t]\right\|_2. \tag{5}$$

To prevent collision between UAVs, there exists a minimum distance d_{min} between UAVs, and the minimum distance between UAVs is constrained to be

$$d^2{}_{\min} \leq \left\|\mathbf{u}_n[t] - \mathbf{u}_j[t]\right\|^2, \forall n, j \in \mathcal{N}, j \neq n. \tag{6}$$

The flight speed of the UAV n during trajectory planning is $\mathbf{v}_n[t] = [v_x[t], v_y[t]]$, and the final position of the UAV n is denoted as

$$\mathbf{u}_n[t+1] = \mathbf{u}_n[t] + \mathbf{v}_n[t] \cdot \tau. \tag{7}$$

We suppose that the UAV has a maximum flight speed of V_{max}. At the beginning of each time slot, the UAV n relies on the velocity variable $\mathbf{v}_n$ to get a new position for trajectory planning.

Based on some previous studies [16, 24], we make assumptions about how UAVs and users communicate and do not specify a specific communication method. It is assumed that the connection between the ground user and the UAV is limited by factors such as the communication coverage of the UAV and the presence of obstacle occlusion during flight, which changes the communication state used for communication with the UAV. Based on the work [10], the indicator of the presence or absence of occlusion between the ground user m and the UAV n, is expressed as

$$\mathbf{BL}_m = [bl_m^1, ..., bl_m^n, ..., bl_m^N]^T, bl_m^n \in \{0, 1\}. \tag{8}$$

$bl_m^n = 1$ means there is an occlusion between user m and UAV n, otherwise there is no occlusion.

2.3.1 User to UAV

Based on the channel gain and the distance between the user and the UAV, as well as the signal-to-noise ratio between the user and the UAV [25], the data transmission rate for the data upload process between the user and the UAV can be further calculated as

$$r_{n,m,t} = B\log_2\left(1 + \frac{\rho P_n}{H^2 + d_{n,m}^2[t]}\right), \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{9}$$

where B represents the bandwidth, ρ is denoted as $\rho = \frac{g_0 G_0}{\sigma^2 + bl_m^n \cdot P_{NLOS}}$, $G_0 \approx 2.2846$ [26], g_0 denotes the power gain of the channel at a reference distance of 1 m, σ^2 denotes the noise power, P_n means the transmitted power of the user, and P_{NLOS} denotes the signal loss in the presence of occlusion.

2.3.2 UAV to UAV

The communication between UAVs is the same as the communication process between UAV and user, which also needs to occupy a channel resource for data transmission. The distance between UAV n_1 and UAV n_2 is expressed as

$$R_{n_1,n_2}[t] = \| \mathbf{u}_{n_1}[t] - \mathbf{u}_{n_2}[t] \|_2, \forall n_1, n_2 \in \mathcal{N}, n_1 \neq n_2. \tag{10}$$

Based on the previous modeling of the communication between the UAV and the user we can similarly obtain the data rate of the UAV n_1 and UAV n_2 transmission as

$$r_{n_1,n_2,t} = B\log_2\left(1 + \frac{\rho' P_n}{H^2 + R_{n_1,n_2}^2[t]}\right), \quad \forall n_1, n_2 \in \mathcal{N}, t \in \mathcal{T}, \tag{11}$$

where P_m is the magnitude of the data transmission power between the UAVs, and $\rho' = \frac{g_0 G_0}{\sigma^2}$. Assume that there is no occlusion between the UAVs.

2.3.3 UAV to RBS

The distance between the UAV n and RBS is expressed as

$$R_{n,RBS}[t] = \| \mathbf{u}_n[t] - \mathbf{u}_{RBS} \|_2, \forall n \in \mathcal{N}, \tag{12}$$

and the communication rate is expressed as

$$r_{n,RBS,t} = B\log_2\left(1 + \frac{\rho' P_n}{H^2 + R_{n,RBS}^2[t]}\right), \quad \forall n \in \mathcal{N}, t \in \mathcal{T}. \tag{13}$$

2.4 Delay model

For ease of expression, the following time delay is calculated assuming that user m generates a task in time slot t. The associated UAV and the collaborating UAV can be denoted as n and n_1, respectively. The actual UAV performing the task is $n' \in \{n, n_1\}$, as determined by the deployment or non-deployment of the service caching content in the UAV.

(1) Task transmission delay

The transmission delay used for the task generated by user m and transmitted to the associated UAV n is expressed as

$$T_{n,m,t}^{Tr} = \frac{D_t}{r_{n,m,t}}, \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}. \tag{14}$$

(2) Computing delay

The computing delay generated by UAV n' is expressed as

$$T_{n',t}^{Com} = \frac{F_t}{f_{n'}}, \forall n' \in \mathcal{N}, t \in \mathcal{T}. \tag{15}$$

Where f represents the computational speed of the UAV, i.e. the CPU main frequency of the UAV.

(3) Task migration delay

When the actual UAV n' performing the task is not the associated UAV n, we need to migrate the task to the UAV n_1. The task migration delay can be expressed as

$$T_{n,n_1,t}^{TrU} = \frac{D_t}{r_{n,n_1,t}}, \forall n, n_1 \in \mathcal{N}, t \in \mathcal{T}. \tag{16}$$

(4) Caching transmission delay

When neither the UAV n nor n_1 deploys the required service caching, the service caching needs to be downloaded from the RBS to complete the user's task execution. Caching transmission delay can be denoted as

$$T_{n,RBS,t}^{TrS} = \frac{D_s}{r_{n,RBS,t}}, \forall s \in \mathcal{S}, n \in \mathcal{N}, t \in \mathcal{T}. \tag{17}$$

Since the data length of the result after the calculation is relatively small, we assume that the delay in the result return is negligible. Consequently, the total task execution delay is expressed as

$$T_{n,m,t} = \begin{cases} T_{n,m,t}^{Tr} + T_{n',t}^{Com}, & \text{P-1} \\ T_{n,m,t}^{Tr} + T_{n',t}^{Com} + T_{n,n_1,t}^{TrU}, & \text{P-2} \\ T_{n,m,t}^{Tr} + T_{n,RBS,t}^{TrS} + T_{n',t}^{Com}, & \text{P-3.} \end{cases} \tag{18}$$

We assume that all tasks need to satisfy a maximum task completion time limit $T^{\max}$, i.e. $T_{n,m,t} \leq T^{\max}$.

