drones

MDPI

Article

UAV-Centric Privacy-Preserving Computation Offloading in Multi-UAV Mobile Edge Computing

Chao Gao [1,2], Dawei Wei [1,*], Keying Li [3] and Wenjin Liu [3]

[1] School of Cyber Engineering, Xidian University, Xi'an 710071, China; 24151213668@stu.xidian.edu.cn
[2] Key Laboratory of Cyberspace Security, Zhengzhou 450001, China
[3] School of Telecommunications Engineering, Xidian University, Xi'an 710071, China; 23009190044@stu.xidian.edu.cn (K.L.); 23009190050@stu.xidian.edu.cn (W.L.)
* Correspondence: weidawei@xidian.edu.cn

Abstract

Unmanned aerial vehicles (UAVs) offer high mobility, cost-effectiveness and flexible deployment, but their limited computing and battery resources constrain their development. Mobile edge computing (MEC) can alleviate these constraints by computation offloading. Although reinforcement learning (RL) has recently been applied to optimize offloading strategies, using raw UAV data poses a risk of privacy leakage. To address this issue, we design a privacy-preserving RL-based offloading approach that applies local differential privacy (LDP) to perturb decision trajectories. We theoretically derive the $\mathcal{O}(\sqrt{M}/\epsilon)$ regret bound and achieve (ϵ, δ)-LDP for the perturbation mechanism. Finally, we evaluate the efficiency of the proposed approach through experiments.

Keywords: multi-UAV; mobile edge computing; computation offloading; reinforcement learning; local differential privacy

Academic Editor: Hiroyuki Tomiyama

Received: 26 August 2025
Revised: 8 October 2025
Accepted: 10 October 2025
Published: 12 October 2025

Citation: Gao, C.; Wei, D.; Li, K.; Liu, W. UAV-Centric Privacy-Preserving Computation Offloading in Multi-UAV Mobile Edge Computing. *Drones* **2025**, *9*, 701. https://doi.org/10.3390/drones9100701

1. Introduction

With the rapid development of UAV technology, UAVs can quickly reach locations inaccessible by other means of transportation due to their high mobility and flexibility. In areas with insufficient network coverage, UAVs can quickly serve as mobile communication relay stations. Consequently, UAV systems have been widely used for traffic monitoring, forest rescue, precision agriculture, disaster emergency response, and aerial remote sensing. However, their performance in complex scenarios is severely restricted by their limited onboard computing resources, insufficient processing power, and battery capacity, as well as communication latency and stability issues faced in high-demand real-time tasks.

In recent years, Internet of Things (IoT) technology has developed rapidly. The IoT connects a large number of devices together to collect and process device data [1]. The numerous IoT devices require a large amount of storage and computing resources [2]. Traditional cloud computing technology uses remote cloud servers as storage and computing servers for IoT devices. In recent years, computing-intensive tasks have emerged due to the rapid development of mobile terminals and IoT devices. However, traditional cloud computing technology involves transmitting these tasks, which can cause traffic congestion and time delays. In response to the above problems, mobile edge computing (MEC) has emerged [3].

By offloading computation-intensive tasks from IoT devices to nearby MEC servers, MEC technology leverages proximate storage and computational resources. This approach

effectively mitigates the network congestion and latency issues inherent in traditional cloud computing paradigms [4]. Among the various MEC technologies, computation offloading can effectively reduce processing delays for mobile UAVs and conserve device energy consumption [5]. In dynamic MEC networks, RL algorithms have been extensively explored for addressing computation offloading problems [6]. Applying mobile edge computing technology to unmanned systems and offloading computing tasks to edge nodes can effectively expand the computing power of UAVs, reduce energy consumption, and improve response efficiency. This technology is the key to overcoming the bottleneck of UAV resources [7].

Due to inherent flaws in RL, training RL algorithms requires powerful cloud servers, which drones must disclose sensitive data to. Attackers can easily obtain this sensitive information and use it to infer the value function in the RL algorithm. The value function represents the private information about the drone's action preferences in a given state, leading to privacy leaks [8–10].

However, the Differential Privacy (DP) technology introduced in existing research will reduce the performance of offloading strategies. Furthermore, the extent of performance loss caused by the addition of DP is unknown until after the offloading strategy has been trained. This leads to significant performance loss in the offloading strategy due to the introduction of DP, wasting computational power and resources. Therefore, establishing the relationship between privacy budget and performance loss before offloading strategy training is an important and necessary research issue, but existing research has failed to establish this quantitative relationship. These limitations pose challenges to the application of RL-based computation offloading methods in practical tasks.

For this purpose, we propose a UAV-centric privacy-preserving RL-based computation offloading approach. This approach adopts the LDP technique to perturb the decision trajectories of UAVs. In order to ensure that the RL-based computation offloading approach converges, we do not simply add LDP noise into decision trajectories. Instead, we model the computation offloading problem as a finite-horizon Markov decision process (FH-MDP), and then add LDP noise to randomize the frequency of each state–action pair in decision trajectories. Finally, we provide a regret bound for the proposed approach. The contributions of this paper are summarized as follows:

1. We propose a privacy-preserving computation offloading approach for multi-UAV MEC from a UAV-centric perspective. To achieve desirable UAV performance, we establish the relationship between privacy budget and the performance of the offloading policy.
2. To protect the privacy of decision trajectories of UAVs, we firstly model the computation offloading problem as a finite-horizon Markov decision process. Then, the LDP mechanism is employed to produce Gaussian noise or Randomized Response noise and perturb the count of state–action pairs in the decision trajectory with Gaussian noise or Randomized Response noise.
3. We theoretically derive the $\mathcal{O}(\sqrt{M}/\epsilon)$ regret bound and achieve (ϵ, δ')-LDP for the Gaussian perturbation mechanism and $(\epsilon, 0)$-LDP for the Randomized Response mechanism. Then, we conduct experiments to evaluate the effectiveness of the proposed approach.

The remainder of this paper is structured as follows. Section 2 reviews related works. Section 3 introduces the system model and problem formulation. The proposed approach is detailed in Section 4. Theoretical analysis and experimental evaluation are presented in Sections 5 and 6, respectively. Supplementary discussions are provided in Section 7, and Section 8 concludes the paper.

2. Related Works

Currently, many papers have proposed new methods for optimizing computational offloading strategies, which can be roughly divided into two categories. The first category involves algorithms based on optimization problems. For example, Yan et al. [11] studied the optimization problem of computational offloading strategies, constructed a potential game in an End-Edge Cloud Computing (EECC) environment, in which each user device selfishly minimizes its payoff, and developed two potential game-based algorithms. He et al. [12] used a queuing model to characterize all computing nodes in an environment and established a mathematical model to describe the scenario. They transformed the offloading decision of the target mobile device into three multi-variable optimization problems to study the trade-off between cost and performance. Zhao et al. [13] approached the partial task offloading problem as a dynamic long-term optimization problem to minimize task delays. Lyapunov stochastic optimization tools were used to separate long-term delay minimization from stability constraints, transforming the problem into a per-slot scheduling problem that meets the system's time-dependent stability requirements. Zhang et al. [14] established a joint optimization problem for offloading decisions, resource allocation, and trajectory planning. Dynamic Terminal Users (TUs) moved using a Gaussian–Markov stochastic model. The goal of the optimization problem was to maximize the minimum secure computing capacity of the TUs. A Joint Dynamic Programming and Bidding (JDPB) algorithm was proposed to solve the optimization problem. Liu et al. [15] introduced cooperative transmission and a reconfigurable intelligent surface (RIS) to jointly optimize user association, beamforming, power allocation, and the task partitioning strategy for local computation and offloading. They exploited the implicit relationships between these and transformed them into explicit forms for optimization. Yang et al. [16] minimized the total energy consumption of all devices by jointly optimizing the binary offloading patterns, CPU frequency, offloading power, offloading time, and intelligent reflecting surface (IRS) phase shift across all devices. Two greedy and penalty-based algorithms were proposed to solve the challenging nonconvex and discontinuous problem.

However, traditional optimization-based algorithms typically assume that the environment is known or static, requiring accurate system models such as channel models. Dynamic changes in the environment require remodeling and solving. And high-dimensional, continuous action spaces, or non-linear problems, are difficult to solve and require a large amount of computation. RL, on the other hand, excels at processing high-dimensional action state spaces through autonomous learning through interaction with the environment. RL can better adapt to dynamic and uncertain environments, discover strategies that are difficult to obtain through analytical methods, and does not require a precise model of the environment, relying on data-driven learning [17]. Therefore, applying RL technology to the training of computational offloading strategy decisions is a relatively common approach in current research.

There are many papers that have proposed RL methods for optimizing strategies in computation offloading. Huang et al. [18] introduced a deep reinforcement learning (DRL) framework that dynamically adjusts task offloading and wireless resources based on changing channel conditions, simplifying the optimization process. Lei et al. [19] developed a Q-learning and DRL framework that balances immediate rewards with long-term goals to minimize time delay and energy cost in mobile edge computing. Wang et al. [20] proposed a task offloading method based on meta-reinforcement learning, which can quickly adapt to new environments with a small number of gradient updates and samples. They proposed a method of coordinating first-order approximation and clipping proxy objectives, and customized sequence-to-sequence (seq2seq) neural networks to model the offloading strategy. Feng et al. [21] developed a distributed offloading algorithm using

deep Q-learning to reduce latency and manage energy cost in edge components of the Internet of Vehicles. Liu et al. [22] applied reinforcement learning based on a Markov chain to reduce task completion times within energy limits and created a cooperative task migration algorithm. The above works use RL and other approaches to optimize system latency and energy cost in computation offloading, but do not consider privacy protection.

Recently, the existing works aim to preserve users' offloading preferences [23], localization [24], and usage patterns [25] for RL-based computation offloading approaches. To provide strict and provable privacy guarantees, DP is adopted to mitigate privacy leakage [9]. Privacy protection in computation offloading can be divided into two main areas: (1) User data privacy protection: Cui et al. [26] proposed a novel lightweight certificate-less edge-assisted encryption scheme (CL-EAED) to offload computationally intensive tasks to edge servers, ensuring that edge-assisted processing does not expose sensitive information, effectively preventing data leakage and improving the efficiency and security of task offloading. A security-aware resource allocation algorithm is proposed for multimedia in MEC [27], focusing on data confidentiality and integrity to minimize latency and energy cost while ensuring security. (2) Privacy protection of offloading strategies: Xu et al. [28] developed a two-stage optimization strategy for offloading in IoT, with the aim of maximizing resource use while minimizing time delay and balancing privacy and performance. Wang et al. [24] proposes a disturbance region determination mechanism and an offloading strategy generation mechanism, which adaptively selects a suitable disturbance region according to a customized privacy factor and then generates an optimal offloading strategy based on the disturbance location within the determined region. However, when privacy protection is introduced, the policy performance will be lost, and the policy performance loss caused by privacy protection in the above studies is inestimable. In order to address this limitation, we propose our approach and make an intuitive comparison with the existing solutions in Table 1.

