Location Privacy-Aware Task Offloading in Mobile Edge Computing

Zhibo Wang, *Senior Member, IEEE,* Yunan Sun, Defang Liu, Jiahui Hu, Xiaoyi Pang, Yuke Hu, and Kui Ren, *Fellow, IEEE*

Abstract—In mobile edge computing (MEC), users can offload tasks to nearby MEC servers to reduce computation cost. Considering that the size of offloaded tasks could disclose user location information, several location privacy-preserving task offloading mechanisms have been proposed under the single-server scenario. However, to the best of our knowledge, none of them could provide a strict privacy protection guarantee or be applicable to the multi-server scenario where the user's location can be inferred more accurately if servers collude with each other. In this paper, we propose a novel location privacy-aware task offloading framework (LPA-Offload) for both single-server and multi-server scenarios, which provides strict and provable location privacy protection while achieving efficient task offloading. Specifically, we propose a location perturbation mechanism that allows each user to perturb its real location within a rational perturbation region and provides a differential privacy guarantee. To make a satisfactory offloading strategy, we propose a perturbation region determination mechanism and an offloading strategy generation mechanism that adaptively select a proper perturbation region according to the customized privacy factor, and then generate an optimal offloading strategy based on the perturbed location within the decided region. The determination of the perturbation region could achieve personalized privacy requirements while reducing computation cost. LPA-Offload is proved to satisfy (ϵ, δ)-differential privacy, and the experiments demonstrate the effectiveness of our framework.

Index Terms—location privacy, differential privacy, mobile edge computing, task offloading

1 Introduction

With the explosive growth of mobile devices, applications become more diversified and complicated, such as face recognition and augmented reality [1], [2]. These applications require intensive computation resources and high energy consumption, which becomes a challenge for mobile devices due to their limited computing resources and battery life. Recently, mobile edge computing (MEC), as a new computing paradigm, allows users to offload computation tasks to MEC servers through wireless channels, which provides more computation resources at the network edge to mobile users. Thus, the computing delay and energy consumption could be reduced while the rich computation capacity of MEC servers helps process resource-intensive and delay-sensitive tasks [3], [4], [5].

However, recent works [6], [7], [8], [9] pointed out that it is possible for untrusted MEC servers to infer user location information from the process of task offloading. Generally speaking, the user would offload tasks according to the wireless channel conditions, which are negatively correlated to distance and determine the computation cost. To reduce computing cost, a mobile user would like to offload more tasks to a MEC server when it is close to the server with good wireless channel conditions, but tends to process more locally when the wireless channel conditions are poor. Therefore, an untrusted MEC server could infer the conditions of wireless channels by monitoring the size of offloaded tasks, and then infer the user's location information (i.e., a larger size of offloaded tasks means a shorter distance from the user to the server).

With the location information, attackers can infer more sensitive information about users, such as the users' identity, social relationships, health status, etc [10], [11]. To avoid location privacy leakage, several works have proposed privacy-preserved task offloading mechanisms [6], [12], [13], [14]. They set a privacy level as an indicator of the optimization goal, and find the lowest-cost offloading strategy under the predefined privacy constraints. Nevertheless, these works provided the user with the same level of privacy guarantee at each time, which results in a constant difference between the size of generated tasks and offloaded tasks. Therefore, an untrusted server could still be able to infer the size of the user's generated tasks and the conditions of wireless channels in a long-term process, and then explore the user's location information. In addition, existing works could only be applied to the single-server scenario. When a mobile user is covered by multiple MEC servers, these servers may collude with each other and jointly infer the location of the user to a certain precision [6]. For example, when a user is within the coverage of three servers, once these servers share the distances away from the user, the exact location of the user can be calculated. Thus, it is necessary to design a novel location privacy-aware task

- *Zhibo Wang is with Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, 430072, P.R. China, and also with the School of Cyber Science and Technology, Zhejiang University, Hangzhou 310027, P.R. China. E-mail: zhibowang@zju.edu.cn.*
- *Yunan Sun, Defang Liu, and Xiaoyi Pang are with Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, 430072, P.R. China. E-mail: {yunan.sun, defangliu, xypang}@whu.edu.cn.*
- *Jiahui Hu, Yuke Hu and Kui Ren are with the School of Cyber Science and Technology, Zhejiang University, Hangzhou 310027, China. E-mail: {jiahuihu, yukehu, kuiren}@zju.edu.cn.*

offloading approach to resolve the issues above and provide a strong and provable location privacy guarantee.

In this paper, we focus on the location privacy issue during task offloading in MEC and aim to provide strict location privacy protection according to the personalized privacy requirement while reducing computation cost. We assume that MEC servers will honestly process tasks according to the protocol, but may collude and learn users' location information. To achieve the goal, we are facing the following challenges. Firstly, untrusted MEC servers could infer the user's location information if the mobile user offloads tasks based on the real location, which could be calculated more accurately if MEC servers collude. Thus, it's a big challenge to protect the user's location privacy from being inferred for both single-server and multi-server scenarios in a long-term process. Secondly, mobile users may have different location privacy requirements and wireless channel conditions, which could influence the efficiency of task offloading. It is necessary to adaptively make a satisfactory offloading strategy according to the customized privacy factor to balance computation cost and location privacy.

To address these challenges, we propose a novel location privacy-aware task offloading framework (LPA-Offload) to realize efficient task offloading in MEC while providing a strict location privacy guarantee for each user. In the single-server scenario, the user would perturb the real distance between the MEC server and itself, and then offload tasks based on that perturbed distance. In the multi-server scenario, to prevent location leakage from the multi-server collusion, the user needs to perturb its real coordinate and make the offloading strategy based on the perturbed coordinate. To be specific, we propose a location perturbation mechanism that utilizes differential privacy method to perturb the mobile user's real location within a rational perturbation region to protect the location privacy. Since the determination of the perturbation region could influence the selection of perturbed location and further affect the level of location privacy leakage and computation cost of task offloading, we propose a perturbation region determination mechanism to adaptively select a proper perturbation region that has the maximum expected utility according to the personalized location privacy requirements. After the perturbation region is determined, we could obtain a privacy-aware offloading strategy through the offloading strategy generation mechanism. However, in the multi-server scenario, the explosive growth of the possible offloading strategies makes the offloading strategy generation NP-hard, and we utilize genetic algorithm (GA) to resolve this issue.

The main contributions of this paper are summarized as follows:

- We propose a location privacy-aware task offloading framework (LPA-Offload) based on differential privacy to protect users' location privacy against untrusted MEC servers for both single-server and multi-server scenarios. To the best of our knowledge, this is the first work to protect the location privacy of the mobile user in the multi-server scenario with a strong theoretical guarantee in task offloading.
- We propose a perturbation region determination mechanism and an offloading strategy generation mechanism to adaptively generate a privacy-aware task offloading strategy according to the personalized privacy requirement, which could achieve the trade-off between computation cost and location privacy.
- We prove that LPA-Offload satisfies (ϵ, δ)-differential privacy. The experiments demonstrate that LPA-Offload has almost the same computation cost as the state-of-art offloading method without location protection, and is not as vulnerable to inference attacks as the existing location privacy-preserving offloading method.

The remainder of this paper is organized as follows. In Section 2, a review of related works in literature is presented. In Section 3, we describe the system model and problem formulation. We introduce the detailed designs of LPA-Offload in Section 4 and give the theoretical analysis in Section 5. Experiment results are presented in Section 6. Finally, the paper is concluded in Section 8.

2 Related Work

In this section, we first introduce the research about task offloading mechanisms without privacy concerns in MEC, and then discuss the related works of privacy-preserving offloading.

2.1 Task Offloading Mechanisms

Generally speaking, computation cost in task offloading consists of computation delay and energy consumption. Thus, the main goals of traditional task offloading include reducing computation delay, minimizing energy consumption, and balancing the delay and energy.

In [15], [16], [17], [18], [19], [20], [21], the authors focus on minimizing computation delay according to computing power and wireless channel conditions. Besides, Liu et al. [5] developed an effective offloading strategy to minimize the time spent under power limit based on the queueing state of the task buffer, the execution state of the local processing unit, and the state of the transmission unit. Chen et al. [1] studied the multi-user computation offloading problem for mobile-edge cloud computing in a multi-channel wireless interference environment, which proved to be NP-hard. Then, they adopt a game theoretic approach for achieving efficient computation offloading in a distributed manner.

Several works studied the offloading mechanisms that minimize energy consumption under delay limit [22], [23]. Kamoun et al. [4] proposed both online and pre-calculated offline strategies that take into account both computation and radio resources. The proposed strategies minimize the average energy consumption while satisfying the predefined delay constraints. Jiang et al. [24] studied the multi-user offloading problem for mobile edge computing (MEC) in a multi-server environment. They formulated the problem of minimizing energy consumption as a multidimensional multiple knapsack (MMKP) problem and proposed a neural network architecture called Multi-Pointer networks (Mptr-Net) to solve it.

In [3], [25], the authors considered both computation delay and energy consumption, and work on balancing the relationship between them. Research [26] transformed the balance of energy consumption, data backlogs, and time consumption into the knapsack problem, which is solved by

Lyapunov optimization technique to ensure the trade-off of energy, delay, and data throughput. Nouri et al. [27] defined the total cost of the network as a weighted combination of the consumed energy and delay for all users of the network, and then formulated the task offloading problem as a mixed integer non-linear programming (MINLP) problem. They utilized the successive convex approximation (SCA) algorithm to find the optimal solution for this problem.

2.2 Privacy-preserving Offloading Mechanism

In the MEC system, user privacy could be disclosed in the process of service requests. He et al. [28] found that a cyber eavesdropper can localize the user up to one MEC coverage area by observing service trajectories. They proposed a suite of chaff control strategies for the user to reduce the eavesdropper's tracking accuracy. Gao et al. [29] found that users have preferences and regularity when they request services, so that the attackers can collect the historical service request information to obtain real-time user locations. Therefore, they introduced deep reinforcement learning to the privacy model in order to reduce the risk of location privacy leakage. The privacy leakage caused by service requests has been widely researched and well solved, while our paper focuses on the privacy issues in task offloading which are also important but lack effective protection. Recent works have pointed out that in task offloading, mobile users' privacy information could be disclosed due to data interaction or wireless transmission. Thus, some privacy-preserving task offloading mechanisms have been proposed to protect mobile users' privacy against untrusted MEC servers.