2.5 Energy model

(1) Task transmission energy

The energy consumption incurred by the user in the process of establishing an association to the UAV and uploading it is expressed as

$$E_{n,m,t}^{Tr} = P_n T_{n,m,t}^{Tr}, \forall n \in \mathcal{N}, m \in \mathcal{M}, t \in \mathcal{T}, \tag{19}$$

where P_n is the user's transmission power.

(2) Computing energy

The computing energy generated by UAV n' is expressed as

$$E_{n',t}^{Com} = k_n \left(f_{n',t}\right)^{v_n} T_{n',t}^{Com}, \forall n' \in \mathcal{N}, t \in \mathcal{T}, \tag{20}$$

where k_n and v_n are two positive parameters.

(3) Caching transmission energy

The caching energy generated by UAV n and RBS is expressed as

$$E_{n,RBS,t}^{TrS} = P_n T_{n,RBS,t}^{TrS}, \forall n \in \mathcal{N}, t \in \mathcal{T}, \tag{21}$$

where P_n denotes the UAV's transmission power.

(4) Task migration energy

Task migration energy generated by UAV n and UAV n_1 can be expressed as

$$E_{n,n_1,t}^{TrU} = P_n T_{n,n_1,t}^{TrU}, \forall n, n_1 \in \mathcal{N}, t \in \mathcal{T}. \tag{22}$$

(5) UAV hovering and flying energy

E_0^{fly} is a fixed value and it denotes the hovering energy consumption of the UAV. The energy consumption of the UAV flighting and hovering can be calculated as

$$E_{n,t}^{Fly} = E_0^{fly} + \|\mathbf{v}_n[t]\|_2 \cdot \tau. \tag{23}$$

Let $E_{n,m,t}^{'} = E_{n,m,t}^{Tr} + E_{n',t}^{Com} + E_{n,t}^{Fly}$. The total energy consumption is expressed as

$$E_{n,m,t} = \begin{cases} E_{n,m,t}^{'} & \text{P-1} \\ E_{n,m,t}^{'} + E_{n,n_1,t}^{TrU}, & \text{P-2} \\ E_{n,m,t}^{'} + E_{n,RBS,t}^{TrS}, & \text{P-3.} \end{cases} \tag{24}$$

3 Problem formulation

The goal of the problem is to minimize the total task execution latency to improve the user' QoS. At the same time, we need to ensure that the energy consumption of UAVs is minimized because the energy of UAVs is precious. Given this, we set the objective of the problem as the sum of energy consumption of the system and the total task execution delay. The problem is denoted as P and expressed as follows.

$$P: \min_{\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{v}_n,n\in\mathcal{N}} \sum_{n=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T}(0.5 \cdot T_{n,m,t} + 0.5 \cdot E_{n,m,t}) \tag{25a}$$

s.t.

$$\|\mathbf{u}_n[t+1] - \mathbf{u}_n[t]\|^2 \leq (V_{\max}\tau)^2, \forall n \in \mathcal{N} \tag{25b}$$

$$[0,0,H] \leq \mathbf{u}_n[t] \leq [W,L,H], \forall n \in \mathcal{N} \tag{25c}$$

$$d^2{}_{\min} \leq \left\|\mathbf{u}_n[t] - \mathbf{u}_j[t]\right\|^2, \forall n, j \in \mathcal{N}, n \neq j \tag{25d}$$

$$\sum_{n=1}^{N} a_n = 1, \forall n \in \mathcal{N}, a_n \in \{0,1\} \tag{25e}$$

$$\sum_{n=1}^{N} b_n = 1, \forall n \in \mathcal{N}, b_n \in \{0,1\} \tag{25f}$$

$$T_{n,m,t} \leq T^{\max}. \tag{25g}$$

As can be seen, (25b)-(25d) are the UAV trajectory planning constraints, and (25e)-(25f) ensure that only one UAV can be selected as the association UAV and the cooperation UAV, and (25g) is the maximum tolerance delay constraint.

4 DRL-based solution

Since the P problem is a complex nonconvex problem, it is a mixed-integer programming problem due to having both integer and continuous variables. The problem is optimized into two subproblems, which are P_1: trajectory planning subproblem and P_2: task offloading subproblem.

The trajectory planning process and the task offloading process are coupled in both subproblems. The location of the UAV affects the user's selection of the UAV target for offloading and the selection of different UAV cooperation strategies. Similarly, the current task offloading decision affect the next step of UAV trajectory planning [2]. Therefore, these two subproblems cannot be solved independently, and the complexity of the solution is high, which is an NP-hard problem.

4.1 MDP

Our problem can be summarized as an MDP. MDP means that the current decision is only related to the current state, independent of the previous state, and the future action decision also needs to consider only the changed environment state after the previous action step, independent of the previous action taken. The UAV makes the action policy selection based on the current environment state at each time of path planning and associated user selection, independent of the previous UAV operation [27]. Usually an MDP contains three parts: state space, action space, and reward function. The final action policy of MDP is selected to maximize the accumulated reward.

4.2 State

In our system, the current state of the system before the start of each time slot is represented as

$$S_t = \{\mathbf{U}[t], \bar{\mathbf{X}}[t], \mathbf{Q}[t], \mathbf{BL}[t], \mathbf{SC}[t]\}, \tag{26}$$

where $\mathbf{U}[t] = [\mathbf{u}_1[t], ..., \mathbf{u}_n[t]]$ represents the current position of the UAV, and $\bar{\mathbf{X}}[t] = [\bar{\mathbf{X}}_1[t], ..., \bar{\mathbf{X}}_m[t]]$ denotes the current position of the user. $\mathbf{Q}[t]$, $\mathbf{BL}[t]$, $\mathbf{SC}[t]$ denote the mission information generated by the ground user, the current user-generated mission situation and whether the user is obscured, and the deployment of the service caching in the UAV, respectively.