Table 1. An intuitive comparison with existing solutions.

Properties	Lei et al.'s [19]	Wang et al.'s [24]	Li et al.'s [27]	Cui et al.'s [26]	Xu et al.'s [28]	Ours
User value privacy	×	×	✓	✓	×	✓
Policy privacy	×	✓	×	×	✓	✓
Time delay	✓	✓	✓	✓	✓	✓
Energy cost	✓	✓	✓	✓	✓	✓
Performance regret	×	×	×	×	×	✓

× and ✓ denote support and nonsupport, respectively.

3. System Model and Problem Formulation

3.1. Network Model

In this paper, the network model of multi-UAV MEC is shown in Figure 1, which contains M UAVs, J edge computing servers (ECSs), and a cloud server (CS). In this system, each UAV uploads its own decision trajectory to the cloud server. The decision trajectory of the m-th UAV consists of a sequence of states s, actions a, and rewards r in chronological order. Formally, the decision trajectory of the m-th UAV is $Tr_m = \{(s_{t,m}, a_{t,m}, r_{t,m}) | t \in [0, T-1], m \in [1, M]\}$. Note that the decision trajectory may encompass sensitive UAV information from which its preferences and location can be deduced. The cloud server will adopt the RL algorithm to train the efficiency of the offloading policy, and then the cloud server will deploy the trained policy to the UAVs. In time slot t, the task of the m-th UAV is denoted by five variables $(I_m, v_l, v_s, B_m, \tau_m)$, where I_m is the size of the task in bits, v_l is the CPU frequency of the m-th UAV, v_s is the CPU frequency of the ECS j, B_m represents the remaining battery capacity of the drone, and τ_m represents the deadline for this task. We assume that all UAVs have the same

CPU frequency, while each ECS also has the same CPU frequency [23]. Then, the UAV makes the decision based on its own offloading policy. In this paper, we consider a full offloading model in which the UAV makes the offloading decision.

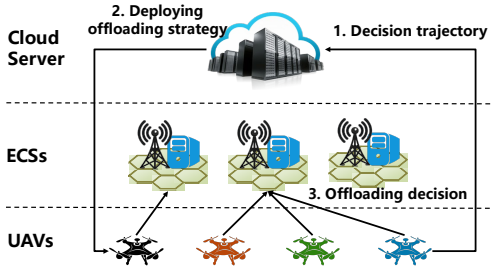


Figure 1. Network model of multi-UAV MEC.

3.2. Cost Model

There are two kinds of costs, time delay and energy cost. Note that due to the abundant energy in cloud servers, the energy consumption and latency of their training offloading policy are not taken into account. If the m-th UAV processes the task locally, the local time delay $D_{l,m}$ is taken by its onboard CPU to complete the computation. This local execution delay is calculated as follows:

$$D_{l,m}(t) = \frac{I_m(t)d}{v_l}. \tag{1}$$

Here, the local time delay $D_{l,m}$ is proportional to the total computational workload, which is the product of the task size $I_m(t)$ and the the number of CPU cycles d required per bit. It is inversely proportional to the UAV's processing speed v_l. A larger task or a more complex task will take longer, while a faster CPU will reduce the delay. According to [29], the local energy consumption $E_{l,m}(t)$ is given by

$$E_{l,m}(t) = \gamma v_l^2 I_m(t)d, \tag{2}$$

where γ is the effective capacitance coefficient based on the CPU architecture [30]. The local energy consumption is proportional to the square of the CPU frequency (v_l^2) and the total number of CPU cycles ($I_m(t) \times d$). If the m-th UAV decides to offload a task to an ECS j, the offloading time delay $D_{s,m}$ is the sum of two components: (1) the time taken to transmit the task data to the ECS over the wireless channel; (2) the time for the ECS to execute the task:

$$D_{s,m}(t) = \frac{I_m(t)}{L(t)} + \frac{I_m(t)d}{v_s}, \tag{3}$$

Here, the first term $\frac{I_m(t)}{L(t)}$ is the transmission delay. It depends on the task size $I_m(t)$ and the achievable transmission rate $L(t)$. The second term $\frac{I_m(t)d}{v_s}$ is the remote computation delay at the ECS, analogous to the local time delay but using the ECS's CPU frequency v_s. Based on [29], the energy cost of offloading $E_{s,m}$ from UAV m to the ECS j is given by

$$E_{s,m}(t) = \frac{pI_m(t)}{L(t)}, \tag{4}$$

where p is the transmission power of the m-th UAV. The energy cost of loading is the product of the transmission power p and the transmission time ($\frac{I_m(t)}{L(t)}$).

3.3. Privacy Threat

We consider an honest-but-curious cloud server with the capability to accurately execute the RL algorithm process for training offloading policy, though it is susceptible to

leaking decision trajectories of UAVs to potential adversaries. Once the adversary acquires the UAV's decision trajectories, they can utilize techniques such as inverse reinforcement learning (IRL) to extract the UAV's offloading preferences and usage patterns from the trajectory [9,23]. Then, this privacy information can be used to determine whether a specific UAV is present in the current area, potentially compromising UAV location privacy. Privacy concerns related to this issue have been extensively discussed in [24] with a heuristic privacy metric, and a similar privacy metric was considered in the healthcare IoT [31] and mobile blockchain network [32]. However, the existing approaches primarily focus on ensuring privacy from the perspective of offloading behavior in the case of a trusted cloud server, neglecting to address the potential privacy risks posed by an honest-but-curious cloud server. Hence, a previous work [33] introduced a privacy-aware offloading approach based on DP. However, the proposed approach fails to preserve UAV privacy during the training process of the offloading policy. Similarly to this paper, existing works [34] have proposed privacy-aware offloading approaches based on LDP. However, they fail to anticipate the impact of the privacy budget on the performance of the offloading policy before training.

3.4. Problem Formulation

In this section, we formulate an FH-MDP $\mathcal{M} = (\mathcal{S}, \mathcal{A}, q, r, T)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $q(\cdot|s,a)$ is a transition distribution for the new state under the current state, r is a distribution of reward with mean $r(s,a) \in [0,1]$, and $T \in \mathbb{N}^+$ is the horizon. In this context, the offloading policy π is defined as a series of mappings from states to actions. Formally, $\pi = \{\pi_t | 1 \leq t \leq T, \pi_t : \mathcal{S} \to \mathcal{A}\}$. Under the LDP privacy guarantee, we aim to find the optimal offloading policy $\pi^* = \{\pi_t^* | 1 \leq t \leq T, \pi_t : \mathcal{S} \to \mathcal{A}\}$ to minimize the total cost $G(t)$ of time delay and energy consumption for all UAVs, where $\pi_t^* = \arg\max_a Q_t^*(s,a)$ and $Q_t^*(s,a)$ is the optimal expected cumulative rewards under a state–action pair, which can be derived by the Bellman equations [35]. The objective is to minimize a weighted total cost $G(t)$ that balances the time delay and energy consumption of the system. The cost function is defined as

$$G(t) = \eta \mathcal{K}(\sum_{m=1}^{M}(D_m(t))) + (1-\eta)\mathcal{K}(\sum_{m=1}^{M}(E_m(t))). \tag{5}$$

$D_m(t)$ and $E_m(t)$ represent the composite delay and energy cost for m-th UAV, respectively, which are the sum of local and offloading components depending on the chosen action. $\mathcal{K}$ is the normalization function to scale the delay and energy values to a comparable range, and the weighted average η allows the system to prioritize either low latency (when η is close to 1) or energy efficiency (when η is close to 0).

Based on the formulated FH-MDP, the state, action, and reward of this paper are defined as follows:

1. *State*: The state $s_m(t) = (L(t), z)$, where $L(t)$ is the transmission rate and $z \in \{-1, 0, 1\}$. If $z = -1$, it means that the task of the m-th UAV has been terminated. The reason for this may be that the remaining battery capacity of the drone has been consumed, or the task has reached the deadline. If $z = 0$, it indicates that the task pf the m-th UAV has been completed; otherwise, it denotes that the processing is still ongoing.
2. *Action*: The action $a_m(t) \in 0, 1$, If $a_m(t) = 0$, it indicates that the m-th UAV processes its task locally; otherwise, the m-th UAV offloads its task to the ECS.
3. *Reward*: The reward is the total cost of all UAVs. Formally, $r_m(t) = G(t)$.

The above defines a standard FH-MDP for the computation offloading problem. However, as discussed in the privacy threat model (Section 3.3), directly using the original decision trajectory $Tr_m = \{(s_{t,m}, a_{t,m}, r_{t,m}) | t \in [0, T-1], m \in [1, M]\}$ for policy training on the edge server introduces privacy risks. To address this issue, we intro-

duce an LDP constraint. The core idea is that each drone perturbs its local trajectory $Tr_m = \{(s_{t,m}, a_{t,m}, r_{t,m}) | t \in [0, T-1], m \in [1, M]\}\}$ before offloading. The edge server then learns the offloading policy based solely on these perturbed trajectories. The next section (Section 4) will elaborate on the LDP perturbation mechanism and the corresponding offloading policy learning mechanism under this privacy guarantee.

4. Proposed Approach

4.1. Overview

In the proposed approach, the core idea is to add LDP noise to the decision trajectory of each UAV during policy training. However, simply introducing LDP noise into the decision trajectory will disrupt the temporal coherence of the decision trajectory, which will prevent algorithmic convergence. Hence, we adopt the Gaussian perturbation mechanism and the Randomized Response mechanism based on [36] to randomize the frequency of each state–action pair in the decision trajectory. Then, the honest-but-curious cloud server adopts the perturbed decision trajectories to train the offloading policy. In general, the proposed approach is divided into two parts, namely the *LDP perturbation mechanism* and *offloading policy learning mechanism*, with their detailed contents provided in the following.

4.2. LDP Perturbation Mechanism

During the training process, each UAV adopts the LDP mechanism to perturb its decision trajectory Tr_m locally for privacy preservation. Then, the perturbed decision trajectory $\widehat{Tr}_m$ is offloaded to the cloud served by each UAV. In this paper, we provide two LDP perturbation mechanisms, the Gaussian perturbation mechanism and the Randomized Response mechanism. The details are shown in Algorithm 1 and Algorithm 2, respectively.