Mobile users may offload tasks containing sensitive data to MEC servers, so the attackers could access user privacy through data interaction. Several works proposed privacy-aware offloading mechanisms to solve this privacy issue. To avoid privacy leakage, Xu et al. [30] divided the computing tasks into several types and enhanced the uncertainty of the computing tasks to increase privacy entropy. Then, an improved Strength Pareto Evolutionary Algorithm (SPEA2) is leveraged to optimize the average time consumption and average privacy entropy jointly. Xu et al. [31] selected the utility-aware offloading strategy by using NSGA-III to achieve the optimization of utilization and time cost. Then the utility and the privacy entropy were taken into consideration as two main metrics to be optimized at the same time. Moreover, He et al. [32] developed a privacy-preserving and cost-efficient (PEACE) task offloading scheme based on the general framework of Lyapunov optimization, to avoid user identification through specific tasks. PEACE task offloading schemes can be applied to both the scenarios of non-colluding and colluding adversaries. Note that, the privacy issue caused by data interaction in task offloading is not within the scope of this paper.

The conditions of wireless channels are related to the distance between the user and the MEC server, which could affect the size of offloaded tasks. Hence, some works found the location information of mobile users could be inferred from the size of offloaded tasks and discussed the causes and solutions of this issue [7], [33]. He et al. [6] proposed a constrained Markov decision process (CMDP) based privacy-aware task offloading scheduling algorithm to minimize the delay and energy consumption while keeping a pre-specified level of privacy. Nguyen et al. [12] regarded task offloading and location privacy preservation as a joint optimization problem and used the reinforcement learning (RL) method to obtain the best offloading scheme while enhancing location privacy. Li et al. [13] formulated the joint optimization problem as a contextual multi-armed bandit (CMAB) problem and proposed a privacy-aware online task offloading (PAOTO) algorithm based on the transformed Thompson Sampling (TS) architecture, which can obtain a suboptimal solution to minimize the delay and energy consumption while protecting location privacy.

Nevertheless, all these mechanisms utilized a threshold to distinguish "good" and "poor" wireless channel conditions, and defined the location privacy disclosure under these two discrete conditions, which however is not practical. Moreover, these works just predefined a static privacy parameter as an indicator of the optimization goal, leading to a situation where the difference between the size of user-generated tasks and user-offloaded tasks is always a constant. Hence, malicious MEC servers are still able to infer the user's location information by monitoring the size of offloaded tasks in a long-term process. Thus, these mechanisms could not provide a strict location privacy guarantee. Meanwhile, the multi-server collusion, which is a more harmful scenario, has not been considered in existing works.

A more recent work, OffloadingGuard [34], also studied the privacy issues in task offloading, but has some differences with ours. Firstly, OffloadingGuard adds noise on the offloading ratio to protect privacy like usage patterns, but cannot measure the level of location privacy directly. However, our work focuses more on location privacy, which is protected by perturbing the real location to a false location. Secondly, OffloadingGuard is only applicable to MEC systems with only one server while our framework can also achieve location privacy-preserving task offloading in the multi-server scenario.

3 System Model and Problem Formulation

In this section, we first briefly introduce the knowledge of differential privacy, then present the system model of task offloading in MEC, and finally describe the problem to be solved in this paper. Table 1 summarizes the key parameter notations in our paper.

3.1 Preliminaries

Differential privacy is used to prevent individual records in a dataset from being identified, which can also be employed to protect location privacy. The definition of differential privacy in [35], [36] is as follows.

Definition 1 (ϵ-Differential Privacy [35]). Given a randomized function M, for any dataset D and D' differing on at most one element, and any subset $O \subseteq Range(M)$, if the algorithm M satisfies:

$$Pr[M(D) \in O] \leq e^{\epsilon} \cdot Pr[M(D') \in O] \tag{1}$$

then the function M satisfies **ϵ-Differential Privacy** where ϵ represents the privacy budget. The closer ϵ is to 0, the better privacy protection will be.

TABLE 1
Key Notations in Our Model.

Notation	Definition
t_i	The i-th task
v_i	The size of task t_i
s_k	The k-th server
χ	Offloading strategy
x_{ik}	Whether task t_i is assigned to server k.
U	The utility of the system
C, PL	The computation cost and privacy leakage level of the system
D, E	The total computing delay and energy consumption
D_{il}, E_{il}	The computing delay and energy consumption of completing task t_i locally
D_{ik}, E_{ik}	The computing delay and energy consumption for the edge server s_k to process task t_i
R_k	The data transmission rate of sever s_k
l, l^*	The real distance and fake distance
Λ, Λ^*	The real location and the perturbed location
$[l_1, l_2]$	The perturbation region in single-server
$R, [\theta_1, \theta_2]$	The perturbation radius and angles in multi-server

Definition 2 ***((ϵ, δ)-Differential Privacy [36]).*** Given a randomized function M, for any dataset D and D' differing on at most one element, and any subset $O \subseteq Range(M)$, if the algorithm M satisfies:

$$Pr[M(D) \in O] \leq e^{\epsilon} \cdot Pr[M(D') \in O] + \delta \tag{2}$$

then the function M satisfies (ϵ, δ)**-Differential Privacy** where δ quantitatively represents the privacy loss. The closer δ approaches 0, the better privacy protection will be.

Definition 3 (Global Sensitivity [37]). For any function $f : \mathcal{D} \to \mathbb{R}^d$, the sensitivity of f w.r.t. $\mathcal{D}$ is

$$\Delta f = \max_{D, D' \in \mathcal{D}} ||f(D) - f(D')|| \tag{3}$$

where Δf also stands for the maximum influence of any element in the dataset on the result of the query function f.

A stochastic noise drawn from Laplace distribution is often used to achieve differential privacy [37]. The core of Laplace mechanism is adding a random noise to the query results.

Definition 4 (Laplace Mechanism [37]). Given a function $f : \mathcal{D} \to \mathbb{R}^d$ over domain D, the mechanism M satisfies ϵ-differential privacy, if

$$M(D) = f(D) + Laplace(\Delta f / \epsilon) \tag{4}$$

The probability density function of Laplace distribution is:

$$Pr[x] = \frac{1}{2\sigma} e^{-\frac{|x-\mu|}{\sigma}}, x \in (-\infty, +\infty) \tag{5}$$

where μ represents the location parameter of Laplace distribution, which is generally set to 0, and $\sigma = \Delta f / \epsilon$. Therefore, the noise level in Laplace mechanism will be adjusted according to the sensitivity of the data.

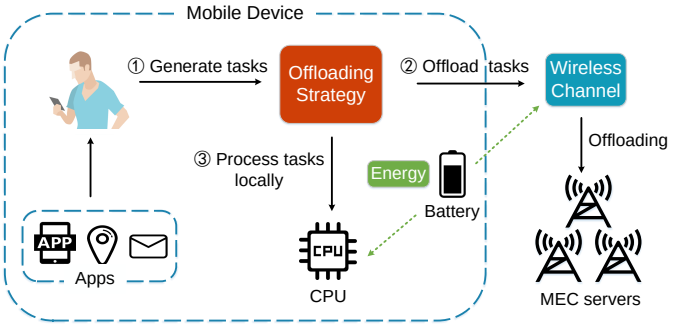


Fig. 1. Mobile edge computing with multiple servers.

3.2 System Model

We consider a general MEC system consisting of several MEC servers and a large number of mobile users with mobile devices. The offloading process between a user and multiple MEC servers is shown in Figure 1. The user could offload a part of the generated tasks to nearby MEC servers to reduce the computation cost and then process the remaining tasks locally. During the process, the user's offloading strategy determines whether or where to offload tasks, which depends on the wireless channel conditions between the user and the MEC servers. In this system, we assume that the MEC servers are honest-but-curious, which will honestly receive and process the tasks according to the protocol but would collude to learn the user's location information. The mobile user would collect the information like channel conditions in the system first, and then offload tasks according to the information. When the wireless channel condition between the mobile user and the edge server is good, the user tends to offload more tasks to the edge server to reduce the computation cost. On the contrary, when the channel condition is poor, the user tends to execute more tasks locally.

3.3 Problem Formulation

Let $T = \{t_1, t_2, \cdots, t_M\}$ denote M tasks, the size of which could be denoted as $V = \{v_1, v_2, \cdots, v_M\}$. Let $S = \{s_1, s_2, \cdots, s_N\}$ denote N edge servers, and we leverage an indicator x_{ik} to represent the allocation state of tasks [38], i.e.,

$$x_{ik} = \begin{cases} 1, & \text{if task } t_i \text{ is assigned to server } s_k \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $x_{ik} = 1$ indicates that task t_i is offloaded to edge server s_k, and $x_{ik} = 0$, otherwise. Hence, the offloading strategy can be denoted by matrix $\chi = \{x_{ik} | t_i \in T, s_k \in S\}$.

The computation cost consists of computing delay and energy consumption, so the total cost of the mobile user can be expressed as

$$C = (1 - \lambda) \cdot D + \lambda \cdot E \tag{7}$$

where D and E are the total computing delay and energy consumption, respectively. The λ is the weight of balancing the energy consumption and the computation delay [1], which can be set according to the user's preference and

the running application demands. The relationship between energy and delay is not the focus of our paper.

Due to the parallel computing of MEC servers and local device, the total delay D can be expressed by

$$D = \max\{D_l, D_1, \cdots, D_N\} \tag{8}$$

where l represents the tasks are processed locally, D_l denotes the corresponding computing delay, and $D_k(s_k \in S)$ represents the computing delay for edge server s_k to complete offloaded tasks. The total energy consumption E is given by:

$$E = E_l + \sum_{k=1}^{N} E_k \tag{9}$$

where E_l represents the energy consumption of processing tasks locally, and $E_k(s_k \in S)$ represents the energy consumption of tasks offloaded to edge server s_k. Hence, we can see that the computation cost could be separated into two categories: local execution and edge server execution.