4.3 Trajectory planning

Since the trajectory planning and computing offloading decisions of UAVs are coupled, their solution process is difficult. We consider the UAV trajectory planning and computing offloading decision centrally and use DRL-based method to solve it. In each time slot, τ is used for UAV trajectory planning and $\tau_{total} - \tau$ is used for UAV task offloading and execution. The position of the UAV at the beginning of time slot t is $\mathbf{U}[t-1]$. The trajectory of the UAV is controlled by the flight speed $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N]$. The position of the UAV at the beginning of the time slot and the flight speed of the UAV at time slot t are used to obtain the position of the UAV after trajectory planning. The new position of UAV n can be obtained by $\mathbf{u}_n[t] = \mathbf{u}_n[t-1] + \mathbf{v}_n[t-1] \cdot \tau$. Due to the use of DRL, new $\mathbf{V}$ can be continuously obtained based on the current state as time changes, thus obtaining a sequence of UAV trajectories for different time slots.

4.4 Aciton

The action of an intelligent agent making a decision is represented as

$$A = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_N]\}, \tag{27}$$

where V denotes the trajectory planning action of UAV, $\boldsymbol{\alpha}$ denotes the user issuing the task chooses an associated UAV to issue the task data, and $\boldsymbol{\beta}$ denotes the collaborative UAV selected during the completion of the task offloading. The actual task executor is determined based on the placement of the service caching in the associated UAV and the collaborative UAV.

4.5 Reward funciton

The whole system is to reduce the delay and energy consumption generated during the operation of the whole system by developing a reasonable offloading strategy, so the reward function must be set in such a way that it can satisfy the optimization goal of the system. The learning process of reinforcement learning is to obtain the maximized long-term reward, and the reward correlates the

calculation of the reward function with the optimization goal of the system, and the reward function is expressed as

$$R_t = -\sum_{n=1}^{N}\sum_{m=1}^{M}(0.5 \cdot T_{n,m,t} + 0.5 \cdot E_{n,m,t}). \tag{28}$$

4.6 DRL and PPO-CO

Reinforcement learning relies on the MDP nature of the event. We use DRL to complete the trajectory planning and offloading decision. The control of flight and offloading is a continuous action. In DRL algorithms, the commonly used algorithms are DQN, DDPG, A3C, Soft Actor-Critic (SAC) and other methods. Among them, DQN has only value network, which can only perform discrete action control and cannot be applied to continuous action space. DDPG and other Actor-Critic (AC) network algorithms have two kinds of networks, one is the action network and the other is the value network. The action network is used to output the action strategy, and the value network is used to evaluate the goodness of the action and output the reward value, and the goal of reinforcement learning is to optimize the reward. The reward can be calculated as

$$\bar{R}_\theta = \sum_v R(v)p_\theta(v) = \mathbb{E}_{v\sim p_\theta(v)}[R(v)], \tag{29}$$

where v represents a sampled trajectory with probability p. The probability of occurrence of a particular trajectory v can be calculated from θ. Next, the total reward of v is calculated. The total reward is weighted using the probability of occurrence of v and summed over all v, which is the expected value.

In order to make the reward as big as possible, so the gradient ascent can be used to maximize the desired reward. It can be expressed as

$$\nabla\bar{R}_\theta = \sum_v R(v)\nabla p_\theta(v). \tag{30}$$

We use the following equation for the update of the policy network:

$$\theta \leftarrow \theta + \eta\nabla\bar{R}_\theta, \tag{31}$$

where η denotes the learning rate. The full name of PPO is Proximal Policy Optimization, and PPO is a simultaneous policy algorithm that can be used for both discrete action problems and continuous action problems. Although the optimization objective of PPO involves importance sampling, it only uses the data of the previous round of policy θ'. PPO objective function adds the constraint of Kullback–Leibler (KL) scatter, the behavior policy θ' and the target policy θ are very close to each other, and the behavior policy and the target policy of PPO can be considered as the same policy, so PPO is a same-policy algorithm. Based on Trust Policy Optimization (TRPO), PPO uses a regular term as the trust region constraint, and the coefficients of this regular term are set according to whether the trust region constraint is observed or not, thus avoiding using the pairwise method. Use the same policy distribution ratio as TRPO, but instead of using the KL scatter constraint. It is used as part of the optimization objective [28]. The objective to be optimized by the PPO is denoted as

$$J^{\theta'}_{\mathrm{PPO}}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta,\theta') \tag{32}$$

$$J^{\theta'}(\theta) = \mathbb{E}_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t \mid s_t)}{p_{\theta'}(a_t \mid s_t)}A^{\theta'}(s_t,a_t)\right]. \tag{33}$$

$$J^{\theta^k}_{\mathrm{PPO}}(\theta) \approx \sum_{(st,a_t)} \min\left(\frac{p_\theta(a_t \mid s_t)}{p_{\theta^k}(a_t \mid s_t)}A^{\theta^k}(s_t,a_t),\right.$$
$$\left.\mathrm{clip}\left(\frac{p_\theta(a_t \mid s_t)}{p_{\theta^k}(a_t \mid s_t)}, 1-\varepsilon, 1+\varepsilon\right)A^{\theta^k}(s_t,a_t)\right). \tag{34}$$

Algorithm 1 shows the update process of the strategy in PPO algorithm, and Algorithm 2 shows the specific operation process of PPO-CO.

Traditional PPO algorithms do not consider the collaboration process between UAVs, we compare the PPO-CO method with the PPO-NO method, which utilizes a reinforcement learning PPO method to obtain the UAV offloading when there is no collaborative strategy between UAVs. The advantage of PPO-CO over traditional methods is that the collaboration process between UAVs is taken into account, and the service caching resources of UAVs can be more fully utilized, resulting in higher rewards.