4.2.1. Gaussian Perturbation Mechanism

At the beginning of Algorithm 1, the input is the decision trajectory of the m-th UAV Tr_m, and the LDP parameters ϵ and c. For each state–action pair (s, a) in Tr_m (Line 1), the cumulative reward $R_m(s, a)$ and visiting frequency to the state–action pair $C_m(s, a)$ are calculated by Equations (6) and (7), respectively (Line 2):

$$R_m(s, a) = \sum_{t=1}^{T} r_{t,m} \mathbb{I}_{\{s_{t,m}=s, a_{t,m}=a\}}, \tag{6}$$

and

$$C_m(s, a) = \sum_{t=1}^{T} \mathbb{I}_{\{s_{t,m}=s, a_{t,m}=a\}}, \tag{7}$$

where $\mathbb{I}$ is the indicator function. Then, the noises $X_m(s, a)$ and $Y_m(s, a)$ are generated independently by Gaussian noise with a standard deviation of σ (Line 3), where $\sigma = \frac{T}{\epsilon}$ [36]. The perturbed cumulative reward $\hat{R}(s, a)$ and perturbed visiting frequency to the state–action pair $\hat{C}(s, a)$ are calculated (Lines 4–5). Moreover, the visiting frequency to the trajectory $C'_m(s, a, s')$ is calculated as follows (Line 7):

$$C'_m = \sum_{t=1}^{T-1} \mathbb{I}_{\{s_{t,m}=s, a_{t,m}=a, s_{t+1,m}=s'\}}, \tag{8}$$

Then the perturbed $\hat{C'}_m(s, a, s')$ is calculated based on the noise $Z_m(s, a, s')$ (Lines 8–9). Finally, the perturbed cumulative reward $\hat{R}_m(s, a)$, perturbed visiting frequency to the state–action pair $\hat{C}_m(s, a)$, and perturbed visiting frequency to the trajectory $C'_m(s, a, s')$ are returned (Line 12).

Algorithm 1: Gaussian perturbation mechanism.

Input: $Tr_m = \{(s_{t,m}, a_{t,m}, r_{t,m}) | t < T\}$, Parameters: ϵ, c;

for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
  Calculate $R_m(s, a)$, and $C_m(s, a)$;
  $X_m(s, a), Y_m(s, a) \sim \mathcal{N}(0, \sigma^2)$;
  $\hat{R}_m(s, a) = X_m(s, a) + R_m(s, a)$;
  $\hat{C}_m(s, a) = Y_m(s, a) + C_m(s, a)$;
  for $s' \in \mathcal{S}$ **do**
    Calculate $C'(s, a, s')$;
    $Z_m(s, a, s') \sim \mathcal{N}(0, \sigma^2)$;
    $\hat{C'}_m(s, a, s') = Z_m(s, a, s') + C'_m(s, a, s')$;
  end for
end for
return $\hat{R}_m(s, a)$, $\hat{C}_m(s, a)$, $C'_m(s, a, s')$

4.2.2. Randomized Response Mechanism

The core of the Random Response mechanism algorithm is to perturb the binary indicators in the decision trajectory through biased Bernoulli sampling. These Bernoulli distributions are parameterized by the privacy budget ϵ_0. The detailed algorithm is shown in Algorithm 2.

Algorithm 2: Randomized Response mechanism.

Input: $Tr_m = \{(s_{t,m}, a_{t,m}, r_{t,m}) | t < T\}$, Parameters: ϵ_0

for $(s, a) \in \mathcal{S} \times \mathcal{A}$ **do**
  for $t = 1, ..., T$ **do**
    Let $I_t = \mathbb{I}_{\{s_{t,m}=s, a_{t,m}=a\}}$
    Sample $X_t \sim \text{Ber}\left(\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1} \cdot r_{t,m} \cdot I_t + \frac{1}{e^{\epsilon_0}+1}\right)$
    Update $\hat{R}_m(s, a) \leftarrow \hat{R}_m(s, a) + k \cdot (X_t - c)$, where $k = \frac{e^{\epsilon_0}+1}{e^{\epsilon_0}-1}$, $c = \frac{1}{e^{\epsilon_0}+1}$
    Sample $Y_t \sim \text{Ber}\left(\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1} \cdot I_t + \frac{1}{e^{\epsilon_0}+1}\right)$
    Update $\hat{C}_m(s, a) \leftarrow \hat{C}_m(s, a) + k \cdot (Y_t - c)$
    if $t \leq T$ **then**
      for $s' \in \mathcal{S}$ **do**
        Let $I'_t = \mathbb{I}_{\{s_{t,m}=s, a_{t,m}=a, s_{t+1,m}=s'\}}$
        Sample $Z_t \sim \text{Ber}\left(\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1} \cdot I'_t + \frac{1}{e^{\epsilon_0}+1}\right)$
        Update $\hat{C}'_m(s, a, s') \leftarrow \hat{C}'_m(s, a, s') + k \cdot (Z_t - c)$
      end for
    end if
  end for
end for
return $\hat{R}_m(s, a)$, $\hat{C}_m(s, a)$, $C'_m(s, a, s')$

At the beginning of Algorithm 2, the input is the decision trajectory of the m-th UAV Tr_m. The algorithm then iterates through each time step t in the trajectory (line 2). For each time step, the algorithm first checks whether the current state–action pair matches the target pair (s, a) using the indicator function I_t (line 3).

The perturbation process consists of three main parts: (1) Reward Perturbation (lines 4–5): For the reward component, the algorithm samples a Bernoulli variable X_t with a probability proportional to the actual reward value $r_{t,m}$ when the state–action pair matches.

The sampling probability is carefully designed to strike a balance between authenticity and random noise. The perturbed reward is then updated using a linear transformation to ensure an unbiased estimate. (2) Access Count Perturbation (Lines 6–7): Similarly, for access counts, the algorithm samples Y_t from a Bernoulli distribution, where the probability reflects whether a state–action pair was actually accessed. This creates a noisy access record while preserving statistical properties. (3) Transition Count Perturbation (Lines 8–14): For state transitions, the algorithm also considers the next state s' and perturbs the transition indicator. This is crucial for preserving the Markov property in the learning model.

The key parameters k and c are derived from the privacy budget ϵ_0 and ensure that the expected value of each perturbation is equal to its true value, thus providing unbiased estimates under local differential privacy. The Bernoulli parameter is derived from the privacy budget ϵ_0 using the transformation $\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1} \cdot \text{true value} + \frac{1}{e^{\epsilon_0}+1}$, thus ensuring the $(\epsilon, 0)$-LDP guarantee.

It should be noted that, according to [36], the noises $X_m(s,a)$, $Y_m(s,a)$, and $Z_m(s,a,s')$ satisfy

$$\left|\hat{R}_m(s,a) - R_m(s,a)\right| \leq f_{m,1}\left(\epsilon, \delta', \delta\right), \tag{9}$$

$$\left|\hat{C}_m(s,a) - C_m(s,a)\right| \leq f_{m,2}\left(\epsilon, \delta', \delta\right), \tag{10}$$

$$\left|\sum_{s'} C'_m - \hat{C'}_m\right| \leq f_{m,3}\left(\epsilon, \delta', \delta\right), \tag{11}$$

and

$$\left|C'_m - \hat{C'}_m\right| \leq f_{m,4}\left(\epsilon, \delta', \delta\right). \tag{12}$$

The above requirements can be satisfied by several perturbation mechanisms, such as Gaussian, Laplace, Randomized Response, and bounded noise mechanisms. But the regret and privacy guarantees of these perturbation mechanisms are of great importance. It can be seen from the above definition that these four functions are increasing functions with respect to m and are decreasing functions with respect to δ. Moreover, according to the characteristics of the local differential privacy mechanism, these four limited strict positive functions can be calculated. We will show the calculation in Section 5.

4.3. Offloading Policy Learning Mechanism

In the offloading policy learning mechanism, the policy is updated by a modified Bellman equation, with details shown in Algorithm 3. In the algorithm, the inputs are the privacy parameters ϵ, δ', δ, and the outputs of Algorithm 1. Then, for each UAV, the privatization reward $\hat{r}_m(s,a)$ and privatization transition function $\hat{w}(s' \mid s,a)$ are calculated based on Equation (13) and Equation (14), respectively, which are shown as follows (Line 2).

$$\hat{r}_m(s,a) = \frac{\hat{R}_m(s,a)}{\hat{C}_m(s,a) + \alpha f_{m,2}(\epsilon, \delta', \delta)}, \tag{13}$$

and

$$\hat{w}_m\left(s' \mid s,a\right) = \frac{\hat{C'}_m(s,a,s')}{\hat{C}_m(s,a) + \alpha f_{m,3}(\epsilon, \delta', \delta)}, \tag{14}$$

where α is a parameter for precision selection and $\alpha \geq 1$. Based on Prop.4 of [36], to achieve the $(\epsilon, 1-2\delta)-$LDP, the $\hat{r}_m(s,a)$ and $r(s,a)$, as well as $w_m(s' \mid s,a)$ and $\hat{w}_m(s' \mid s,a)$, should meet the following constraints, respectively:

$$|r_m(s,a) - \hat{r}_m(s,a)| \leq p_m(s,a), \tag{15}$$

and

$$|w_m - \hat{w}_m| \leq p'_m(s,a), \tag{16}$$

where $\epsilon > 0, \delta' \geq 0, \delta > 0, \alpha > 1$. Then, the modified Bellman equation is shown as follows to update the Q function (Line 3).

$$Q_{t,m}(s,a) = \hat{r}_m(s,a) + b_{t,m}(s,a) + \hat{w}_m(\cdot \mid s,a)^\top V_{t+1,m}, \tag{17}$$

where $b_{t,m}(s,a) = (T-t+1) \cdot p'_m(s,a) + p_m(s,a)$ is a bias compensation term to ensure convergence under LDP noise. Therefore, the offloading policy is calculated by (Line 3)

$$\pi_{t,m}(s) = \arg\max Q_{t,m}(s,a). \tag{18}$$

Algorithm 3: Offloading policy learning mechanism.

Input: $\hat{R}_m(s,a)$, $\hat{C}_m(s,a)$, $C'_m(s,a,s')$, Parameters ϵ_0, δ_0, δ

1: **for** $m \in [1, M]$ **do**
2: Compute $\hat{r}_m(s,a)$, $\hat{w}_m(s' \mid s,a)$ via Equation (13) and Equation (14), respectively;
3: According to Equations (17) and (18), update the offloading policy π_m;
4: Send π_m to m-th UAV. After executing the offloading policy, m-th UAV collects the decision trajectory Tr_m;
5: m-th UAV sends back privatized version $\mathcal{L}(Tr_m)$ via Algorithm 1;
6: **end for**

Finally, the cloud server sends π_m to the m-th UAV, which will collect the decision trajectory Tr_m and send back a privatized version $\widehat{Tr}_m$ via Algorithm 1.