Computation cost of local execution. When the wireless channel condition is poor, the mobile user tends to calculate the tasks locally. Let f_l denote the CPU-cycle frequency of the mobile device. According to the calculation in [1], [39], the local computation delay is computed as:

$$D_l = \sum_{i=1}^{M}(1 - \sum_{k=1}^{N} x_{ik})D_{il} = \sum_{i=1}^{M}(1 - \sum_{k=1}^{N} x_{ik})\frac{v_i\gamma}{f_l} \tag{10}$$

where D_{il} means the latency of completing task t_i locally, v_i represents the size of task t_i, γ is the computation intensity (in CPU cycles per bit), and $v_i\gamma$ represents the total number of CPU cycles of tasks t_i completed locally.

The energy consumption of completing tasks on the mobile device can be given by

$$E_l = \sum_{i=1}^{M}(1 - \sum_{k=1}^{N} x_{ik})E_{il} = \sum_{i=1}^{M}(1 - \sum_{k=1}^{N} x_{ik})\kappa f_l^2 v_i\gamma \tag{11}$$

where E_{il} represents the energy consumption of completing task t_i locally, and κ is the consumed energy per CPU cycle, which is determined by the chip structure of the mobile device.

Computation cost of edge server execution. When the wireless channel is in good condition, the mobile device transmits tasks to edge servers through the wireless channel, which can reduce the delay and energy consumption.

The computation delay of the edge server consists of two parts: the data transmission time and the task execution time on the edge server. Similar to [1], the size of the outcome is generally much smaller than that of input data, so we neglect the time overhead from edge server to mobile user. Thus, the computation delay for edge server s_k to complete offloaded tasks is defined as:

$$D_k = \sum_{i=1}^{M} x_{ik}D_{ik} = \sum_{i=1}^{M} x_{ik}(\frac{v_i}{R_k} + \frac{v_i\gamma}{f_k}) \tag{12}$$

where D_{ik} means the delay for the edge server s_k to complete task t_i, f_k denotes the CPU-cycle frequency of the edge server s_k and R_k is the data transmission rate of sever s_k. By using frequency division multiple access (FDMA), according to Shannon Hartley theorem [40], the data transmission rate R_k can be modeled as:

$$R_k = B\log_2(1 + \frac{H_k p}{Bn_0}) \tag{13}$$

where the available system bandwidth for the mobile user is B Hz, and the transmit power of each MEC server is p. The n_0 is the background noise, and $H_k = {d_k}^{-\vartheta}$ is the channel power gain between the user and the server s_k, where ϑ is the path-loss exponent, d_k is the distance from user to server s_k. We can see that the condition of wireless channel R_k is closely related to the transmission distance between the user and the edge server.

Similarly, the energy consumption of tasks offloaded to edge server s_k can be computed as

$$E_k = \sum_{i=1}^{M} x_{ik}E_{ik} = \sum_{i=1}^{M} x_{ik}\frac{v_i p}{R_k} \tag{14}$$

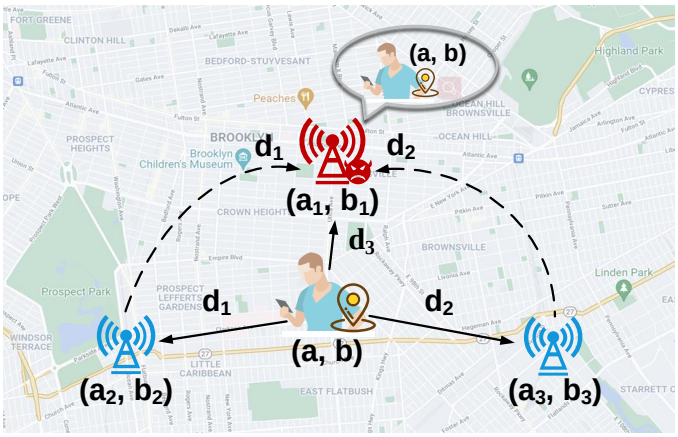


Fig. 2. Location privacy disclosure in multi-server scenario.

3.4 Design Objective

According to the calculation in [41], [42], the user could make a reasonable offloading strategy χ based on the conditions of wireless channels to minimize the total cost, which can be modeled as

$$\min_{\chi} C = \min_{\chi} \lambda \cdot E + (1 - \lambda) \cdot D \tag{15}$$

However, task offloading based on the conditions of wireless channels could disclose the user's location privacy. It can be seen from the definition of H_k that the wireless channel condition is closely related to the distance between the user and the edge server. If an untrusted MEC server s_k detects the system bandwidth of the mobile user and the background noise of the wireless channel, it could calculate the distance from the user to itself:

$$d_k = [\frac{Bn_0}{p}(2^{\frac{R_k}{B}} - 1)]^{\vartheta} \tag{16}$$

where R_k could be estimated according to the size of offloaded tasks. Therefore, an untrusted MEC server with prior knowledge can infer the wireless channel condition by analyzing the size of offloaded tasks, and then calculate the distance between itself and the user. In reality, a mobile

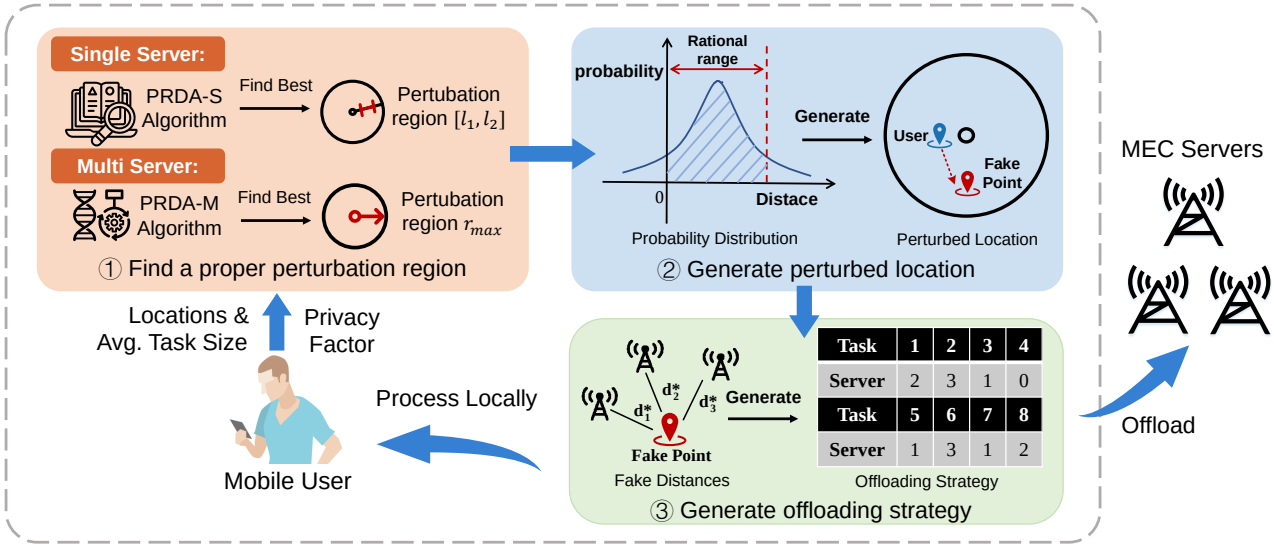


Fig. 3. The workflow of location privacy-aware task offloading framework for one mobile user.

user often performs task offloading with multiple edge servers, as shown in Figure 2. Once these servers collude, they could estimate the location of the mobile user to a certain precision. For example, the user's coordinate is (a, b), and its three edge servers may calculate their distances d_1, d_2, d_3 away from the user. If three servers collude, they can calculate (a, b) by solving the binary quadratic equation.

In this paper, we aim to design a location privacy-aware task offloading framework (LPA-Offload) that considers both resource optimization and location privacy protection. Let PL denote the privacy leakage level of the user, and the design objective of this paper is to minimize the weighted sum of computation cost C and privacy leakage level PL (i.e. maximizing system utility U), which can be modeled as:

$$\begin{aligned} \max\ U &= \max\ -[(1-\omega)C + \omega PL] \\ &= \max\ -(1-\omega)[\lambda \cdot E + (1-\lambda) \cdot D] - \omega PL \end{aligned} \tag{17}$$

where ω is a customized privacy factor that reflects the personalized location privacy requirement of the mobile user [43], [44]. In real-world systems, due to the differences in identities and preferences, users have different needs for location privacy protection [45], [46]. For example, users involved in confidentiality work are more privacy-sensitive than others. The larger the ω is, the more attention the mobile user pays to location privacy protection. By adaptively adjusting the ω, mobile users can achieve the trade-off between location privacy and computation cost according to their dynamic and personalized privacy requirements.

4 Location Privacy-Aware Task Offloading Framework

As mentioned in Sec.1, there are two main challenges to address: 1) how to protect the user's location information so that attackers cannot infer it by monitoring the size of offloaded tasks, and 2) how to adaptively decide the offloading strategy for each user according to the personalized location privacy requirement that achieves trade-off between computation cost and location privacy. To solve these challenges, we propose a location privacy-aware task offloading framework (LPA-Offload) that ensures strict and provable location privacy protection while achieving efficient task offloading. In this section, we first give an overview of LPA-Offload and then describe its design details.

4.1 Overview of Location Privacy-Aware Task Offloading Framework

Figure 3 shows the workflow of LPA-Offload for one mobile user. Our framework develops a location perturbation mechanism for user location protection and further proposes a perturbation region determination mechanism and an offloading strategy generation mechanism for each user to adaptively make an efficient privacy-preserving offloading strategy according to the user's privacy requirement. To be specific, when a mobile user joins the MEC systems, our framework would help the user adaptively select a proper perturbation region based on its location, privacy factor, and average task size. Then, in the determined region, the user could generate fake locations based on our designed probability density function (PDF) following the DP mechanism. After that, the user would generate efficient offloading strategies according to the fake locations, which would guide the user to offload suitable tasks to the server and process the rest locally. Note that the computation cost is still calculated based on the real locations, and only the offloading strategies are generated with the fake locations.