Algorithm 1 PPO

Initialize policy parameters θ, value function parameters ϕ.
for $i \in \{1,2,...,\}$ **do**
  Run policy of θ, collect $\{a_t, s_t, r_t\}$, and estimate advantages $\hat{A}_t = \sum_{t'>t}\gamma^{t'-t}r_{t'} - V_\phi(s_t)$.
end for
for $j \in \{1,2,...,\}$ **do**
  $J^{\theta'}_{\mathrm{PPO}}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta,\theta')$
  update θ by gradient method: $\theta \leftarrow \theta + \eta\nabla\bar{R}_\theta$.
end for
for $k \in \{1,2,...,\}$ **do**
  $L_{BL}(\phi) = -\sum_{t=1}^{T}\left(\sum_{t'>t}\gamma^{t'-t}r_{t'} - V_\phi(s_t)\right)^2$
  update ϕ by gradient method of L_{BL}
end for

Algorithm 2 PPO-CO

Input: Training max slot T, max_episodes.
Output: Trained policy θ.
Initialize the policy θ.
for episode from 1 to max_episodes **do**
  Initialize the environment, $step = 0$.
  for slot t from 1 to T **do**
    Observe the current state s_t, and get action a_t by using old model θ'.
    Run trajectory planning of UAV based on the flight speed $\mathbf{v}_n$ from a_t
    Select one offloading policy from {**P-1**,**P-2**,**P-3**} by a_t.
    Upload and execute the task.
    Calculate the reward r_t by $\{s_t, a_t\}$ and get new state s_{t+1}.
    Store the $\{s_t, a_t, r_t\}$ into relay buffer.
    if step meets the conditions for updating **then**
      Using **Algorithm 1** to update θ.
    end if
    step=step+1
    if $step > max\_step$ or done **then**
      break
    end if
  end for
end for

5 Experimentation and performance verification

In this section, simulation experiments are set up to verify the effectiveness of PPO-CO. The structure of this section is as follows. Firstly, the experimental environment, the settings of the relevant parameters, and the setup of the comparison experiment are introduced. Then, we analyse the most available super parameter settings for the PPO-CO experiment. Finally, we compare PPO-CO with other algorithms in different scene settings, and verify the effectiveness of the proposed method.

5.1 Experiment setting

In order to verify the validity of the experiments, this paper refers to the environment setting parameters and communication parameters of some simulation experiments with similar UAV offloading scenarios. The experiment assumes the position movement of UAVs and users in a two-dimensional space of 100 m × 100 m, the flight altitude angle of UAVs is kept at 100 m [29], and the coordinate position of the service caching base station is denoted as $\mathbf{u}_{RBS}$ = [400 m, 400 m, 100 m] [30, 31]. The benchmark experiment was set up with three UAVs and five randomly distributed movable users at different locations on the ground, with a maximum movement speed of 1 m/s for the ground users and a maximum flight speed of 50 m/s for the UAVs [29]. The size of user task is 1−1.5 Mbits and the size of service caching is 1-9 Mbits. The number of service caching types is four and the percentage of caching deployment of UAVs is from 25% to 100%. The total time of system operation T=400 s, the time of one time slot is $\tau_{total} = 10$ s [32], and the time used for UAV flight trajectory planning is $\tau = 2$ s. The transmission bandwidth is set to 1 MHz [33], the noise power of the receiver is expressed as $\sigma^2 = -100$ dbm with no obstruction by the UAV, the power gain of the channel at a reference distance of 1 m is expressed as $g_0 = -50$ dB [6], the transmission power of the user is 0.1 W [32], and the transmission power between UAVs is P_m = 1 W [32]. The power of the UAV to download the service caching from RBS is $P^{trS} = 0.5$ W. The computing power of the UAV is $f_{UAV} = 1.5$ GHz [32]. k_n and v_n are 3 and 10^{-27} respectively.

For comparison, four baseline approaches are described as follows:

- **PPO-NO**: It based on the PPO algorithm for solving, but no task migration and collaborative execution between UAVs. When UAVs cannot meet the service caching requirements of users, they need to use RBS to provide caching services.
- **A3C-CO**: The A3C-CO method is based on the DRL algorithm A3C and combined with our proposed collaborative offloading strategy. A3C can be applied to the use of discrete and continuous actions. The Actor-Critic is put into multiple threads for synchronized training, which can effectively utilize the computing resources and improve the training utility.
- **DQN-CO**: The DQN-CO method is formulated based on the DRL algorithm DQN and our proposed collaborative offloading strategy. DQN can only be applied to discrete actions, and we discretize the action space to adapt to the requirements of DQN.
- **Randomly**: The location trajectory planning, offloading, and collaboration decisions of UAVs are randomized. user-generated tasks will be randomly assigned to an associated UAV or collaborative UAVs. The significance of the Randomly offloading strategy is to obtain the effect improvement of the PPO-CO method on the target outcome using the baseline method.

5.2 Experimental results and analysis

5.2.1 Optimal parameter setting

The convergence of PPO-CO in different learning cases can be seen in Fig. 2. The discount rate of 0.96 is able to reach convergence of the PPO-CO model at 4000 episodes when the learning rate is 0.001. The discount rate at 0.99 and 0.90 do not reach convergence in 6000 episodes. It can be found that both higher and lower reward discount rates do not result in better model discount rates. In this paper, a discount rate of 0.96 is used as the best discount rate.

The model convergence of the PPO-CO model is compared in Fig. 3 for three cases of learning rate 0.001, 0.01, and 0.1 when the discount rate is 0.96. At the learning rates of 0.01 and 0.1, the model does not obtain the optimal solution due to the higher learning rates and the larger parameter adjustment steps of the model. The best convergence and reward results can be obtained at the learning rate of 0.001, and 0.001 is taken as the most hyperparameterized value of the learning rate.