Our perturbation mechanism adds LDP noise to the frequency of state–action pairs, maintaining the convergence of reinforcement learning while preserving privacy. The Markov property relies on the independence of state transitions, and the frequency perturbation is performed on aggregate statistics, which does not destroy the Markov property of individual state transitions. According to the theory of Garcelon et al. [36], as long as the perturbed reward and transition probability estimates are asymptotically consistent and updated via the modified Bellman equation, the RL algorithm will converge under the LDP condition. In Section 5, we prove that the regret bound is $\mathcal{O}(\sqrt{M}/\epsilon)$, which theoretically guarantees the convergence of the algorithm in the long run.

5. Theoretical Analysis

In this section, we give the formal proof of $(\epsilon, \delta')-$LDP and regret bound of the proposed approach. The derivation of the regret bound implies the convergence of the algorithm under LDP perturbations. That is, when the number of training rounds is large enough, the policy performance will be close to the optimal policy. Firstly, we prove that the Gaussian perturbation mechanism has the $(\epsilon, \delta')-$LDP guarantee through Theorem 1.

Theorem 1. *Given* $0 \leq \epsilon < 1$ *and* $\delta' > 0$, *and* $c > 4\ln\left(\frac{24}{\delta'}\right)$, *the Gaussian perturbation mechanism can achieve* (ϵ, δ')*-LDP.*

Proof. Based on Prop.10 in [36], we assume that for the two trajectories $Tr = \{(s_t, a_t, r_t) | t < T\}$, $Tr' = \{(s'_t, a'_t, r'_t) | t < T\}$, and the perturbed trajectories $\mathcal{L}(Tr) = (\hat{R}_{Tr}, \hat{C}_{Tr}, \hat{C'}_{Tr})$ and $\mathcal{L}(Tr') = (\hat{R}_{Tr'}, \hat{C}_{Tr'}, \hat{C'}_{Tr'})$.

For a given vector $r \in \mathbb{R}^{S\times A}$, and since the Gaussian distribution is symmetric, $\forall(s,a)$, we have

$$\frac{\mathbb{P}\left(\hat{R}_{Tr}(s,a)=r(s,a)\right)}{\mathbb{P}\left(\hat{R}_{Tr'}(s,a)=r(s,a)\right)}=\prod_{s,a}\frac{\mathbb{P}\left(X_{Tr}=\sum_{t=1}^{T}r_t\mathbb{I}_{\{s_t=s,a_t=a\}}-r\right)}{\mathbb{P}\left(X_{Tr'}=\sum_{t=1}^{T}r'_t\mathbb{I}_{\left\{s'_t=s,a'_t=a\right\}}-r\right)}. \tag{19}$$

However, considering the squared term and following from Cauchy–Schwartz, we have the inequality

$$\frac{\mathbb{P}\left(\hat{R}_{Tr}(s,a)=r(s,a)\right)}{\mathbb{P}\left(\hat{R}_{Tr'}(s,a)=r(s,a)\right)}\leq\exp\left(\frac{1}{2\sigma^2}\left(2\sqrt{2}T\sqrt{\sum_{s,a}r^2}+3T^2\right)\right), \tag{20}$$

where $\sigma=cT/\epsilon$, where c is a constant value. Then, for $c^2\geq 4\ln\left(\frac{3}{\delta_1}\right)$, δ_1 should satisfy $\mathbb{P}(X_{Tr}\in R_2)\leq\delta_1$ and $\mathbb{P}(Y_{Tr'}\in R_2)\leq\delta_1$. Therefore, $\forall(s,a)$, we have

$$\mathbb{P}\left(\hat{R}_{Tr}(s,a)=r(s,a)\right)\leq(\epsilon/3)\mathbb{P}\left(\hat{R}_{Tr'}(s,a)=r(s,a)\right)+\delta_1. \tag{21}$$

The same goes for $\hat{C}$ and $\hat{C}'$. Then, because $X_{Tr}(s,a)$, $Y_{Tr}(s,a)$, and $Z_{Tr}(s,a,s')$ are independent, for any two trajectories Tr and Tr', we have

$$\begin{aligned}&\mathbb{P}\left(\hat{R}_{Tr}=r,\hat{C}_{Tr}=n,\hat{C}'_{Tr}=n'\right)=\mathbb{P}\left(\hat{R}_{Tr}=r\right)\mathbb{P}\left(\hat{C}_{Tr}=n\right)\mathbb{P}\left(\hat{C}'_{Tr}=n'\right)\\&\leq e^{\epsilon}\mathbb{P}\left(\hat{R}_{Tr}=r'\right)\mathbb{P}\left(\hat{C}_{Tr'}=n\right)\cdot\mathbb{P}\left(\hat{C}'_{Tr'}=n'\right)+2\delta_1\exp(2\epsilon/3)+2\delta_1^2\exp(\epsilon/3)+\delta_1^3.\end{aligned} \tag{22}$$

Thus, by choosing $\delta_1=\delta'/8$, it holds that $2\delta_1\exp(2\epsilon/3)+2\delta_1^2\exp(\epsilon/3)+\delta_1^3\leq\delta'$ for $\epsilon\leq 1$, and so we can conclude that the Gaussian mechanism is (ϵ,δ')-LDP. □

Then, we prove that the Randomized Response mechanism has the $(\epsilon,0)-$LDP guarantee through Theorem 2.

Theorem 2. *For $\epsilon>0$ and the parameter $\epsilon_0=\epsilon/6T$, Algorithm 2 is $(\epsilon,0)-LDP$.*

Proof. Like the proof of Theorem 1, consider two trajectories $Tr=\{(s_g,a_g,r_g)|g<G\}$, $Tr'=\{(s'_g,a'_g,r'_g)|g<G\}$, the perturbed trajectory $\mathcal{G}(Tr)=(\hat{R}_{Tr},\hat{C}_{Tr},\hat{C}'_{Tr})$ and $\mathcal{G}(Tr')=(\hat{R}_{Tr'},\hat{C}_{Tr'},\hat{C}'_{Tr'})$.

For a given $r\in\left\{\frac{-1}{e^{\epsilon_0}-1},\frac{e^{\epsilon_0}}{e^{\epsilon_0}-1}\right\}^{G|\mathcal{S}||\mathcal{A}|}$, and for each $(g,s,a)\in G\times\mathcal{S}\times\mathcal{A}$, define $y^r_{g,s,a}=\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1}r+\frac{1}{e^{\epsilon_0}+1}$ belongs to $\{0,1\}$ because $r\in\left\{\frac{-1}{e^{\epsilon_0}-1},\frac{e^{\epsilon_0}}{e^{\epsilon_0}-1}\right\}^{G|\mathcal{S}||\mathcal{A}|}$, and we have

$$\begin{aligned}&\frac{\mathbb{P}\left(\forall(g,s,a),\hat{R}_{Tr}(g,s,a)=r_{g,s,a}\mid Tr\right)}{\mathbb{P}\left(\forall(g,s,a),\hat{R}_{Tr'}(g,s,a)=r_{g,s,a}\mid Tr'\right)}\\&=\prod_{g,s,a}\left(\frac{(e^{\epsilon_0}-1)r_g\mathbb{G}+1}{(e^{\epsilon_0}-1)r'_g\mathbb{G}'+1}\right)^{y^r_{g,s,a}}\times\left(\frac{e^{\epsilon_0}-(e^{\epsilon_0}-1)r_g\mathbb{G}}{e^{\epsilon_0}-(e^{\epsilon_0}-1)r'_g\mathbb{G}'}\right)^{1-y^r_{g,s,a}},\end{aligned} \tag{23}$$

where $\mathbb{G}=\mathbb{1}_{\{s_g=s,a_g=a\}}$, $\mathbb{G}'=\mathbb{1}_{\left\{s'_g=s,a'_g=a\right\}}$.

Then, for a given (g,s,a), since $r_g\in[0,1]$, the above formula can be simplified as

$$\frac{(e^{\epsilon_0}-1)r_g\mathbb{1}_{\{s_g=s,a_g=a\}}+1}{(e^{\epsilon_0}-1)r'_g\mathbb{1}_{\left\{s'_g=s,a'_g=a\right\}}+1}\leq\exp(\epsilon_0\mathbb{S})\frac{e^{\epsilon_0}-(e^{\epsilon_0}-1)r_g\mathbb{1}_{\{s_g=s,a_g=a\}}}{e^{\epsilon_0}-(e^{\epsilon_0}-1)r'_g\mathbb{1}_{\left\{s'_g=s,a'_g=a\right\}}}\leq\exp(\epsilon_0\mathbb{S}), \tag{24}$$

where $\mathbb{S}=\mathbb{1}_{\{s_g=s,a_g=a\}}+\mathbb{1}_{\left\{s'_g=s,a'_g=a\right\}}$. Therefore, through the inequality (24), we get

$$\begin{aligned}&\frac{\mathbb{P}\left(\forall(g,s,a),\hat{R}_{Tr}(g,s,a)=r_{g,s,a}\mid Tr\right)}{\mathbb{P}\left(\forall(g,s,a),\hat{R}_{Tr'}(g,s,a)=r_{g,s,a}\mid Tr'\right)}\\&\leq\prod_{g,s,a}\exp\left(y^{r}_{g,s,a}\epsilon_0\mathbb{S}+\left(1-y^{r}_{g,s,a}\right)\epsilon_0\mathbb{S}\right)=\prod_{g,s,a}\exp(\epsilon_0\mathbb{S})=\exp(2\epsilon_0 G).\end{aligned}\tag{25}$$

The same is true for $\hat{C}$ and $\hat{C}'$. From this, it can be concluded that when $\epsilon_0=\epsilon/6G$, the formula based on the Randomized Response mechanism is $(\epsilon,0)-$LDP. □

Then, we prove the regret bound of the proposed approach. Before proving, we provide the definition of regret.