4.2 Location Perturbation Mechanism

To protect the mobile users' location privacy with a strict and provable privacy guarantee during task offloading, we intend to ask each mobile user to perturb its real location with differential privacy and then decide the offloaded tasks according to the perturbed location. However, to ensure the rationality of task offloading, the location after perturbation should be constrained in the coverage of MEC servers since the server could only provide computing services for

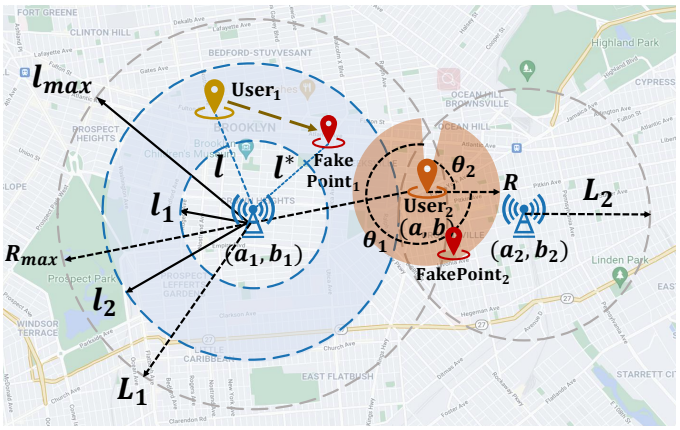


Fig. 4. Location perturbation. $User_1$ is the single-server scenario, and $User_2$ is the multi-server scenario. $FakePoint_1$ and $FakePoint_2$ are perturbed locations, respectively.

users within its coverage and it is unreasonable to offload tasks outside the coverage. This raises challenges to the differential privacy-based perturbation mechanism since the noise added to the data is often randomly selected from a Laplace distribution, which varies from $-\infty$ to $+\infty$. Hence, we redesign the probability density function of Laplace to limit the range of the data distribution. In the following, we discuss the issue of location perturbation under single-server and multi-server separately.

Single-server scenario. In the case of sparse distribution of edge servers, the user could only be covered by a single MEC server. As described in Section 3.4, an untrusted MEC server could infer the distance between the mobile user and itself in the single-server scenario. Thus, we need to perturb the real distance to a fake distance within the MEC server's coverage, as shown in Figure 4 ($User_1$). Let l, l^* denote the real distance and fake distance, respectively, and l_{max} represents the maximum coverage radius of the edge server. The user perturbs l to l^* within the range of $[l_1, l_2]$, which is related to the user's privacy requirement (i.e. ω). To ensure the rationality of task offloading, the range l_1, l_2 should be constrained in $[0, l_{max}]$.

Therefore, we design a new distribution to replace Laplace distribution in our mechanism, which can keep the perturbed distance within the range of $[l_1, l_2]$ while satisfying ϵ-differential privacy. The probability density function (PDF) of perturbing l to l^* is presented as follows:

$$pdf(l^*|l) = \begin{cases} \frac{\epsilon}{2\Delta l} e^{-\frac{\epsilon|l^*-l|}{\Delta l}} \cdot \frac{1}{K} & \text{, if } l^* \in [l_1, l_2] \\ 0 & \text{, otherwise} \end{cases} \tag{18}$$

where $K = 1 - \frac{1}{2}e^{\frac{\epsilon(l_1-l)}{\Delta l}} - \frac{1}{2}e^{-\frac{\epsilon(l_2-l)}{\Delta l}}$, $\Delta l = l_2 - l_1$, and $l_1, l_2 \in [0, l_{max}]$. The rationality of $pdf(l^*|l)$ is proved in Appendix A.

Multi-server scenario. In the case of dense distribution of edge servers, a user can be covered by several MEC servers. As described in Section 3.4, not only the distance but also the exact coordinate of the mobile user could be calculated in the multi-server scenario. Thus, we need to perturb the user's coordinates to protect the user's location privacy, as shown in Figure 4 ($User_2$). We take the user's real coordinate as the origin and perturb the coordinate in a circle with the radius R. To satisfy the rationality of task offloading, R should be constrained in $[0, R_{max}]$, where R_{max} is the longest distance from the user to the coverage boundary of edge servers. However, the mobile user may be located at the edge of MEC servers' coverage (Figure 4 ($User_2$)), and part of the angle θ would be out of the coverage, which needs to be discarded when the radius R is determined. Therefore, the perturbation region for the user can be denoted as $\{R, [\theta_1, \theta_2]\}$.

Since the variables R and θ are independently distributed and obey range-constrained Laplace distribution and uniform distribution respectively, the probability density function for the multi-server scenario can be formed by multiplying the two distributions, which is designed as

$$\begin{aligned} D_{\epsilon,R}(r,\theta) &= D_{\epsilon,R}(r) \cdot D_{\epsilon,R}(\theta) \\ &= \begin{cases} \frac{1}{\theta_2-\theta_1}\left(\frac{\epsilon}{2\Delta r} e^{-\frac{\epsilon r}{\Delta r}} \cdot \frac{1}{Z}\right) & \text{, if } r \in [0, R], \theta \in [\theta_1, \theta_2] \\ 0 & \text{, otherwise} \end{cases} \end{aligned} \tag{19}$$

where $Z = \frac{1-e^{-\epsilon}}{2}$ and $\Delta r = R - 0 = R$. We denote the real location $\Lambda = (a, b)$, then the perturbed location can be denoted as $\Lambda^* = (a + r\cos\theta, b + r\sin\theta)$. To simplify, suppose that the mobile user at the edge is covered by two MEC servers, the locations of which are (a_1, b_1) and (a_2, b_2), as shown in Figure 4. Then we can get that

$$\begin{cases} \theta_1 = -(\pi + \arccos\frac{R^2+(a-a_1)^2+(b-b_1)^2-{L_1}^2}{2R\sqrt{(a-a_1)^2+(b-b_1)^2}} \\ \qquad\qquad - \arctan\sqrt{\frac{(b-b_1)^2}{(a-a_1)^2}}) \\ \theta_2 = \arccos\frac{R^2+(a-a_2)^2+(b-b_2)^2-{L_2}^2}{2R\sqrt{(a-a_2)^2+(b-b_2)^2}} - \arctan\sqrt{\frac{(b-b_2)^2}{(a-a_2)^2}} \end{cases} \tag{20}$$

where L_1 and L_2 are the coverage of two servers, respectively. The proof of the rationality of $D_{\epsilon,R}(r,\theta)$ is shown in Appendix B.

4.3 Perturbation Region Determination Mechanism

As mentioned in the above subsection, the region of perturbation ($[l_1, l_2]$ or $\{R, [\theta_1, \theta_2]\}$) is related to the user's privacy requirement. Hence, given the privacy factor and locations of MEC servers, the key issue is how to adaptively determine the perturbation region for the location perturbation mechanism, so that the offloading strategy based on the perturbed location could get the maximum utility. In this subsection, we introduce the perturbation region determination mechanism to achieve a trade-off between computation cost and privacy leakage according to the personalized privacy factor.

Let ϕ denote the perturbation region (i.e. $[l_1, l_2]$ in single-server and $\{R, [\theta_1, \theta_2]\}$ in multi-server). C_{real} denotes the computation cost when generating offloading strategy according to the real location. $\mathcal{C}_\phi$ and PL_ϕ represent the expected computation cost and privacy leakage respectively for all possible offloading strategies when the perturbation region is ϕ. Then, the offloading utility maximization problem ζ in Eq. (17) could be reconstructed as:

$$\begin{aligned} \zeta &= \max_{\phi} U \\ &= \max_{\phi} -[(1-\omega)(C_\phi - C_{real}) + \omega PL_\phi] \\ \text{s.t.} &\sum_{k=1}^{N} x_{ik} \le 1, \quad x_{ik} \in \{0, 1\} \end{aligned} \tag{21}$$

Algorithm 1: Perturbation Region Determination Algorithm for Single-Server Scenario

Input: Real distance l, Privacy factor ω, Average Task size V
Output: Optimal perturbation region $[{l_1}^*, {l_2}^*]$

1 **for** l_1= *0 to* l_{max} *(step* SL*)* **do**
2 **for** l_2= l_1 *to* l_{max} *(step* SL*)* **do**
3 **Generate** a series of perturbed distances $P = \{l^* \,|l^* \in [l_1, l_2]\}$ (Eq. (18));
4 **for** $l^* \in P$ **do**
5 **Search** the computation cost and privacy leakage for l^* in the table F;
6 **if** l^* *not found in* F **then**
7 **Make** the optimal offloading strategy for l^* through traversal;
8 **Calculate** the computation cost C_{l^*} and privacy leakage PL_{l^*}, and add them to the table F;
9 **end**
10 **end**
11 **Calculate** the average cost C_ϕ and privacy leakage PL_ϕ to obtain the expected utility U_ϕ (Eq. (21));
12 **end**
13 **end**
14 **Compare** the expected U_ϕ of all regions and obtain the optimal region $[{l_1}^*, {l_2}^*]$ that has the maximum utility.
15 **return** $[{l_1}^*, {l_2}^*]$

Algorithm 2: Perturbation Region Determination Algorithm for Multi-Server Scenario

Input: User location Λ, Privacy factor ω, Average Task size V
Output: Optimal region $\{R_{opt}, [{\theta_1}^*, {\theta_2}^*]\}$

1 **for** R= *0 to* R_{max} *(step* SL*)* **do**
2 **Calculate** the angles $[\theta_1, \theta_2]$ within MEC servers' coverage for R (Eq. (20));
3 **Generate** a series of perturbed locations $P = \{\Lambda^* | r_{\Lambda^*,\Lambda} \in [0, R], \theta \in [\theta_1, \theta_2]\}$(Eq. (19));
4 **for** $\Lambda^* \in P$ **do**
5 **Search** the computation cost and privacy leakage for Λ^* in the table F;
6 **if** Λ^* *not found in* F **then**
7 **Make** the optimal offloading strategy for Λ^* with GA algorithm;
8 **Calculate** the computation cost C_{Λ^*} and privacy leakage PL_{Λ^*}, and add them to the table F;
9 **end**
10 **end**
11 **Calculate** the average cost C_ϕ and privacy leakage PL_ϕ to obtain the expected utility U_ϕ (Eq. (21));
12 **end**
13 **Compare** the expected U_ϕ of all regions and obtain the optimal region $\{R_{opt}, [{\theta_1}^*, {\theta_2}^*]\}$ that has the maximum utility.
14 **return** $\{R_{opt}, [{\theta_1}^*, {\theta_2}^*]\}$

where $C_\phi - C_{real}$ represents the difference between the expected computation cost of the offloading strategies according to real distances and fake distances.