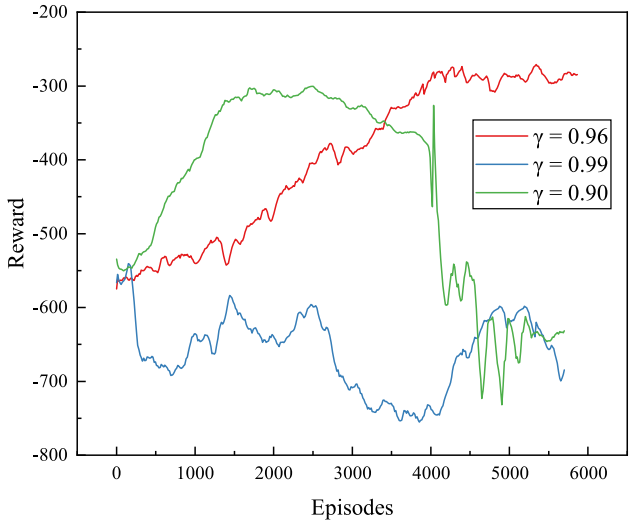

Fig. 2 Convergence of PPO-CO for different discount

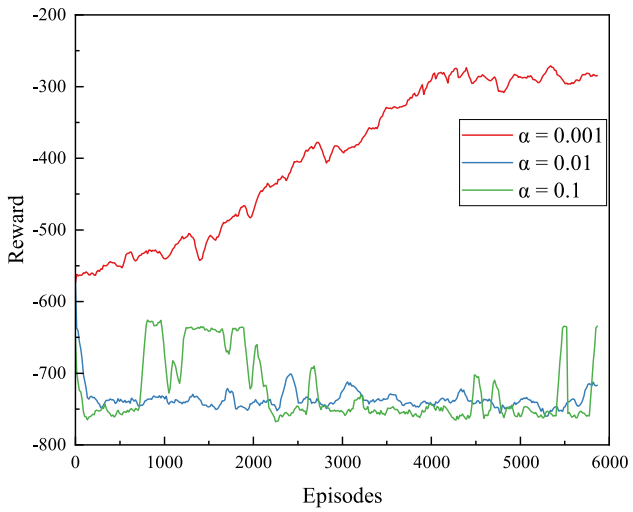

Fig. 3 Convergence of PPO-CO at different learning rates

5.2.2 Performance comparison

In Fig. 4, we compare the convergence performance of PPO-CO, PPO-NO, A3C-CO, DQN-CO and Randomly. We evaluate the average total reward of each episodes as the reward is directly related to the UAV energy consumption and the execution delay of the user's task correlated. Higher reward means lower energy consumption and latency. All four RL algorithms reach a state of convergence in Fig. 4. The PPO-CO algorithm ends up with the highest reward value. Since PPO-NO does not include the UAV collaborative offloading process, the UAV has to rely on the RBS when it cannot satisfy the task execution conditions. This leads to a higher time cost, which ultimately reduces the rewards. A3C-CO, although it converges faster, is lower than PPO-CO in terms of rewards. DQN-CO, because it can only deal with discrete actions, has discretized actions which lead to the inability of the method to find the actions with higher reward values. After that, comparisons will be made from different task sizes, caching deployment ratios, user and UAV sizes to observe the effectiveness and stability of the experimental results under different situations.

The rewards of the five methods for different total task sizes in an episode are shown in Fig. 5. In the case of

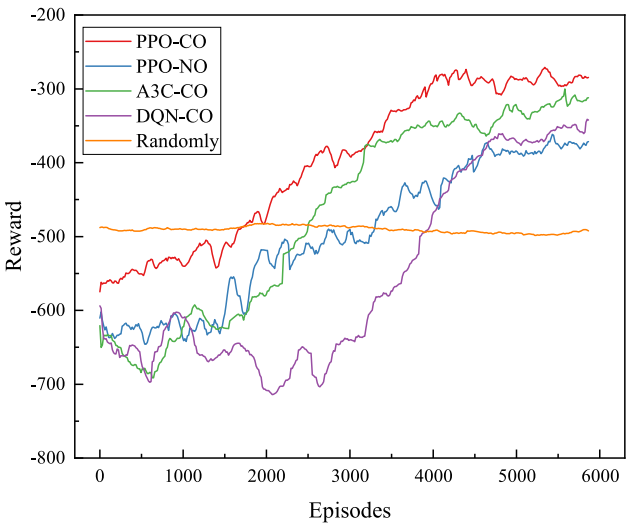

Fig. 4 Convergence under different methods

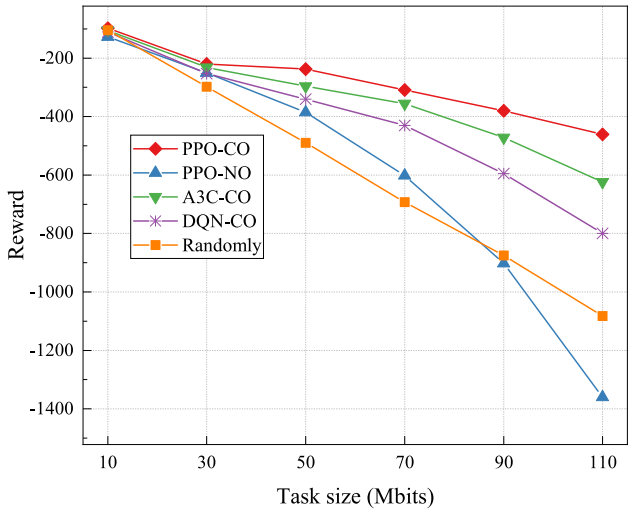

Fig. 5 Comparison of different task size reward

smaller task sizes, the difference in rewards between the different methods is small. In the case of larger task sizes, the reward value of PPO-NO is even lower than that of the Randomly method. The reason for this situation is that the difference in the size of the rewards produced by different offloading methods is not significant in the case of small task volumes due to the relatively small latency and energy consumption incurred by the transmission and execution of the tasks. In the case of a large number of tasks, because Randomly, despite being a randomized decision, takes into account collaboration between UAVs, the PPO-NO method without offloading leads to higher latency and energy consumption in the face of a large number of tasks. The PPO-CO consistently outperforms the A3C-CO and DQN-CO methods in the offloading approach with collaboration. The results demonstrate that our proposed method is capable of lower energy consumption and delay with different task sizes.