Definition 1. *Given the finite-horizon Markov decision process (FH-MDP) $\mathcal{M}=(\mathcal{S},\mathcal{A},q,r,T)$ in Section 3.4, the regret $\Gamma(M)$ in this paper is defined as the performance of the offloading policy, which is defined as the cumulative difference between $V_1^*(s_{1,m})$ and $V_1^{\pi_m}(s_{1,m})$ for all UAVs:*

$$\Gamma(\mathcal{M})=\sum_{m=1}^{M}(V_1^*(s_{1,m})-V_1^{\pi_m}(s_{1,m})).\tag{26}$$

Theorem 3. *For any number of states $|\mathcal{S}|\geq 3$, actions $|\mathcal{A}|\geq 2$, and $T\geq 2\log_{|\mathcal{A}|}(|\mathcal{S}|-2)+2$, the lower regret bound of the proposed approach satisfies $\mathbb{E}_{\mathcal{M}}(\Gamma(\mathcal{M}))\geq\Omega\left(\frac{T\sqrt{|\mathcal{S}||\mathcal{A}|M}}{\min\{\exp(\epsilon)-1,1\}}\right)$, where $\mathcal{M}$ is the FH-MDP in this paper.*

Proof. We set the FH-MDP with $|\mathcal{S}|$ states and $|\mathcal{A}|$ actions. Our FH-MDP is a $|\mathcal{A}|$-ary tree with $|\mathcal{S}|-2$ states, and each node has $|\mathcal{A}|$ child nodes. We use $x_t,\ldots,x_T$ to represent the leaf nodes of this tree. Each leaf node can be converted to receive a reward of 1 or 0. And each leaf node converts to reward 0 and reward 1 with the same probability. And we set that there exists a unique action $a^\star$ and leaf $x_{t^\star}$ such that $\mathbb{P}(1\mid x_{t^\star},a^\star)=\frac{1}{2}+\Delta$, $\mathbb{P}(0\mid x_{t^\star},a^\star)=\frac{1}{2}-\Delta$ for a chosen Δ, and when $\Delta=0$, $\mathbb{P}_0(1\mid x_{t^\star},a^\star)=\frac{1}{2}$, $\mathbb{P}_0(0\mid x_{t^\star},a^\star)=\frac{1}{2}$.

We assume that $T\geq 2\ln(|\mathcal{S}|-2)/\ln(|\mathcal{A}|)+2$, $b-1$ is the depth of the tree, and the depth of the leaf node is b-1 or b-2. Here, we assume that all leaf nodes $x_t,\ldots,x_T$ are at $b-1$. So the number of leaf nodes is $T=|\mathcal{A}|^{b-1}\geq(|\mathcal{S}|-2)/2$. Thus, our value function is as follows for a policy π:

$$V^{\pi}(0)=(T-b)(1/2+\Delta\mathbb{P}(s_{b-1}=x_{i^\star},a_{b-1}=a^\star)).\tag{27}$$

Therefore, according to Definition 1, the regret is given as follows:

$$U(M,O)=(T-b)\Delta(M-\sum_{m=1}^{M}\mathbb{P}(s_{m,b-1}=x_{t^\star},a_{m,b-1}=a^\star))\tag{28}$$

where we define that

$$F(M,O)=\sum_{m=1}^{M}\mathbb{E}_{\left\{s_{m,b-1}=x_{t^\star},a_{m,b-1}=a^\star\right\}}.$$

Thus, $F(M,O)$ is a function of the history observed by the algorithm, and $O=(x_{t^\star},a^\star)$ is the optimal state–action pair.

Considering the LDP setting, the history can be written as

$$\mathcal{L}(\mathcal{T}_M)=\{\mathcal{L}(Tr_m)\mid m\leq M\},\tag{29}$$

where $\mathcal{L}$ is a local privacy protection mechanism satisfying ϵ-LDP and $Tr_m=\{(s_{t,m},a_{t,m},r_{t,m})\mid t\leq T\}$ is the history trajectory. Thus $F(M,O)$ is a function of $\mathcal{L}(\mathcal{T}_M)$. And according to [37], we have

$$\mathbb{E}(F(M,O)) \leq \mathbb{E}_0(F(M,O)) + M\sqrt{\mathrm{KL}(\mathbb{P}_0(\mathcal{L}(\mathcal{T}_M))\|\mathbb{P}(\mathcal{L}(\mathcal{T}_M)))}, \tag{30}$$

and because of the privacy mechanism $\mathcal{L}$, we also have

$$\mathbb{E}(F(M,O)) \leq \mathbb{E}_0(F(M,O)) + M\sqrt{\mathrm{KL}(\mathbb{P}_0(\mathcal{T}_M)\|\mathbb{P}(\mathcal{T}_M))}. \tag{31}$$

We set $Bound1 = \mathrm{KL}(\mathbb{P}_0(\mathcal{L}(\mathcal{T}_M))\|\mathbb{P}(\mathcal{L}(\mathcal{T}_M)))$ and $Bound2 = \mathrm{KL}(\mathbb{P}_0(\mathcal{T}_M)\|\mathbb{P}(\mathcal{T}_M))$, and $\mathbb{E}(F(M,O))$ is a special case when $\Delta = 0$.

According to Equations (30) and (31), we have

$$\mathbb{E}(F(M,K)) \leq \mathbb{E}_0(F(M,K)) + M\min\{\sqrt{\mathrm{KL}(\mathbb{P}_0(\mathcal{L}(\mathcal{T}_M))\|\mathbb{P}(\mathcal{L}(\mathcal{T}_M))}, \sqrt{\mathrm{KL}(\mathbb{P}_0(\mathcal{T}_M)\|\mathbb{P}(\mathcal{T}_M))}\}. \tag{32}$$

Therefore, according to [36], we get

$$Bound1 \leq 2(e^\epsilon - 1)^2 * \ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_0(F(M,O)), \tag{33}$$

and

$$Bound2 = \frac{1}{2}\ln\left(\frac{1}{1-4\Delta^2}\right)\mathbb{E}_0(F(M,O)). \tag{34}$$

Therefore, combining Equations (33) and (34),

$$\mathbb{E}(F(M,O)) \leq \mathbb{E}_0(F(M,O)) + M\min\left\{\sqrt{2}(e^\epsilon - 1), \frac{1}{\sqrt{2}}\right\}\sqrt{\mathbb{E}_0(F(M,O))\ln\left(\frac{1}{1-4\Delta^2}\right)}. \tag{35}$$

Here, we set that $\mathbb{E}_O$ is the expectation of the random variable $(x_{t^\star}, a^\star)$, and then there is

$$\mathbb{E}_O\mathbb{E}_0(F(M,O)) = \frac{M}{T|\mathcal{A}|}. \tag{36}$$

Thus, according to Jensen's inequality, we have

$$\mathbb{E}_O U(M,O) \geq (H-d)\Delta K\left(1 - \frac{1}{LA} - \min\left\{\sqrt{2}(e^\epsilon - 1), \frac{1}{\sqrt{2}}\right\}\sqrt{\frac{K}{LA}\ln\left(1 + \frac{4\Delta^2}{1-4\Delta^2}\right)}\right). \tag{37}$$

So when $T|\mathcal{A}| \geq 2$, $M \geq \frac{T|\mathcal{A}|}{\min\{8(e^\epsilon-1),4\}^2}$, $\Delta = \sqrt{\frac{T|\mathcal{A}|}{K}} \times \frac{1}{16\sqrt{2}\min\left\{(e^\epsilon-1),\frac{1}{2}\right\}}$, we have

$$\min\left\{\sqrt{2}(\exp(\epsilon) - 1), \frac{1}{\sqrt{2}}\right\}\sqrt{\frac{M}{T\mathcal{A}}\ln\left(1 + \frac{4\Delta^2}{1-4\Delta^2}\right)} \leq \frac{1}{4}. \tag{38}$$

Therefore, we have

$$\max U(M,O) \geq \mathbb{E}_O U(M,O) \geq \frac{(T-b)\sqrt{MT\mathcal{A}}}{64\min\left\{(\exp(\epsilon) - 1), \frac{1}{2}\right\}}, \tag{39}$$

and

$$U(M,O^\star) \geq \frac{(T-b)\sqrt{MT|\mathcal{A}|}}{64\min\left\{(\exp(\epsilon) - 1), \frac{1}{2}\right\}}, \tag{40}$$

where there is $O^\star$ that $\max U(M,O) = U(M,O^\star)$, because O is a finite random variable. Finally, we can prove that there exists an FH-MDP such that its regret is $\Omega\left(\frac{T\sqrt{|\mathcal{S}||\mathcal{A}|M}}{\min\{1,\exp(\epsilon)-1\}}\right)$. □

6. Experiment Evaluation

6.1. Experimental Settings

In this section, we conduct the experiments to evaluate the convergence and the regret of the proposed approach. The experiments are simulated using Python and tested on drones. In the experiment, according to [38], we set the number of UAVs to $M = 25$ and the number of ECSs to $J = 6$. For each UAV, the discount factor δ is selected from $\{0.1, 0.2, 0.3, 0.5, 0.7, 0.8, 0.9\}$, the task size I_m is selected from {100 KB, 300 KB}, and the transmission rate p is selected from {2 KB/s, 5 KB/s}. The CPU frequency of the m-th UAV v_l and j-th ECS v_s are set to 1 GHz and 3 GHz, respectively, while the number of CPU cycles required to complete each bit of the task is $d = 1000$. The effective capacitance coefficient γ is set to 10^{-26}. These parameters were chosen based on the performance of commercial drones and edge servers. For example, the task size {100 KB, 300 KB} represents a single image frame or sensor data packet, while the CPU frequency (1 GHz for UAVs and 3 GHz for edge servers) is consistent with typical embedded processors such as the ARM Cortex-A78 and lightweight edge servers. The transmission rate simulates realistic wireless channel conditions over 4G/5G links. Then, to train the offloading policy, the episode is set to 7500, and there are $T = 2$ learning steps in each episode.

Finally, we designed five experiments:

(1) We adopt the average loss to evaluate the convergence of the proposed approach by varying task sizes and privacy budgets using the Gaussian perturbation mechanism and the Randomized Response mechanism, respectively.

(2) We show the regret value of the proposed approach under different transmission rates and privacy budgets using the Gaussian perturbation mechanism and the Randomized Response mechanism, respectively.

(3) We compare our approach with a scheme without privacy protection to analyze the performance of the scheme after adding LDP perturbation.

(4) We select different discount factors, a fixed task size, and a transmission rate under the Gaussian perturbation mechanism to evaluate the impact of discount factors on the approach.

(5) We compare our approach with other privacy protection schemes based on reinforcement learning to see the advantages of our scheme.

6.2. Experiment Results

In this section, we use average loss to comprehensively evaluate user-related performance metrics, including time delay and energy consumption. The metric defined by Equation (5) is a weighted sum of time delay and energy consumption that directly reflects the performance at the user level. Furthermore, we use regret to measure the cumulative performance gap between the private mechanism and the offloading policy.