Let ϖ denote the possible perturbed location and C_ϖ represents the computation cost based on the real location and offloading strategy generated from perturbed location ϖ. The expected cost $\mathcal{C}_\phi$ can be calculated by:

$$\begin{aligned}\mathcal{C}_\phi &= \frac{\int_\phi Pr(\varpi|\phi) \cdot C_\varpi \, \mathrm{d}\varpi}{\int_\phi Pr(\varpi|\phi) \, \mathrm{d}\varpi} \\ &= \int_\phi Pr(\varpi|\phi) \cdot C_\varpi \, \mathrm{d}\varpi\end{aligned} \tag{22}$$

where $Pr(\varpi|\phi)$ is the probability when the perturbed location is ϖ.

Moreover, we use Kullback Leibler divergence (KL divergence) [47] to measure privacy leakage. The level of privacy leakage in the single-server scenario can be denoted as:

$$PL_{l_1,l_2} = -\log \int_{l_1}^{l_2} Q\left(l^*|l\right) \log \frac{Q\left(l^*|l\right)}{P\left(l^*|l\right)} dl^* \tag{23}$$

where l and l^* represent the real distance and perturbed distance of the mobile user, respectively.

Similarly, we define the privacy leakage level in the multi-server scenario as:

$$PL_R = -\log \int_0^R \int_{\theta_1}^{\theta_2} Q\left(\Lambda^*|\Lambda\right) \log \frac{Q\left(\Lambda^*|\Lambda\right)}{P\left(\Lambda^*|\Lambda\right)} d\theta dr \tag{24}$$

where Λ and Λ^* represent the real location and perturbed location of the mobile user, respectively. When PL is larger, the level of privacy leakage is higher.

Based on Eq. (23) and (24), when the perturbation region increases, the user's location will have a higher probability to be perturbed to a farther location, which incurs a low level of privacy leakage but a significant increase in computation cost. Therefore, it is necessary to determine a proper perturbation region for the trade-off between computation cost and privacy leakage.

Perturbation region determination for single-server scenario. In the single-server scenario, the perturbation region determination algorithm determines a proper region $[l_1, l_2]$ to achieve the best utility. The general process of perturbation region determination algorithm for single-server scenario (PRDA-S) is shown in Algorithm 1. As shown in line 3, since the distances are not discrete, the probability density function can be simulated by generating a series of perturbed distances l^* based on Eq. (18). Then, we use a table F to record the computation cost and privacy leakage of all these l^* to avoid repeated computation. For all possible regions $[l_1, l_2]$, we compare the expected utility and select the optimal region $[{l_1}^*, {l_2}^*]$ which has the maximum utility.

Perturbation region determination for multi-server scenario. Similarly, the perturbation region determination algorithm for multi-server scenario (PRDA-M) determines a proper region $\{R, [\theta_1, \theta_2]\}$ to achieve the trade-off between computation cost and privacy leakage. As shown in

Algorithm 3: Location Privacy-aware Task Offloading

Input: User's location, Server number N, Task size V
Output: Offloading strategy χ^*

1 **if** *Server number N==1* **then**
2 **Call** Algorithm 1 to determine perturbation region $[l_1, l_2]$;
3 **Generate** a perturbed location according to Eq. (18) with the determined region;
4 **Make** an optimal offloading strategy χ^* with traversal method for the fake distance;
5 **else**
6 **Call** Algorithm 2 to determine perturbation region $\{R_{opt}, [{\theta_1}^*, {\theta_2}^*]\}$;
7 **Generate** a perturbed location according to Eq. (19) with the determined region;
8 **Make** an optimal offloading strategy χ^* with GA algorithm for the fake location;
9 **end**
10 **return** χ^*;

Algorithm 2, we first calculate the possible angles $[\theta_1, \theta_2]$ for R based on Eq. (20). Then, similar to the single-server scenario, we generate a series of Λ^* to fit the probability density function and then make the optimal offloading strategies for all these potential perturbed locations Λ^*. The user calculates the computation cost and privacy leakage for all Λ^*, and also uses a table F to record the results for efficient computing. By gradually increasing the region R, the goal is to find the optimal region $\{R_{opt}, [{\theta_1}^*, {\theta_2}^*]\}$ which has the maximum utility. Note that, the offloading strategy generation mechanism in Algorithm 2 (Line 7) will be detailed in Sec.4.4.

4.4 Offloading Strategy Generation Mechanism

After the perturbation region is selected, the location perturbation mechanism could be determined to perturb the user's real location. Thus, the mobile user could generate a perturbed location according to Eq. (18) or (19), which provides a location privacy guarantee with the differential privacy method. Then, the mobile user could make a satisfactory offloading strategy based on the fake distances between MEC servers and the perturbed location. The general process of location privacy-aware task offloading is shown in Algorithm 3.

In the single-server scenario, the mobile user could only offload tasks to one server or process locally, so the traversal method could be used to find the optimal offloading strategy. However, in the multi-server scenario, the user could offload tasks to multiple MEC servers, which process the tasks in parallel. The increase in the server number results in the explosive growth of the possible offloading strategies, which makes the offloading strategy generation NP-hard. We prove it as follows:

Theorem 1. Offloading strategy generation problem is an NP-hard problem for the multi-server scenario.

Proof 1. Consider a special case where all MEC servers and the local device have the same latency for completing task i, that is $D_{il} = D_{i1} = \cdots = D_{iN} = D_i$, with $\lambda_2 = 1$, and $\lambda_1 = 0, \omega = 0$ [38]. In this case, offloading utility maximization problem is equivalent to

$$\begin{aligned} \min_{\chi} \ \max_{k \in N} \quad & \sum_{i=1}^{M} x_{ik} D_i \\ \text{s.t.} \quad & \sum_{k=1}^{N} x_{ik} \leq 1, i = 1, 2, 3 \ldots, M \\ & x_{ik} \in \{0, 1\} \end{aligned} \tag{25}$$

The special case is known as the makespan minimization problem for identical parallel machines [48], which is NP-hard. Therefore, offloading utility maximization problem in the multi-server scenario is also NP-hard according to [49].

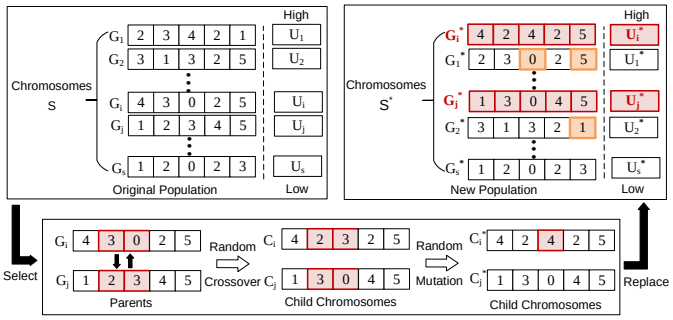


Fig. 5. The process of GA algorithm for offloading strategy generation.

Thus, we utilize genetic algorithm (GA) to resolve the issue because the structure of chromosomes could well represent the offloading strategy which consists of the relationship between multiple tasks and servers. To better illustrate the GA-based offloading strategy generation, we give an example in Figure 5, where chromosome G_i denotes the possible offloading strategy and each element represents an association of a sub-task and a server [50]. For example, $G_i = [4, 3, 0, 2, 5]$ represents that tasks t_1 to t_5 are assigned to server s_4, s_3, s_0, s_2, and s_5 sequentially, where s_0 denotes the local device.

The algorithm is shown in Figure 5 and described as follows.

1) **Initialize** a series of chromosomes S (the total size is s) randomly to represent different offloading strategies.
2) **Evaluate** the utility U of all chromosomes, which is the opposite of the weighted sum of C and PL.
3) **Select** $s * \beta$ chromosomes as parents randomly where α is the crossover probability.
4) **Generate** child chromosomes from multiple pairs of parents by random crossover and mutation operations with the mutation probability β.
5) **Add** child chromosomes to S and sort the utility U to keep the former s chromosomes.
6) **Repeat** steps 3) to 5) until the target iteration is reached or no more suitable chromosomes are generated.

The optimal offloading strategy χ^* is the chromosome with the largest utility (the top chromosome).

In GA, the selection, crossover, and mutation operators are utilized to generate more new offloading strategies

which may contain better ones with less computation cost and privacy leakage (higher utility). The higher crossover probability α and mutation probability β could generate more offloading strategies one time which would help find the optimal offloading strategies faster or a higher-utility strategy in the pre-set iterations. However, the crossover probability and mutation probability are constrained by the storage and computing capability of the device because too many strategies would affect the availability of the device.

5 Theoretical Analysis

In this section, we theoretically analyze the proposed mechanisms for both single-server and multi-server scenarios from the perspective of privacy protection.

Theorem 2. The location perturbation mechanism for the single-server scenario satisfies ϵ-differential privacy.

Proof 2. We define the adjacent distance l, l', which have the same optimal region $[{l_1}^*, {l_2}^*]$. The probability that l gets confused into l^* is $Pr(l^*|l)$. The probability that l' gets confused into l^* is $Pr(l^*|l')$. We prove that the ratio of $Pr(l^*|l)$ to $Pr(l^*|l')$ satisfies the definition of ϵ-differential privacy after applying the proposed mechanism.

$$\begin{aligned}\frac{Pr(l^*|l)}{Pr(l^*|l')} &= \frac{\frac{\epsilon}{2\Delta l}e^{-\frac{\epsilon|l^*-l|}{\Delta l}}\cdot\frac{1}{K}}{\frac{\epsilon}{2\Delta l}e^{-\frac{\epsilon|l^*-l'|}{\Delta l}}\cdot\frac{1}{K}} \\ &= e^{\frac{\epsilon|l^*-l'|-\epsilon|l^*-l|}{\Delta l}} \\ &\le e^{\frac{\epsilon|l'-l|}{\Delta l}} \\ &\le e^{\epsilon}\end{aligned}$$

Therefore, ϵ-DP is achieved in the location perturbation mechanism for the single-server scenario.

Theorem 3. The location perturbation mechanism for multi-server scenario satisfies (ϵ, δ)-differential privacy.