In figure 6, we consider the effect of different service caching sizes on the reward. When the size of the service caching is small, the difference in rewards between the different offloading schemes is small. The main reason is that a smaller service caching size will produce smaller transmission latency and energy consumption, when the impact on the overall reward is smaller. Collaboration between UAVs will play a crucial role when the size of the service cache increases. From the changes in the lines of PPO-NO, it can be seen that a larger service caching data size in the no-collaboration scenario will greatly increase the energy consumption and latency of the system. The results prove that PPO-CO is able to obtain the highest reward at different caching sizes.

Figure 7 shows the impact of the change in the proportion of service caching types deployed in the UAV. The proportion of caching deployments is the same for the collaborative and non-collaborative approach at 100%, and the impact of the proportion of caching deployments on the reward is positively correlated. This is because when the caching deployment of UAVs reaches 100%, the problem will degrade to a scenario where service caching is not considered. At this point the collaboration policy will not work. Collaboration between UAVs makes sense when the service caching in the UAV is not able to cover all of the caching types, and collaboration reduces the energy consumption and latency required for task completion. Collaboration on resource-constrained devices like UAVs facilitates the expansion of the problem of insufficient resources for individual UAVs and contributes to the operational efficiency of the whole system.

In Figures 8 and 9 we summarize the rewards for different numbers of UAVs and user scenarios. Figure 8

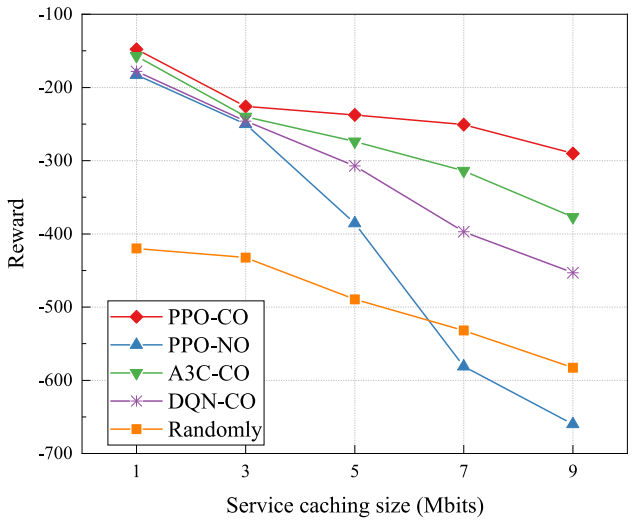

Fig. 6 Comparison of different service caching size

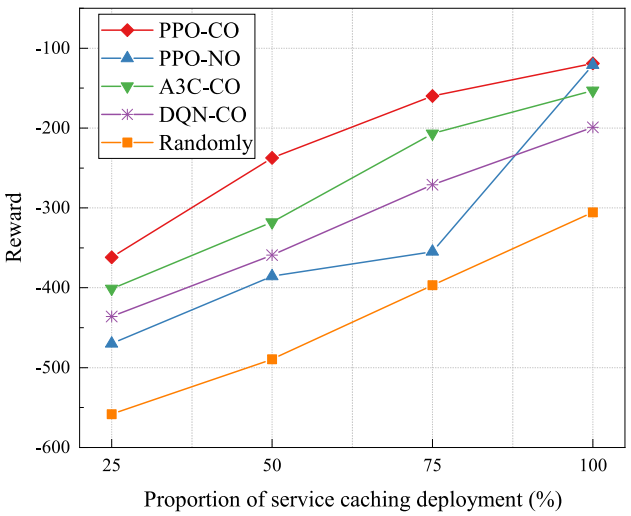

Fig. 7 Comparison of different caching deployment ratios

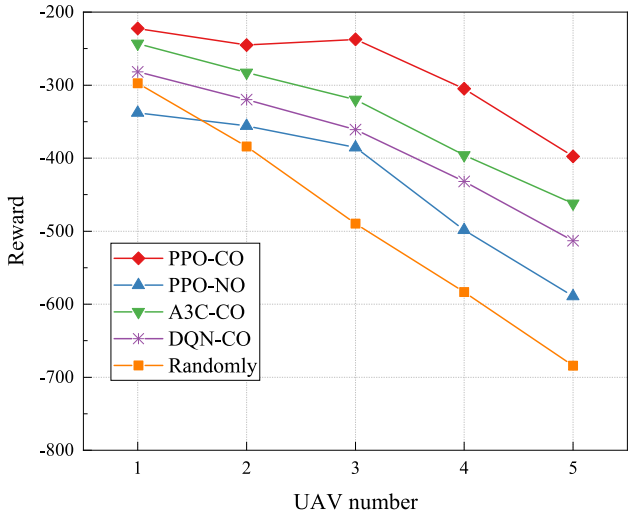

Fig. 8 Comparison of different number of UAV

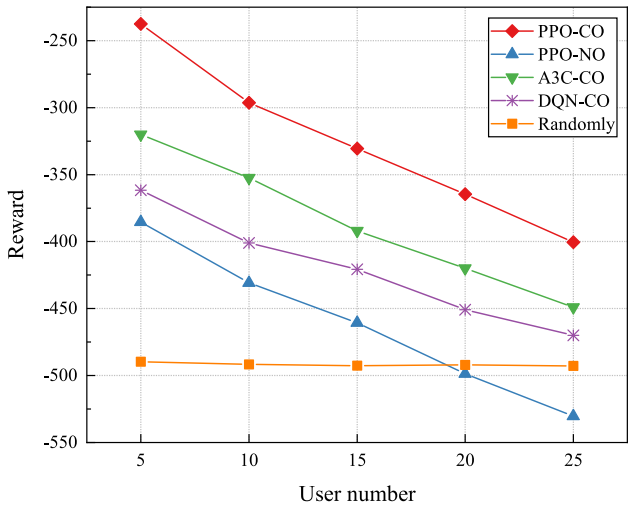

Fig. 9 Comparison of different number of user

shows the overall system reward for different number of UAVs. At a low number of UAVs, the system as a whole has a high reward due to the low energy consumption of individual UAVs. The gap between PPO-CO and PPO-NO methods gradually shows the advantage of collaboration as the number of UAVs increases due to the smaller space for collaboration when the number of UAVs is small. Overall the benefits from collaboration at different UAV sizes are higher than those from the non-collaborative approach and the random collaboration strategy.