6.2.1. Convergence Evaluation of the Approach

The convergence of the proposed approach based on the Gaussian perturbation mechanism and the Randomized Response mechanism is shown in Figure 2. The privacy parameters ϵ are set to $\{0.1, 0.2, 0.3, 0.9\}$, respectively, and we set the task sizes to $I_m = 100$ KB and 300 KB. From the experimental results, we can see the following: (1) The average loss of the offloading strategy gradually decreases with an increase in iterations, and finally, the algorithm reaches convergence. (2) For different task sizes, when the iterations reach 500, the algorithm based on the Gaussian perturbation mechanism reaches convergence, and when the iterations reach 300, the the algorithm based on the Randomized Response mechanism reaches convergence. The convergence speed of the Randomized Response mechanism is faster than the convergence speed of the algorithm based on the Gaussian per-

turbation mechanism. (3) As the task size increases, the convergence speed of the algorithm gradually slows down, and the task size has a certain influence on the convergence speed of the algorithm. (4) The offloading policy based on the Random Response perturbation mechanism has less performance loss compared to the Gaussian perturbation mechanism.

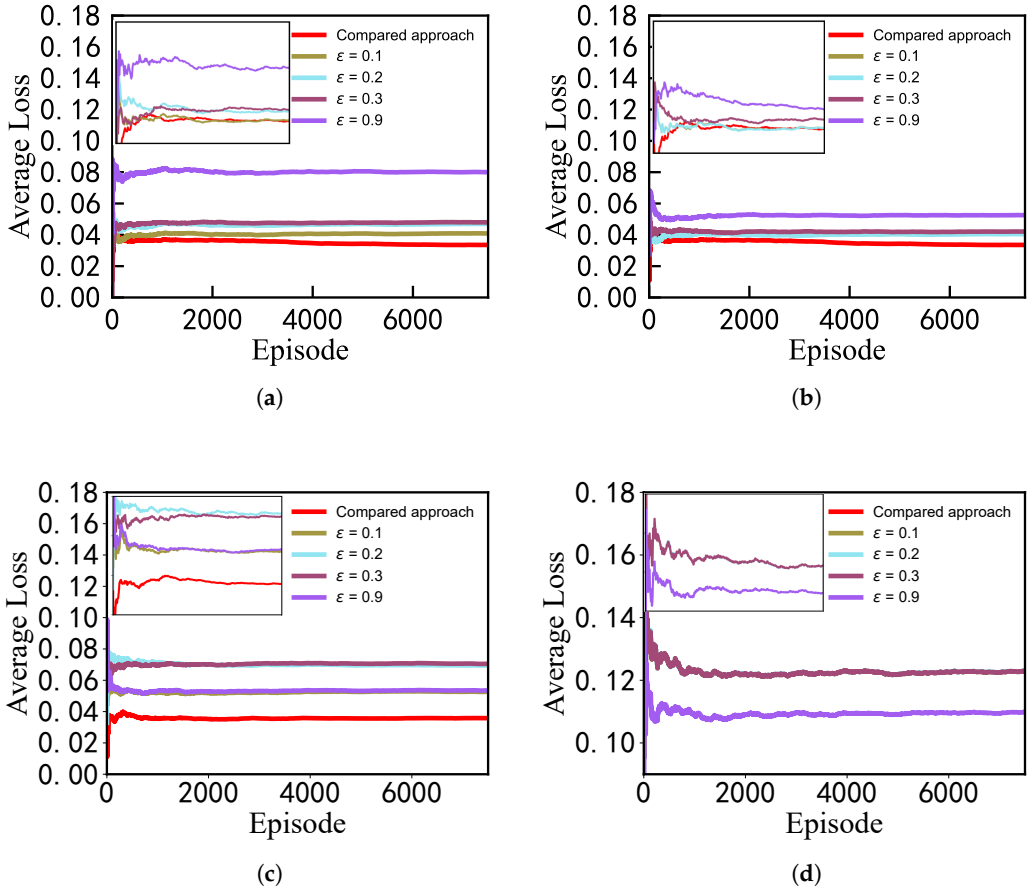


Figure 2. (**a**) $I_m = 100$ KB (Gaussian perturbation mechanism). (**b**) $I_m = 300$ KB (Gaussian perturbation mechanism). (**c**) $I_m = 100$ KB (Randomized Response mechanism). (**d**) $I_m = 300$ KB (Randomized Response mechanism).

Although an increase in task size will lead to a slower convergence speed, the algorithm can still converge quickly within a reasonable range. Therefore, the proposed approach has good robustness and adaptability under different task sizes and privacy parameters.

6.2.2. Regret Value Evaluation of the Approach

The regret value of the proposed approach based on the Gaussian perturbation mechanism and the Randomized Response mechanism is shown in Figure 3. The privacy parameters ϵ are set to $\{0.1, 0.2, 0.3, 0.9\}$, respectively, and we set the transmission rate p to 2 KB/s and 5 KB/s. We can see the following: (1) As the number of iterations increases, the regret value of the approach gradually increases. The deviation between the trained strategy performance and the optimal strategy performance gradually accumulates, and the regret value can be quantitatively estimated. (2) As the transmission rate of the wireless communication increases, the regret value generally shows a slight decreasing trend. For the same transmission rate of the wireless communication, the regret values under different privacy parameters are different. (3) For the same transmission rate of the wireless communication, the regret value of the approach is greatly affected by the privacy parameter. (4) Compared with the approach based on the Gaussian perturbation

mechanism, the approach based on the Random Response perturbation mechanism has a smaller regret value.

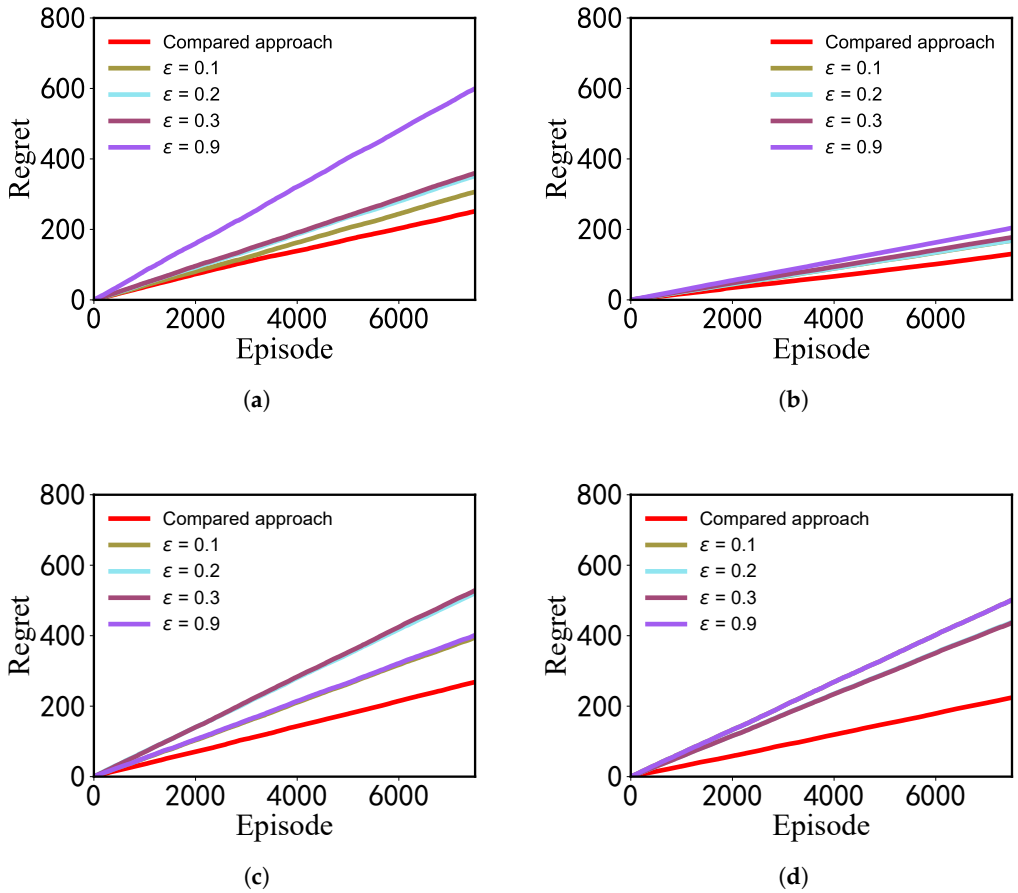


Figure 3. (**a**) $p = 2$ KB/s (Gaussian perturbation mechanism). (**b**) $p = 5$ KB/s (Gaussian perturbation mechanism). (**c**) $p = 2$ KB/s (Randomized Response mechanism). (**d**) $p = 5$ KB/s (Randomized Response mechanism).

6.2.3. Comparison with Non-Private RL Baseline

To explicitly evaluate the performance loss imposed by the LDP perturbation mechanism in offloading policy learning, we compared our approach with a non-private reinforcement learning baseline [39] (the red "Compared approach" curve in Figures 2 and 3). We adapted the states, actions in the literature [39].

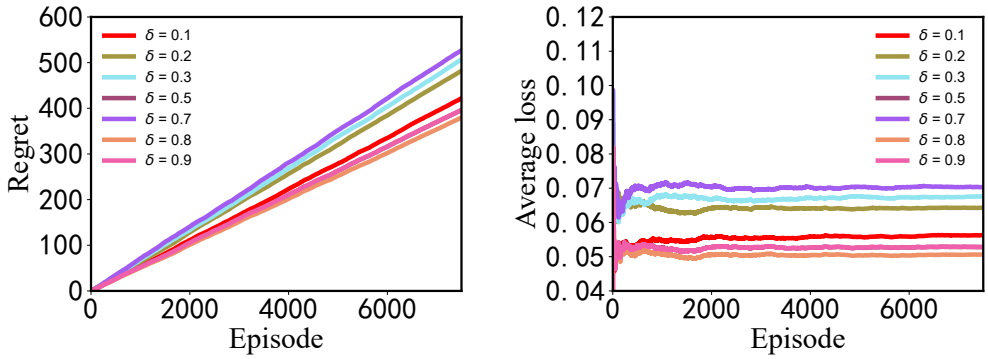
Figure 2 show that the non-private baseline achieves the lowest average loss and the fastest convergence, as expected. Our LDP perturbation approach experiences a slight increase in loss and a slight slowdown in convergence, especially under strict privacy budgets. Figure 3 shows that the non-private baseline maintains the lowest regret throughout training. The regret of our LDP perturbation approach increases with increasing privacy requirements, but remains within an acceptable range.