Proof 3. We define the adjacent point Λ, Λ', which are under the same optimal perturbation radius R_{opt}. The probability of Λ being confused with Λ^* is $Pr(\Lambda^*|\Lambda)$ and Λ' being confused with Λ^* is $Pr(\Lambda^*|\Lambda')$. We prove that the ratio of $Pr(\Lambda^*|\Lambda)$ to $Pr(\Lambda^*|\Lambda')$ satisfies the definition of (ϵ, δ)-differential privacy after applying the proposed mechanism.

$$\begin{aligned}\frac{Pr(\Lambda^*|\Lambda)}{Pr(\Lambda^*|\Lambda')} &= \frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}\left(\frac{\frac{\epsilon}{2Z\Delta r}e^{-\frac{\epsilon r}{\Delta r}}}{\frac{\epsilon}{2Z\Delta r}e^{-\frac{\epsilon r'}{\Delta r}}}\right) \\ &= \frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}\cdot e^{\frac{\epsilon(r'-r)}{\Delta r}} \\ &\le \frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}\cdot e^{\frac{\epsilon R_{opt}}{\Delta r}} \\ &= \frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}\cdot e^{\epsilon} \\ &= e^{\epsilon}+\left(\frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}-1\right)\cdot e^{\epsilon}\end{aligned}$$

Assumed that $\delta = (\frac{\theta_2^{'}-\theta_1^{'}}{\theta_2-\theta_1}-1)\cdot e^{\epsilon}\cdot Pr(\Lambda^*|\Lambda')$, we can get that $Pr(\Lambda^*|\Lambda) \le e^{\epsilon}Pr(\Lambda^*|\Lambda')+\delta$. Therefore, (ϵ, δ)-DP is achieved in the location perturbation mechanism for the multi-server scenario.

6 Performance Evaluation

In this section, we evaluate the performance of LPA-Offload with some existing algorithms to validate its effectiveness. We consider a MEC architecture in which many MEC servers with a service coverage of 1km are deployed in a 6km×6km area, and mobile users are randomly distributed in this area. To simulate the different server densities, we distribute servers randomly in this large space, and the number of MEC servers ranges from 50 to 100. In this setting, each user would access 2-10 servers normally.

Setup. We follow the settings in [51] that the CPU-cycle frequency of the mobile device is set to 1 GHz, and that of all MEC servers is set to 10 GHz. For Eq. (11) and Eq. (13), according to the settings of [1], [52], we set the available channel bandwidth $B = 0.1$ MHz (the average bandwidth of multiple users), the transmit power $p = 100$ mW, the background noise power $n_0 = -174$ dBm, the path loss factor $\theta = 4$, the parameters of energy cost $\kappa = 10^{-27}$ J/cycle and $\gamma = 1000$ cycle/bit. For simplicity, we set $\lambda = 0.5$, which means the mobile users have the same preference for computing delay and energy consumption, and the generated size of each task is uniformly distributed in $[1, 10]$ MB. Besides, the settings related to the GA method are that the total size of chromosomes s is 200, the crossover probability α is 0.7, and the mutation probability β is 0.2.

Environment. All the experiments are run in Python and on the same machine with 8G RAM, IntelCore i5 processor. We run each experiment 100 times and report the average results.

Metrics. We use two metrics to evaluate the effectiveness of our proposed framework: privacy leakage (PL) and average cost (AvgCost).

- Privacy Leakage: when the perturbed location is less than d_{min} away from the real location, the user's location information would be disclosed. Thus, Privacy Leakage is the probability that the distance d^* between the perturbed location and the real location is less than or equal to d_{min}, which can be expressed as:

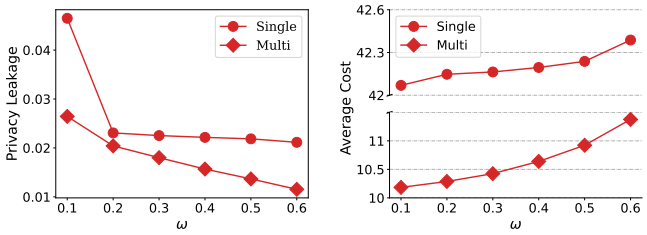
$$\text{Privacy Leakage} = \frac{1}{H}\sum_{j=1}^{H} I(d_j^* \le d_{min})$$

  where H is the number of times the real locations are perturbed. $I(d_j^* \le d_{min}) = 1$ if $d_j^* \le d_{min}$ is true, otherwise its value is 0.
- Average Cost: the average computation cost of task offloading with different random location distributions for H times, which can be defined by:

$$\text{Average Cost} = \frac{1}{H}\sum_{j=1}^{H} C(j)$$

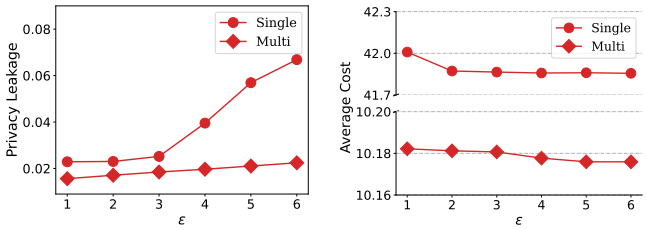
Benchmark. We compare LPA-Offload with the following algorithms:

- OFDMA: The state-of-art algorithm [53] makes the offloading strategy according to the real location and the channel condition without location privacy protection.
- PATO: The state-of-the-art location privacy-aware offloading mechanism [6] without differential privacy, where the user's privacy protection level is customized as an indicator of the optimization goal. It provides the



(a) Impact of ω on privacy leakage (b) Impact of ω on cost

Fig. 6. Evaluation on parameter ω.



(a) Impact of ϵ on privacy leakage (b) Impact of ϵ on cost

Fig. 7. Evaluation on parameter ϵ.

same level of privacy guarantee for users at each time and is vulnerable to inference attacks. Since it assumes that the task size is always consistent and cannot be applied to the multi-server scenarios, we only compare it with ours in the inference attack.

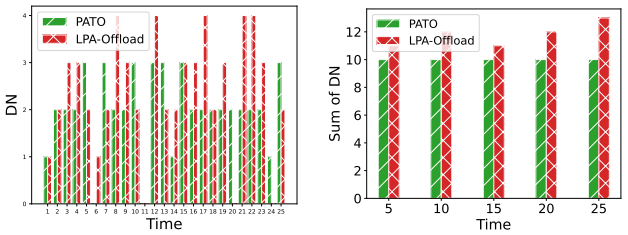
- MaxRange: A particular case of LPA-Offload. It selects the largest perturbation region (i.e. $l_1 = 0, l_2 = l_{max}$ or $R = R_{max}$) to perturb location based on Eq. (18) and Eq. (19) without trade-off between computation cost and privacy leakage.

6.1 Evaluations of Parameters

The parameter ω is the privacy factor that represents the user's personalized privacy requirement, and ϵ is the privacy budget in differential privacy. The two parameters affect the trade-off between computation cost and privacy leakage level. In Fig.6 and Fig.7, we evaluate the effects of these two parameters on the performance of LPA-Offload in both single-server and multi-server scenarios.

Parameter ω. In Fig. 6(a), we can see that with the increase of the privacy factor ω, the privacy leakage of LPA-Offload decreases in both scenarios. The reason is that, with higher ω, the user focuses more on location privacy, and tends to use a larger perturbation region to perturb the real location. In this case, the fake location has a smaller probability of being close to the real location. Since the user offloads tasks according to the fake distances between MEC servers and the perturbed location, the attacker cannot infer the real location according to the offloading strategy. Hence, LPA-Offload can provide higher privacy protection with a higher ω.

Fig. 6(b) shows that the average cost increases for a larger value of ω. This is because, with a larger perturbation region, the user has a higher probability to perturb the location to a farther location, which is the basis for formulating the offloading strategy. A farther perturbed location would cause the offloading strategy far from the optimal strategy based on the real location, and increases the average computation cost. Thus, with lower ω, LPA-Offload can achieve lower computation cost.



(a) Comparison on DN (b) Comparison on sum of DN for 5-time slots

Fig. 8. Evaluation of the vulnerability of PATO and LPA-Offload to inference attacks.

Parameter ϵ. Fig. 7(a) and 7(b) show that the average cost of LPA-Offload decreases while privacy leakage increases for a larger value of ϵ. The reason is that a larger ϵ provides a weaker privacy guarantee, which means that the probability of the perturbed location being far away from the real location is smaller, leading to a lower cost and a higher privacy leakage.

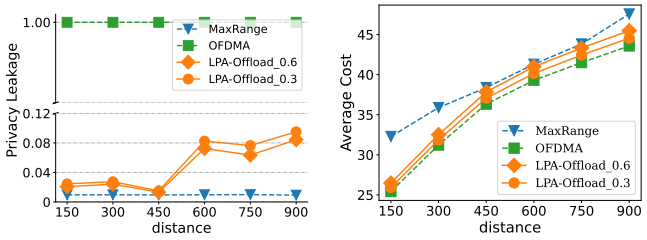
In addition, the average cost of the multi-server scenario is always smaller than that of the single-server scenario. It's because more servers working in parallel could reduce the computing delay. A smaller computation cost would also prompt the perturbation region to increase to reduce privacy leakage. Thus, the privacy leakage of the multi-server is also smaller than that of the single-server scenario.

6.2 Evaluations on Inference Attack

We use the EUA dataset [54], containing the geographical locations of 125 cellular base stations and 816 mobile users in Melbourne central business district area, to verify the effectiveness of LPA-Offload against the inference attack. We randomly select the location of the mobile user and generate a random number of tasks at each time.

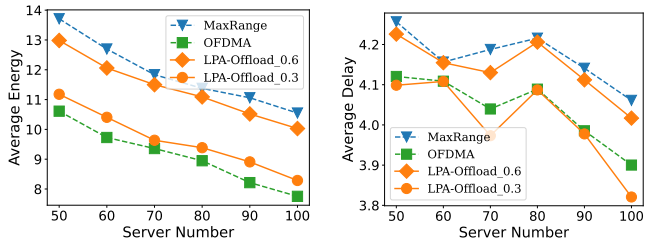
As discussed in Sec.3, the attacker will use the wireless channel condition to infer the locations of mobile users. If the difference between the number of offloaded tasks and generated tasks is always a similar value, the attacker could calculate the size of generated tasks according to the offloaded tasks, and further infer the conditions of wireless channels. Based on the experiment in [6], we use the difference between the number of offloaded tasks and generated tasks (DN) as the metric to evaluate the effectiveness of our framework on inference attacks. If DN (or the sum of DN for a fixed period) is always close to a similar value, it means that the mechanism is vulnerable to inference attacks, which would lead to location privacy leakage. We compare LPA-Offload with PATO on DN and the sum of DN for a period of time, as shown in Figures 8(a) and 8(b).