Fig. 9 shows the reward values for different numbers of ground users. For different user sizes, our proposed PPO-CO method is able to obtain the highest reward values compared to other methods. This shows that our proposed offloading method is able to provide computational offloading services with low latency and low energy consumption for a certain user size. The rewards value is decreasing as the number of users increases. This is because when the number of users increases, more users will increase the time and energy cost due to the limitations of the computational and communication capabilities of the system setup. When the number of users is large, the PPO-NO method that does not consider the inter-UAV collaborative offloading strategy will get lower than the Randomly method that considers randomized collaboration.

6 Conclusion

In this paper we propose a collaborative strategy for UAV-assisted computational offloading considering service caching, which aims to reduce the energy consumption of UAVs and the latency of user task execution. We consider the dependence of user tasks on specific service caching during computing offloading. We construct a system model containing UAVs, ground-mobile users, and RBS, and jointly consider UAV-user association, collaborative offloading strategy, service caching provisioning, and UAV trajectory planning, and model them as a optimization problem. Then we propose a PPO-based offloading strategy to ensure timely execution of user tasks and reduce the energy consumption of the UAVs to obtain longer service duration. Compared to PPO-NO, our collaborative strategy can more fully utilize the limited service caching resources and obtain lower energy consumption and latency, proving the superiority of the proposed collaborative offloading approach. Compared to A3C-CO and DQN-CO, two RL approaches that consider collaboration, our proposed approach is able to obtain lower system cost. When the caching deployed in UAVs have low coverage, our proposed collaborative offloading approach is able to obtain the maximum benefit improvement compared to the no-collaboration strategy. In addition, our proposed method can be adapted to different scenarios with different task data volumes, service caching sizes, number of UAVs and users.

Funding The authors did not receive support from any organization for the submitted work.

Data availability The raw/processed data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval Not applicable.

Consent for publication All authors approved the final manuscript and the submission to this journal.

References

1. Chai, S., & Lau, V. K. N. (2021). Multi-UAV trajectory and power optimization for cached UAV wireless networks with energy and content recharging-demand driven deep learning approach. *IEEE Journal on Selected Areas in Communications, 39*(10), 3208–3224. https://doi.org/10.1109/jsac.2021.3088694
2. Lan, T., Qin, D., & Sun, G. (2021). Joint optimization on trajectory, cache placement, and transmission power for minimum mission time in uav-aided wireless networks. *ISPRS International Journal of Geo-Information, 10*(7), 426.
3. Chen, W.W., Su, Z., Xu, Q.C., Luan, T.H., Li, R.D. (2020). VFC-based cooperative UAV computation task offloading for post-disaster rescue. IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, pp. 228–236
4. Nguyen, M. D., Le Bao, L., & Girard, A. (2021). *Trajectory Control and Resource Allocation for UAV-Based Networks with Wireless Backhauls*. https://doi.org/10.1109/icc42927.2021.9500502
5. Zhang, K., Gui, X., Ren, D., & Li, D. (2021). Energy-latency tradeoff for computation offloading in UAV-assisted multiaccess edge computing system. *IEEE Internet of Things Journal, 8*(8), 6709–6719. https://doi.org/10.1109/jiot.2020.2999063
6. He, X., Jin, R., & Dai, H. (2022). Multi-hop task offloading with on-the-fly computation for multi-UAV remote edge computing. *IEEE Transactions on Communications, 70*(2), 1332–1344. https://doi.org/10.1109/tcomm.2021.3129902
7. Samir, M., Ebrahimi, D., Assi, C., Sharafeddine, S., & Ghrayeb, A. (2021). Leveraging UAVS for coverage in cell-free vehicular networks: A deep reinforcement learning approach. *IEEE Transactions on Mobile Computing, 20*(9), 2835–2847. https://doi.org/10.1109/tmc.2020.2991326
8. Yang, Z., Bi, S., & Zhang, Y.-J.A. (2021). *Dynamic Trajectory and Offloading Control of UAV-enabled MEC under User Mobility*. https://doi.org/10.1109/ICCWorkshops50388.2021.9473504
9. Anokye, S., Ayepah-Mensah, D., Seid, A. M., Boateng, G. O., & Sun, G. (2022). Deep reinforcement learning-based mobility-aware UAV content caching and placement in mobile edge networks. *IEEE Systems Journal, 16*(1), 275–286. https://doi.org/10.1109/jsyst.2021.3082837
10. Wang, L., Wang, K., Pan, C., Xu, W., Aslam, N., & Hanzo, L. (2021). Multi-agent deep reinforcement learning-based trajectory planning for multi-UAV assisted mobile edge computing. *IEEE Transactions on Cognitive Communications and Networking, 7*(1), 73–84. https://doi.org/10.1109/tccn.2020.3027695
11. Seid, A. M., Boateng, G. O., Anokye, S., Kwantwi, T., Sun, G., & Liu, G. (2021). Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: A deep reinforcement learning approach. *IEEE Internet of Things Journal, 8*(15), 12203–12218. https://doi.org/10.1109/jiot.2021.3063188
12. Ma, X., Zhou, A., Zhang, S., Wang, S.G. (2020). Cooperative service caching and workload scheduling in mobile edge computing. IEEE Infocom 2020 - IEEE Conference on Computer Communications, pp. 2076–2085.
13. Qiao, G., Leng, S., Maharjan, S., Zhang, Y., & Ansari, N. (2020). Deep reinforcement learning for cooperative content caching in vehicular edge computing and networks. *IEEE Internet of Things Journal, 7*(1), 247–257. https://doi.org/10.1109/jiot.2019.2945640
14. Zhong, S., Guo, S., Yu, H., & Wang, Q. (2021). Cooperative service caching and computation offloading in multi-access edge computing. *Computer Networks*. https://doi.org/10.1016/j.comnet.2021.107916
15. Al-Hilo, A., Samir, M., Assi, C., Sharafeddine, S., & Ebrahimi, D. (2021). UAV-assisted content delivery in intelligent transportation systems-joint trajectory planning and cache management. *IEEE Transactions on Intelligent Transportation Systems, 22*(8), 5155–5167. https://doi.org/10.1109/tits.2020.3020220
16. Li, X., Liu, J., Zhao, N., & Wang, X. (2022). UAV-assisted edge caching under uncertain demand: A data-driven distributionally robust joint strategy. *IEEE Transactions on Communications, 70*(5), 3499–3511. https://doi.org/10.1109/TCOMM.2022.3161021
17. Zhang, G., Zhang, S., Zhang, W., Shen, Z., & Wang, L. (2021). Joint service caching, computation offloading and resource allocation in mobile edge computing systems. *IEEE Transactions on Wireless Communications, 20*(8), 5288–5300. https://doi.org/10.1109/twc.2021.3066650
18. Zhang, M., Mohammed, E.-H., & Ng, S. X. (2021). Intelligent caching in UAV-aided networks. *IEEE Transactions on Vehicular Technology, 71*(1), 739–752.
19. Zhan, C., Hu, H., Liu, Z., Wang, Z., & Mao, S. (2021). Multi-UAV-enabled mobile-edge computing for time-constrained IoT applications. *IEEE Internet of Things Journal, 8*(20), 15553–15567. https://doi.org/10.1109/jiot.2021.3073208
20. Wu, R., Tang, G., Chen, T., Guo, D., Luo, L., & Kang, W. (2022). A profit-aware coalition game for cooperative content caching at the network edge. *IEEE Internet of Things Journal, 9*(2), 1361–1373. https://doi.org/10.1109/jiot.2021.3087719
21. Xu, Z., Zhou, L., Dai, H., Liang, W., Zhou, W., Zhou, P., Xu, W., & Wu, G. (2022). Energy-aware collaborative service caching in a 5G-enabled MEC with uncertain payoffs. *IEEE Transactions on Communications, 70*(2), 1058–1071. https://doi.org/10.1109/tcomm.2021.3125034
22. Wang, J., Na, Z., & Liu, X. (2021). Collaborative design of multi-UAV trajectory and resource scheduling for 6g-enabled internet of things. *IEEE Internet of Things Journal, 8*(20), 15096–15106. https://doi.org/10.1109/jiot.2020.3031622
23. Zheng, G., Xu, C., Long, H., & Sheng, Y. (2021). Service caching based task offloading and resource allocation in multi-UAV assisted MEC. *Networks*. https://doi.org/10.1109/iccc52777.2021.9580248
24. Liao, Z., Ma, Y., Huang, J., Wang, J., & Wang, J. (2021). Hotspot: A UAV-assisted dynamic mobility-aware offloading for mobile-edge computing in 3D space. *IEEE Internet of Things Journal, 8*(13), 10940–10952. https://doi.org/10.1109/jiot.2021.3051214
25. Ji, J., Zhu, K., Yi, C., & Niyato, D. (2021). Energy consumption minimization in UAV-assisted mobile-edge computing systems: Joint resource allocation and trajectory design. *IEEE Internet of Things Journal, 8*(10), 8570–8584. https://doi.org/10.1109/jiot.2020.3046788
26. Xiong, Z., Zhang, Y., Lim, W. Y. B., Kang, J., Niyato, D., Leung, C., & Miao, C. (2021). UAV-assisted wireless energy and data transfer with deep reinforcement learning. *IEEE Transactions on Cognitive Communications and Networking, 7*(1), 85–99. https://doi.org/10.1109/tccn.2020.3027696
27. Yao, J., & Ansari, N. (2021). Caching in dynamic IoT networks by deep reinforcement learning. *IEEE Internet of Things Journal, 8*(5), 3268–3275. https://doi.org/10.1109/jiot.2020.3004394
28. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347