Adding LDP perturbation results in a controllable but quantifiable performance loss, which is the inherent cost of achieving strict privacy guarantees. This loss can be calculated based on our theoretical analysis and adjusted using the privacy budget ϵ, making our approach practical even in environments with varying privacy requirements.

6.2.4. The Impact of Different Discount Factors on the Approach

The impact of different discount factors based on the Gaussian perturbation mechanism on the approach is shown in Figure 4. The discount factor δ is set to $\{0.1, 0.2, 0.3, 0.5, 0.7, 0.8, 0.9\}$, the task size I_m is set to 100 KB, and the transmission rate p

is set to 2 KB/s. The following can be seen from the figure: (1) With the increase of the number of iterations, the regret value of the approach gradually increases under different discount factors, but different discount factors bring different regret values. Among them, when $\delta = 0.3$ the regret value is the smallest , while the regret value is the largest when $\delta = 0.9$. (2) For different discount factors, when the number of iterations is less than 2000, the average performance loss fluctuates greatly. When the number of iterations reaches 2000, the average performance loss of the approach reaches stability. Corresponding to the regret value, the average performance loss is the smallest when $\delta = 0.3$, and the average performance loss is the largest when $\delta = 0.9$.



(**a**) (**b**)

Figure 4. (**a**) Regret. (**b**) Average loss.

6.2.5. Impact of Privacy Mechanisms on Network Performance

While the LDP perturbation mechanism proposed in this paper effectively protects the privacy of drone offloading preferences, it also affects network performance in three aspects:

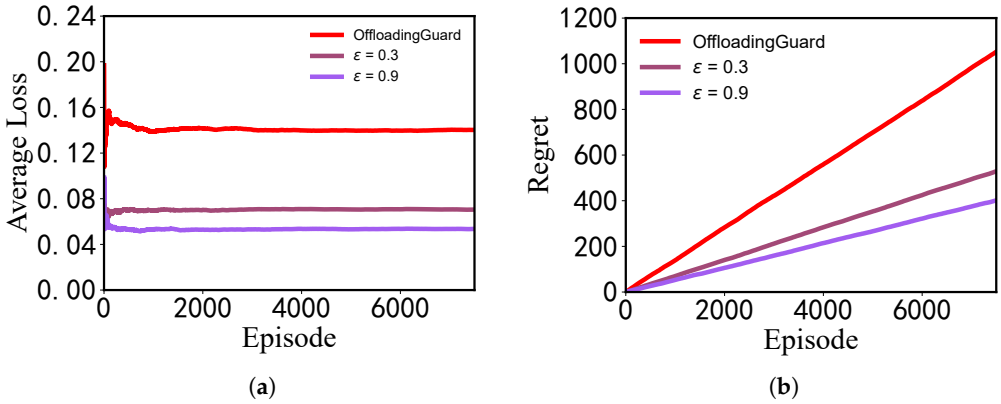
1. Communication overhead: Since each drone must locally perturb its trajectory before uploading, this adds a small amount of computational and communication overhead. However, experiments show that this overhead is acceptable.
2. Convergence speed: Privacy noise slows down the convergence of policy learning, especially when ϵ is small, as shown in Figure 2.
3. Policy performance loss: The addition of LDP noise perturbs the estimated state–action frequency, thus affecting the learned offloading policy.

However, through theoretical analysis in Section 5, we establish a theoretical regret bound $\mathcal{O}(\sqrt{M}/\epsilon)$, which not only provides strict privacy guarantees but also allows for a tunable performance trade-off, making it suitable for practical drone applications.

6.2.6. The Comparison with Other RL Baseline

We compared our approach with OffloadingGuard [33]. In the experiment, we selected the Random Response perturbation mechanism in our approach, and set the privacy parameter ϵ to $\{0.3, 0.9\}$ and the task size to $I_m = 100$ KB. A comparison of the convergence performance of the two methods is shown in Figure 5.

The experiment shows the following: (1) Our approach has a lower average loss than OffloadingGuard and converges faster than OffloadingGuard as the number of iterations increases. (2) The regret values of both methods increase with increasing iterations, but our approach has a smaller regret value than OffloadingGuard, indicating that the introduction of the privacy mechanism in our approach reduces the performance loss of the offloading strategy. Our approach is superior to OffloadingGuard.



(**a**) (**b**)

Figure 5. (**a**) Average loss. (**b**) Regret.

Another major innovation of our approach is that it theoretically establishes a quantitative relationship between the introduced privacy perturbation mechanism and the performance loss of the offloading strategy, reducing unnecessary resource waste.

In practical applications, it is necessary to balance privacy protection and algorithm performance. Our experimental results verify the effectiveness and robustness of the approach in different wireless communication link conditions and with different privacy parameters.

7. Discussion

7.1. Partially Offloading

Our research focuses on a full offloading model, where computing tasks are performed entirely locally on the drone or entirely by edge computing servers. However, partially offloading computing tasks to edge servers, where both local and edge servers perform the computation, is a more efficient and versatile approach in MEC [40]. In this section, we discuss the feasibility of extending our approach to support partial offloading.

Extending our approach to partial offloading involves the following changes: (1) Action space: The action space is generalized from a binary action space $a_m(t) \in 0, 1$ to a continuous fraction. For example, the action can be redefined as $a_m(t) = \rho \in [0, 1]$, where ρ represents the proportion of the task offloaded to the edge server, and the remainder is executed locally. (2) Cost model: The time delay and energy cost functions need to be restructured. The total time delay of a task is the maximum of the local computation time delay and the offload delay. The energy consumption is the sum of the local computation energy and the offload transmission energy. (3) State space: The state space contains more detailed information about the current computational task load of the drone and edge server.

The core privacy protection mechanisms of our approach remain highly applicable: (1) LDP perturbation mechanism: The LDP perturbation mechanism (Algorithms 1 and 2) operates on the frequencies of state–action pairs in the decision trajectory. When expanding the continuous action space, the perturbation mechanism remains essentially unchanged. In theory, the (ϵ, δ)-LDP privacy guarantees would still hold. (2) Offloading policy learning mechanism: The offloading policy learning mechanism (Algorithm 3) needs to be adapted to the expanded action space, possibly using deep RL techniques.

7.2. Malicious Servers

Our approach protects the privacy of the offloading policy training in an RL algorithm based on honest-but-curious servers. However, malicious servers may still interfere with the algorithm's execution process or steal sensitive data. To address these malicious servers, we can deploy the RL algorithm training process in a trusted execution environment (TEE).

TEEs implement secure computing through memory isolation within an independent processing environment, offering hardware-based security and integrity protection. Currently, there are many mature TEE solutions, such as ARM Trustzone and Intel SGX. The existing TEE ecosystem is relatively mature and supports deployment on multiple CPU architectures. Although our solution does not directly design protection mechanisms against malicious servers, by leveraging TEEs and existing mature technologies we can deploy our solution on potentially malicious servers, maintaining the same functionality and performance without changing the core structure of our algorithm, effectively extending our approach to malicious-server scenarios [41].

7.3. Partical Use

The proposed UAV-centric privacy-preserving computation offloading scheme offers significant practical advantages in scenarios where UAVs handle privacy-sensitive tasks. It effectively addresses a critical vulnerability: if an honest-but-curious server discloses offloading strategies, attackers controlling edge computing servers (ECSs) could exploit this information to lure UAVs into offloading sensitive tasks to compromised ECSs. In public safety surveillance, this approach prevents the exposure of UAV patrol routes and deters the induced offloading of high-residency video footage. In precision agriculture, it safeguards crop data offloading policies and prevents leaks of yield-related information. For infrastructure inspection, it thwarts the disclosure of defect-data offloading preferences and blocks unauthorized transfers of vulnerability logs. In emergency relay scenarios, it avoids leakage of rescue-zone offloading strategies and thereby prevents voice data breaches. By incorporating local differential privacy (LDP) mechanisms, the proposed method perturbs the frequencies of state–action pairs, thereby securing offloading strategies and fundamentally eliminating the risk of such malicious induction. This provides essential protection for the future large-scale deployment of UAV-assisted edge computing networks.

8. Conclusions

In conclusion, we propose a UAV-centric privacy-preserving offloading approach via RL. Differing from existing works, the proposed approach adds LDP noise to randomize the frequency of each state–action pair in the decision trajectories. We provide two perturbation mechanisms, the Gaussian perturbation mechanism and the Random Response mechanism, and prove that they each achieve the (ϵ, δ')-LDP and $(\epsilon, 0)$-LDP guarantee, respectively. Furthermore, we theoretically derive the regret bound of this approach as $\mathcal{O}(\sqrt{M}/\epsilon)$, and establish a quantitative relationship between the privacy budget and the performance loss of the offloading strategy before training the offloading strategy. Experiments verify the convergence and efficiency of our approach under different task sizes, transmission rates, and privacy parameters. Theoretical analysis and experimental results demonstrate that our approach provides a feasible and practical solution for protecting drone privacy in real-world MEC applications, enabling system designers to effectively balance privacy and performance based on specific mission requirements.

However, our work has some limitations. Our current system model assumes complete task offloading, but partial offloading is often more efficient and applicable. Furthermore, exploring the combination of other privacy-preserving techniques with our offloading approach and discussing better privacy–utility trade-offs is another promising direction.