In Figure 8(a), the difference between generated tasks and offloaded tasks for both algorithms is always changing with time slot, which means LPA-Offload and PATO can



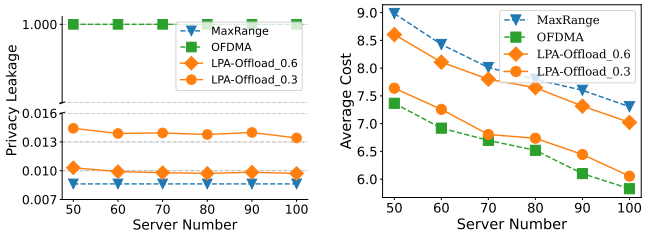
(a) Impact of distance on PL (b) Impact of distance on cost

Fig. 9. Evaluation of distance on privacy leakage and average cost for single-server scenario



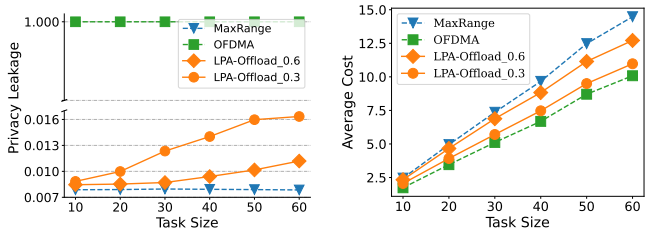
(a) Impact of server number on Energy (b) Impact of server number on delay

Fig. 11. Evaluation of server number on average energy and delay for multi-server scenario



(a) Impact of server number on PL (b) Impact of server number on cost

Fig. 10. Evaluation of server number on privacy leakage and average cost for multi-server scenario



(a) Impact of task size on PL (b) Impact of task size on cost

Fig. 12. Evaluation of task size on privacy leakage and average cost for multi-server scenario

protect mobile users from the inference attack since the attacker cannot get the real wireless channel condition. However, in Figure 8(b), we can see that the sum of the difference for 5-time slots in PATO is always 10, which means the wireless channel condition could be inferred in a long-term process and then leak location privacy. As for LPA-Offload, the difference between generated tasks and offloaded tasks is not constant because differential privacy brings randomness to the offloading strategy. Thus, our framework is not vulnerable to inference attacks and ensures stronger location privacy protection.

6.3 Evaluations of Privacy Leakage and Average Cost

In Figure 9, 10 and 12, we compare the privacy leakage and average cost of LPA-Offload (including LPA-Offload_0.3 and LPA-Offload_0.6, i.e. $\omega = 0.3$ and $\omega = 0.6$ respectively) and benchmarks (including MaxRange and OFDMA) against the distance, server number, and task size.

Single-server scenario. Figure 9(a) shows the PL of four algorithms against the distance between the user and the MEC server. We observe that LPA-Offload has a similar privacy leakage level to MaxRange, which means that both of them provide strict location privacy protection and have a low possibility of location privacy disclosure. Meanwhile, OFDMA does not have any location privacy protection, so the PL is close to 1. Although MaxRange has a better performance in privacy guarantee due to its larger perturbation region, this scheme is at the cost of greatly increasing the computation consumption.

In 9(b), LPA-Offload has almost the same average cost as OFDMA and is much better than MaxRange. The results demonstrate that our framework for the single-server scenario does not increase the computation cost obviously while providing strong location privacy protection. MaxRange increases the computation cost significantly because it has a larger perturbation region and a greater probability of perturbing the user's location to a farther location, which will lead to a greater gap between the offloading strategy and the optimal strategy.

In addition, with the increase of the distance, the average cost increases significantly because the channel condition becomes worse based on Eq. (13). Moreover, the PL increases with a larger distance because the perturbation region will be appropriately reduced to decrease the cost. However, the specific value of PL depends on the perturbation region, so there is a certain fluctuation.

What's more, LPA-Offload_0.6 has a smaller PL and a larger AvgCost than LPA-Offload_0.3. It's because the privacy requirement of the mobile user in LPA-Offload_0.6 is stronger and the mobile user would use a larger perturbation region to protect its location privacy.

Multi-server scenario. Figures 10 and 12 show the PL and AvgCost of four algorithms against the number of MEC servers and the task size. Similarly, LPA-Offload ensures low levels of privacy leakage and computation cost, which has a better performance than the other two algorithms. Specifically, OFDMA cannot provide a strict privacy guarantee, while MaxRange increases the computation cost obviously.

What's more, in Fig. 10(a) and 10(b), with the increase of the number of MEC servers, the cost of task offloading decreases gradually because more servers working in parallel can reduce processing time and have better channel conditions. In this situation, the user could focus more on privacy protection, and enlarge the perturbation region

appropriately to reduce the level of privacy leakage. We also evaluate the average energy and delay in Fig. 11. It could be found that the average energy and delay both have a decreasing trend with the increase of server number, while the energy has a greater impact on cost and the delay has some randomness.

In Fig. 12(a) and Fig. 12(b), with the increase in task size, the computation cost increases obviously because more tasks needed to be processed. In this case, the user would tend to narrow the perturbation region to reduce the computation cost, which leads to an increase in PL.

6.4 Evaluations on Runtime

The actual execution time of the proposed framework is shown in Table 2. As we described in the workflow, when the mobile user joins the system, our framework would help the user to select a proper perturbation region based on its location, privacy factor, and average task size first. The process of determining region is relatively long but only needs to be performed once. After that, the user could generate a series of fake locations and make different offloading strategies quickly, so that our framework is practical.

TABLE 2
The actual execution time of the proposed framework.

Runtime Type	Single-Server	Multi-Server
Region Determination (s)	442.35889	1980.74575
Location Perturbation (s)	1.20741	0.00099
Strategy Generation (s)	0.05781	97.52521

7 Discussion

In this section, we discuss the feasibility and limitation of LPA-Offload and introduce our future work.

Mobility. LPA-Offload could be applied for mobile users constrained in a small range because different locations of one user are adjacent, and their optimal perturbation regions are overlapped. With the average location of the mobile user, our framework could get a reasonably perturbed location and a satisfactory offloading strategy for location privacy preservation and efficient task offloading. But for mobile users with a large range of mobility, it might be difficult for our framework to make satisfactory offloading strategies given that the locations of one user might be far away from each other and even covered by different MEC servers. In future work, we would consider achieving online location privacy-aware task offloading on the location-varying scenario by reinforcement learning.

Multi-user scenario. LPA-Offload is effective in the multi-user scenario when the wireless channel conditions are relatively stable. Since the real bandwidth usage in the system is unknown to users, the mobile user would monitor the wireless channel and the transmission time to get the bandwidth and transmit power, and then make the offloading strategy on these values. When the wireless channel conditions are relatively stable, the estimated bandwidth and transmit power are close to the true values so that the user could make a satisfactory offloading strategy. But in the variable wireless channel conditions, there would be multi-channel wireless interference between users, and the available bandwidth and transmit power would change significantly. The estimated bandwidth and transmit power would be inaccurate for the users. In the future, we would consider using reinforcement learning to realize online task offloading on the bandwidth-varying scenario with multi-users.

8 Conclusion

In this paper, we proposed a location privacy-aware task offloading framework (LPA-Offload) based on differential privacy to protect the mobile user's location privacy while achieving efficient task offloading. In our framework, the location perturbation mechanism is utilized to perturb the user's location within a rational region. To balance location privacy protection and computation cost, we proposed a perturbation region determination mechanism to select a proper perturbation region that has the maximum expected utility, and then utilized the offloading strategy generation mechanism to generate a satisfactory offloading strategy according to the generated perturbed location. LPA-Offload is proved to satisfy (ϵ, δ)-differential privacy. Experimental results demonstrate the effectiveness of our framework, which has almost the same cost as the state-of-art method without privacy protection while providing a strong privacy guarantee.

References

[1] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2015.

[2] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal *et al.*, "Mobile-edge computing introductory technical white paper," *White paper, mobile-edge computing (MEC) industry initiative*, vol. 29, pp. 854–864, 2014.

[3] S. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Energy efficiency and delay tradeoff for wireless powered mobile-edge computing systems with multi-access schemes," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1855–1867, 2019.

[4] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint resource allocation and offloading strategies in cloud enabled cellular networks," in *2015 IEEE International Conference on Communications (ICC)*. IEEE, 2015, pp. 5529–5534.

[5] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *2016 IEEE international symposium on information theory (ISIT)*. IEEE, 2016, pp. 1451–1455.

[6] X. He, J. Liu, R. Jin, and H. Dai, "Privacy-aware offloading in mobile-edge computing," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–6.

[7] J. Dong, D. Geng, and X. He, "Privacy-aware task offloading via two-timescale reinforcement learning," in *2020 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE, 2020, pp. 220–225.

[8] Z. Wang, X. Pang, Y. Chen, H. Shao, Q. Wang, L. Wu, H. Chen, and H. Qi, "Privacy-preserving crowd-sourced statistical data publishing with an untrusted server," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1356–1367, 2018.

[9] Z. Wang, J. Li, J. Hu, J. Ren, Q. Wang, Z. Li, and Y. Li, "Towards privacy-driven truthful incentives for mobile crowdsensing under untrusted platform," *IEEE Transactions on Mobile Computing*, 2021.

[10] L. Wang, G. Qin, D. Yang, X. Han, and X. Ma, "Geographic differential privacy for mobile crowd coverage maximization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[11] Z. Wang, J. Hu, Q. Wang, R. Lv, J. Wei, H. Chen, and X. Niu, "Task-bundling-based incentive for location-dependent mobile crowdsourcing," *IEEE Communications Magazine*, vol. 57, no. 2, pp. 54–59, 2019.

[12] D. C. Nguyen, P. N. Pathirana, M. Ding, and A. Seneviratne, "Privacy-preserved task offloading in mobile blockchain with deep reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2536–2549, 2020.