29. Yu, Y., Bu, X., Yang, K., Yang, H., Gao, X., & Han, Z. (2021). UAV-aided low latency multi-access edge computing. *IEEE Transactions on Vehicular Technology, 70*(5), 4955–4967. https://doi.org/10.1109/tvt.2021.3072065
30. Zhang, S., Liu, W., & Ansari, N. (2022). On tethered UAV-assisted heterogeneous network. *IEEE Transactions on Vehicular Technology, 71*(1), 975–983. https://doi.org/10.1109/TVT.2021.3130130
31. Do-Duy, T., Nguyen, L. D., Duong, T. Q., Khosravirad, S. R., & Claussen, H. (2021). Joint optimisation of real-time deployment and resource allocation for UAV-aided disaster emergency communications. *IEEE Journal on Selected Areas in Communications, 39*(11), 3411–3424. https://doi.org/10.1109/JSAC.2021.3088662
32. Wang, Y., Fang, W., Ding, Y., & Xiong, N. (2021). Computation offloading optimization for UAV-assisted mobile edge computing: a deep deterministic policy gradient approach. *Wireless Networks, 27*(4), 2991–3006. https://doi.org/10.1007/s11276-021-02632-z
33. Wang, D., Liu, Q., Tian, J., Zhi, Y., Qiao, J., Bian, J. (2021). Deep reinforcement learning for caching in D2D-enabled UAV-relaying networks. 10.1109/iccc52777.2021.9580299

Ruizhong Du received the PhD degree in information security from School of Computer Science, Wuhan University, China in 2012. Since 1997, he has been working at the School of Cyberspace Security and Computer, Hebei University China. He is currently the associate dean, a doctoral supervisor, and a professor at the School of Cyberspace Security and Computer, Hebei University. He is the secretary-general of the Hebei Cyberspace Security Society and the executive director of the Hebei Computer Society His research interests include network security, edge computing and trusted computing.

Guangwen Yang was born in 1997. He received the B.S. degree from Nanjing Tech University, Nanjing, China, in 2020. Currently, he is studying for a M.S. degree in School of Cyber Security and Computer at Hebei University, Baoding, China. His main research interests are edge computing resource management and edge computing security.

Mingyue Li was born in 1993. She received the MEng degree from Hebei University, China in 2019, and the PhD degree from Nankai University, China in 2022. In 2022, she joined Hebei University, China, where she is currently a lecturer at School of Cyber Security and Computer. Her research interests mainly include network security and searchable encryption technology.