Author Contributions: Conceptualization, D.W.; methodology, C.G.; software, C.G.; validation, D.W. and C.G.; formal analysis, C.G.; investigation, D.W.; writing—original draft preparation, C.G.; writing—review and editing, D.W.; visualization, K.L. and W.L.; supervision, D.W.; funding acquisition, D.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 62302363), the Major Research Plan of the National Natural Science Foundation of China (Grant No. 92267204), the Fundamental Research Funds for the Central Universities (Project No. ZYTS25072), and the Open Foundation of Key Laboratory of Cyberspace Security, Ministry of Education of China (No. KLCS20240404).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhou, Y.; Liu, L.; Wang, L.; Hui, N.; Cui, X.; Wu, J.; Peng, Y.; Qi, Y.; Xing, C. Service-aware 6G: An intelligent and open network based on the convergence of communication, computing and caching. *Digit. Commun. Netw.* **2020**, *6*, 253–260. [CrossRef]
2. Javanmardi, S.; Nascita, A.; Pescapè, A.; Merlino, G.; Scarpa, M. An integration perspective of security, privacy, and resource efficiency in IoT-Fog networks: A comprehensive survey. *Comput. Netw.* **2025**, *270* , 111470. [CrossRef]
3. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. Mobile Edge Computing: Survey and Research Outlook. *arXiv* **2017**, arXiv:1701.01090.
4. Cai, Q.; Zhou, Y.; Liu, L.; Qi, Y.; Pan, Z.; Zhang, H. Collaboration of Heterogeneous Edge Computing Paradigms: How to Fill the Gap Between Theory and Practice. *IEEE Wirel. Commun.* **2024**, *31*, 110–117. [CrossRef]
5. Zhao, M.; Zhang, R.; He, Z.; Li, K. Joint Optimization of Trajectory, Offloading, Caching, and Migration for UAV-Assisted MEC. *IEEE Trans. Mob. Comput.* **2025**, *24*, 1981–1998. [CrossRef]
6. Waqar, N.; Hassan, S.A.; Mahmood, A.; Dev, K.; Do, D.; Gidlund, M. Computation Offloading and Resource Allocation in MEC-Enabled Integrated Aerial-Terrestrial Vehicular Networks: A Reinforcement Learning Approach. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 21478–21491. [CrossRef]
7. Saeedi, I.D.I.; Al-Qurabat, A.K.M. A comprehensive review of computation offloading in UAV-assisted mobile edge computing for IoT applications. *Phys. Commun.* **2025**, 102810. [CrossRef]
8. Lan, W.; Chen, K.; Li, Y.; Cao, J.; Sahni, Y. Deep Reinforcement Learning for Privacy-Preserving Task Offloading in Integrated Satellite-Terrestrial Networks. *IEEE Trans. Mob. Comput.* **2024**, *23*, 9678–9691. [CrossRef]
9. Li, J.; Ou, W.; Ouyang, B.; Ye, S.; Zeng, L.; Chen, L.; Chen, X. Revisiting Location Privacy in MEC-Enabled Computation Offloading. *IEEE Trans. Inf. Forensics Secur.* **2025**, *20*, 4396–4407. [CrossRef]
10. Wei, D.; Xi, N.; Ma, X.; Shojafar, M.; Kumari, S.; Ma, J. Personalized Privacy-Aware Task Offloading for Edge-Cloud-Assisted Industrial Internet of Things in Automated Manufacturing. *IEEE Trans. Ind. Inform.* **2022**, *18*, 7935–7945. [CrossRef]
11. Ding, Y.; Li, K.; Liu, C.; Li, K. A Potential Game Theoretic Approach to Computation Offloading Strategy Optimization in End-Edge-Cloud Computing. *IEEE Trans. Parallel Distributed Syst.* **2022**, *33*, 1503–1519. [CrossRef]
12. He, Z.; Xu, Y.; Zhao, M.; Zhou, W.; Li, K. Priority-Based Offloading Optimization in Cloud-Edge Collaborative Computing. *IEEE Trans. Serv. Comput.* **2023**, *16*, 3906–3919. [CrossRef]
13. Zhao, W.; Shi, K.; Liu, Z.; Wu, X.; Zheng, X.; Wei, L.; Kato, N. DRL Connects Lyapunov in Delay and Stability Optimization for Offloading Proactive Sensing Tasks of RSUs. *IEEE Trans. Mob. Comput.* **2024**, *23*, 7969–7982. [CrossRef]
14. Zhang, Y.; Kuang, Z.; Feng, Y.; Hou, F. Task Offloading and Trajectory Optimization for Secure Communications in Dynamic User Multi-UAV MEC Systems. *IEEE Trans. Mob. Comput.* **2024**, *23*, 14427–14440. [CrossRef]
15. Liu, Z.; Li, Z.; Gong, Y.; Wu, Y. RIS-Aided Cooperative Mobile Edge Computing: Computation Efficiency Maximization via Joint Uplink and Downlink Resource Allocation. *IEEE Trans. Wirel. Commun.* **2024**, *23*, 11535–11550. [CrossRef]
16. Yang, Y.; Gong, Y.; Wu, Y. Intelligent-Reflecting-Surface-Aided Mobile Edge Computing With Binary Offloading: Energy Minimization for IoT Devices. *IEEE Internet Things J.* **2022**, *9*, 12973–12983. [CrossRef]
17. Li, K.; Wang, X.; He, Q.; Yang, M.; Huang, M.; Dustdar, S. Task Computation Offloading for Multi-Access Edge Computing via Attention Communication Deep Reinforcement Learning. *IEEE Trans. Serv. Comput.* **2023**, *16*, 2985–2999. [CrossRef]
18. Huang, L.; Bi, S.; Zhang, Y.A. Deep Reinforcement Learning for Online Computation Offloading in Wireless Powered Mobile-Edge Computing Networks. *IEEE Trans. Mob. Comput.* **2020**, *19*, 2581–2593. [CrossRef]
19. Lei, L.; Xu, H.; Xiong, X.; Zheng, K.; Xiang, W.; Wang, X. Multiuser Resource Control With Deep Reinforcement Learning in IoT Edge Computing. *IEEE Internet Things J.* **2019**, *6*, 10119–10133. [CrossRef]
20. Wang, J.; Hu, J.; Min, G.; Zomaya, A.Y.; Georgalas, N. Fast Adaptive Task Offloading in Edge Computing Based on Meta Reinforcement Learning. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 242–253. [CrossRef]
21. Feng, T.; Wang, B.; Zhao, H.; Zhang, T.; Tang, J.; Wang, Z. Task Distribution Offloading Algorithm Based on DQN for Sustainable Vehicle Edge Network. In Proceedings of the 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), Tokyo, Japan, 28 June–2 July 2021 ; IEEE: New York, NY, USA, 2021; pp. 430–436.

22. Liu, C.; Tang, F.; Hu, Y.; Li, K.; Tang, Z.; Li, K. Distributed Task Migration Optimization in MEC by Extending Multi-Agent Deep Reinforcement Learning Approach. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1603–1614. [CrossRef]
23. Wei, D.; Zhang, J.; Shojafar, M.; Kumari, S.; Xi, N.; Ma, J. Privacy-Aware Multiagent Deep Reinforcement Learning for Task Offloading in VANET. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 13108–13122. [CrossRef]
24. Wang, Z.; Sun, Y.; Liu, D.; Hu, J.; Pang, X.; Hu, Y.; Ren, K. Location Privacy-Aware Task Offloading in Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2024**, *23*, 2269–2283. [CrossRef]
25. Gan, S.; Siew, M.; Xu, C.; Quek, T.Q.S. Differentially Private Deep Q-Learning for Pattern Privacy Preservation in MEC Offloading. In Proceedings of the ICC 2023—IEEE International Conference on Communications, Rome, Italy, 28 May–1 June 2023; IEEE: New York, NY, USA, 2023; pp. 3578–3583.
26. Cui, X.; Tian, Y.; Zhang, X.; Lin, H.; Li, M. A Lightweight Certificateless Edge-Assisted Encryption for IoT Devices: Enhancing Security and Performance. *IEEE Internet Things J.* **2025**, *12*, 2930–2942. [CrossRef]
27. Li, Z.; Hu, H.; Huang, B.; Chen, J.; Li, C.; Hu, H.; Huang, L. Security and performance-aware resource allocation for enterprise multimedia in mobile edge computing. *Multimed. Tools Appl.* **2020**, *79*, 10751–10780. [CrossRef]
28. Xu, X.; He, C.; Xu, Z.; Qi, L.; Wan, S.; Bhuiyan, M.Z.A. Joint Optimization of Offloading Utility and Privacy for Edge Computing Enabled IoT. *IEEE Internet Things J.* **2020**, *7*, 2622–2629. [CrossRef]
29. Min, M.; Xiao, L.; Chen, Y.; Cheng, P.; Wu, D.; Zhuang, W. Learning-Based Computation Offloading for IoT Devices With Energy Harvesting. *IEEE Trans. Veh. Technol.* **2019**, *68*, 1930–1941. [CrossRef]
30. Ji, T.; Luo, C.; Yu, L.; Wang, Q.; Chen, S.; Thapa, A.; Li, P. Energy-Efficient Computation Offloading in Mobile Edge Computing Systems With Uncertainties. *IEEE Trans. Wirel. Commun.* **2022**, *21*, 5717–5729. [CrossRef]
31. Wang, K.; Chen, C.; Tie, Z.; Shojafar, M.; Kumar, S.; Kumari, S. Forward Privacy Preservation in IoT-Enabled Healthcare Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 1991–1999. [CrossRef]
32. Nguyen, D.C.; Pathirana, P.N.; Ding, M.; Seneviratne, A. Privacy-Preserved Task Offloading in Mobile Blockchain with Deep Reinforcement Learning. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 2536–2549. [CrossRef]
33. Pang, X.; Wang, Z.; Li, J.; Zhou, R.; Ren, J.; Li, Z. Towards Online Privacy-preserving Computation Offloading in Mobile Edge Computing. In Proceedings of the IEEE INFOCOM 2022-IEEE Conference on Computer Communications, Virtual, 2–5 May 2022; IEEE: New York, NY, USA, 2022; pp. 1179–1188.
34. You, F.; Yuan, X.; Ni, W.; Jamalipour, A. Learning-Based Privacy-Preserving Computation Offloading in Multi-Access Edge Computing. In Proceedings of the GLOBECOM 2023-2023 IEEE Global Communications Conference, Kuala Lumpur, Malaysia, 4–8 December 2023; IEEE: New York, NY, USA, 2023; pp. 922–927.
35. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018.
36. Garcelon, E.; Perchet, V.; Pike-Burke, C.; Pirotta, M. Local Differential Privacy for Regret Minimization in Reinforcement Learning. In Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Online, 6–14 December 2021; pp. 10561–10573.
37. Bubeck, S.; Cesa-Bianchi, N. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Found. Trends Mach. Learn.* **2012**, *5*, 1–122. [CrossRef]
38. Tao, M.; Li, X.; Feng, J.; Lan, D.; Du, J.; Wu, C. Multi-Agent Cooperation for Computing Power Scheduling in UAVs Empowered Aerial Computing Systems. *IEEE J. Sel. Areas Commun.* **2024**, *42*, 3521–3535. [CrossRef]
39. Ren, Y.; Sun, Y.; Peng, M. Deep Reinforcement Learning Based Computation Offloading in Fog Enabled Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4978–4987. [CrossRef]
40. Mao, Y.; Zhang, J.; Letaief, K.B. Dynamic Computation Offloading for Mobile-Edge Computing With Energy Harvesting Devices. *IEEE J. Sel. Areas Commun.* **2016**, *34*, 3590–3605. [CrossRef]
41. Wang, W.; Ji, H.; He, P.; Zhang, Y.; Wu, Y.; Zhang, Y. WAVEN: WebAssembly Memory Virtualization for Enclaves. In Proceedings of the 32nd Annual Network and Distributed System Security Symposium, NDSS 2025, San Diego, CA, USA, 24–28 February 2025.