[13] T. Li, H. Liu, J. Liang, H. Zhang, L. Geng, and Y. Liu, "Privacy-aware online task offloading for mobile-edge computing," in *International Conference on Wireless Algorithms, Systems, and Applications*. Springer, 2020, pp. 244–255.

[14] P. Sun, Z. Wang, L. Wu, Y. Feng, X. Pang, H. Qi, and Z. Wang, "Towards personalized privacy-preserving incentive for truth discovery in mobile crowdsensing systems," *IEEE Transactions on Mobile Computing*, vol. 21, no. 1, pp. 352–365, 2022.

[15] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3590–3605, 2016.

[16] J. Ren, J. Liu, Y. Zhang, Z. Li, F. Lyu, Z. Wang, and Y. Zhang, "An efficient two-layer task offloading scheme for mec system with multiple services providers," in *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 2022, pp. 1519–1528.

[17] J. Zhang, J. Du, Y. Shen, and J. Wang, "Dynamic computation offloading with energy harvesting devices: A hybrid-decision-based deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9303–9317, 2020.

[18] Q. Tang, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Decentralized computation offloading in iot fog computing system with energy harvesting: A dec-pomdp approach," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4898–4911, 2020.

[19] D. Zhang, L. Tan, J. Ren, M. K. Awad, S. Zhang, Y. Zhang, and P.-J. Wan, "Near-optimal and truthful online auction for computation offloading in green edge-computing systems," *IEEE Transactions on Mobile Computing*, vol. 19, no. 4, pp. 880–893, 2019.

[20] T. Liu, S. Sheng, L. Fang, Y. Zhang, T. Zhang, and W. Tong, "Latency-minimized and energy-efficient online task offloading for mobile edge computing with stochastic heterogeneous tasks," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2019, pp. 376–383.

[21] Z. Wang, J. Li, J. Hu, J. Ren, Z. Li, and Y. Li, "Towards privacy-preserving incentive for mobile crowdsensing under an untrusted platform," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2053–2061.

[22] X. Huang, K. Xu, C. Lai, Q. Chen, and J. Zhang, "Energy-efficient offloading decision-making for mobile edge computing in vehicular networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, no. 1, pp. 1–16, 2020.

[23] Y. Sun, T. Wei, H. Li, Y. Zhang, and W. Wu, "Energy-efficient multimedia task assignment and computing offloading for mobile edge computing networks," *IEEE Access*, vol. 8, pp. 36 702–36 713, 2020.

[24] Q. Jiang, Y. Zhang, and J. Yan, "Neural combinatorial optimization for energy-efficient offloading in mobile edge computing," *IEEE Access*, vol. 8, pp. 35 077–35 089, 2020.

[25] Z. Chen and X. Wang, "Decentralized computation offloading for multi-user mobile edge computing: A deep reinforcement learning approach," *EURASIP Journal on Wireless Communications and Networking*, vol. 2020, no. 1, pp. 1–21, 2020.

[26] X. Lyu, W. Ni, H. Tian, R. P. Liu, X. Wang, G. B. Giannakis, and A. Paulraj, "Optimal schedule of mobile edge computing for internet of things using partial information," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2606–2615, 2017.

[27] N. Nouri, P. Rafiee, and A. Tadaion, "Noma-based energy-delay trade-off for mobile edge computation offloading in 5g networks," in *2018 9th International Symposium on Telecommunications (IST)*. IEEE, 2018, pp. 522–527.

[28] T. He, E. N. Ciftcioglu, S. Wang, and K. S. Chan, "Location privacy in mobile edge clouds," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, 2017, pp. 2264–2269.

[29] H. Gao, W. Huang, T. Liu, Y. Yin, and Y. Li, "Ppo2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2022.

[30] Z. Xu, X. Liu, G. Jiang, and B. Tang, "A time-efficient data offloading method with privacy preservation for intelligent sensors in edge computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2019, no. 1, pp. 1–12, 2019.

[31] X. Xu, C. He, Z. Xu, L. Qi, S. Wan, and M. Z. A. Bhuiyan, "Joint optimization of offloading utility and privacy for edge computing enabled iot," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2622–2629, 2019.

[32] X. He, R. Jin, and H. Dai, "Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1814–1824, 2019.

[33] M. Min, X. Wan, L. Xiao, Y. Chen, M. Xia, D. Wu, and H. Dai, "Learning-based privacy-aware offloading for healthcare iot with energy harvesting," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4307–4316, 2018.

[34] X. Pang, Z. Wang, J. Li, R. Zhou, J. Ren, and Z. Li, "Towards online privacy-preserving computation offloading in mobile edge computing," in *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*, 2022, pp. 1179–1188.

[35] C. Dwork, "Differential privacy," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2006, pp. 1–12.

[36] C. Dwork and J. Lei, "Differential privacy and robust statistics," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 371–380.

[37] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[38] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, 2017.

[39] S. Cheng, Z. Chen, J. Li, and H. Gao, "Task assignment algorithms in data shared mobile edge computing systems," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 997–1006.

[40] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[41] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Transactions on Wireless Communications*, vol. 12, no. 9, pp. 4569–4581, 2013.

[42] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint computation offloading and user association in multi-task mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 12, pp. 12 313–12 325, 2018.

[43] X. He, R. Jin, and H. Dai, "Deep pds-learning for privacy-aware offloading in mec-enabled iot," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4547–4555, 2019.

[44] X. Pang, Z. Wang, D. Liu, J. C. Lui, Q. Wang, and J. Ren, "Towards personalized privacy-preserving truth discovery over crowdsourced data streams," *IEEE/ACM Transactions on Networking*, vol. 30, no. 1, pp. 327–340, 2021.

[45] P. Sun, H. Che, Z. Wang, Y. Wang, T. Wang, L. Wu, and H. Shao, "Pain-fl: Personalized privacy-preserving incentive for federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3805–3820, 2021.

[46] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, and H. Qi, "Personalized privacy-preserving task allocation for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1330–1341, 2018.

[47] T. Van Erven and P. Harremos, "Rényi divergence and kullback-leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[48] M. Pinedo and K. Hadavi, "Scheduling: theory, algorithms and systems development," in *Operations Research Proceedings 1991*. Springer, 1992, pp. 35–42.

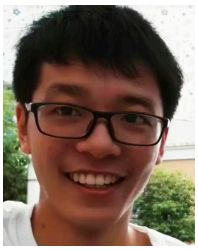
[49] M. R. Garey and D. S. Johnson, ""strong"np-completeness results: Motivation, examples, and implications," *Journal of the ACM (JACM)*, vol. 25, no. 3, pp. 499–508, 1978.

[50] A. A. Al-Habob, O. A. Dobre, and A. G. Armada, "Sequential task scheduling for mobile edge computing using genetic algorithm," in *2019 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2019, pp. 1–6.

[51] C.-F. Liu, M. Bennis, and H. V. Poor, "Latency and reliability-aware task offloading and resource allocation for mobile edge computing," in *2017 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2017, pp. 1–7.
[52] Y. Hao, L. Hu, Y. Qian, and M. Chen, "Profit maximization for video caching and processing in edge cloud," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 7, pp. 1632–1641, 2019.
[53] K. Cheng, Y. Teng, W. Sun, A. Liu, and X. Wang, "Energy-efficient joint offloading and wireless resource allocation strategy in multi-mec server systems," in *2018 IEEE international conference on communications (ICC)*. IEEE, 2018, pp. 1–6.
[54] P. Lai, Q. He, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, and Y. Yang, "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *Proc. of ICSOC*, 2018, pp. 230–245.

Zhibo Wang received the B.E. degree in Automation from Zhejiang University, China, in 2007, and his Ph.D degree in Electrical Engineering and Computer Science from University of Tennessee, Knoxville, in 2014. He is currently a Professor with the School of Cyber Science and Technology, Zhejiang University, China. His currently research interests include Internet of Thingks, AI security, data security and privacy. He is a Senior Member of IEEE and a Member of ACM.

Yunan Sun received the B.E. degree in Information Security from Wuhan University, China, in 2020. She is currently pursuing her Master degree at School of Cyber Science and Engineering, Wuhan University. Her research interest focuses on privacy protection in edge computing and Internet of Things.



Defang Liu received the B.E. degree in Information Security from Wuhan University, China, in 2020. He is currently pursuing his Master degree at School of Cyber Science and Engineering, Wuhan University. His research interest focuses on privacy protection in Internet of Things and edge intelligence.

Jiahui Hu received the M.S. degree in Cyber Security from Wuhan University, China, in 2019. She is currently pursuing her doctor degree at School of Cyber Science and Engineering, Zhe-Jiang University. Her research interest focuses on federated learning and privacy.

Xiaoyi Pang received the B.E. degree in Information Security from Wuhan University, China, in 2018. She is currently pursuing her Ph.D at School of Cyber Science and Engineering, Wuhan University. Her research interest focuses on privacy protection in mobile crowdsensing systems and edge intelligence.

Yuke Hu received the BS degree in information security from Wuhan University. He is working toward the PhD degree at Zhejiang University. His research interests include data security and privacy in multi-scenario applications. He is a student member of IEEE.

Kui Ren received the Ph.D. degree from the Worcester Polytechnic Institute. He is currently a Professor and the Associate Dean of the College of Computer Science and Technology, Zhejiang University, where he also directs the School of Cyber Science and Technology. Before that, he was the SUNY Empire Innovation Professor of The State University of New York at Buffalo. His current research interests include Data Security, IoT Security, AI Security, and Privacy. He received Guohua Distinguished Scholar Award from ZJU in 2020, IEEE CISTC Technical Recognition Award in 2017, SUNY Chancellor's Research Excellence Award in 2017, Sigma Xi Research Excellence Award in 2012 and NSF CAREER Award in 2011. Kui has published extensively in peer-reviewed journals and conferences and received the Test-of-time Paper Award from IEEE INFOCOM and many Best Paper Awards from IEEE and ACM including MobiSys'20, ICDCS'20, Globecom'19, ASIACCS'18, ICDCS'17, etc. His h-index is 74, and his total publication citation exceeds 32000 according to Google Scholar. Kui is a Fellow of IEEE, a Fellow of ACM and a Clarivate Highly-Cited Researcher. He is a frequent reviewer for funding agencies internationally and serves on the editorial boards of many IEEE and ACM journals. He currently serves as Chair of SIGSAC of ACM China.