Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

PPDO: a privacy preservation-aware delay optimization task-offloading algorithm for collaborative edge computing*

Chao JING[†‡1,2], Jianwu XU[1]

[1]College of Computer Science and Engineering, Guilin University of Technology, Guilin 541004, China

[2]Guangxi Key Laboratory of Embedded Technology and Intelligent System, Guilin 541004, China

[†]E-mail: jingchao@glut.edu.cn

Received Oct. 31, 2023; Revision accepted Jan. 5, 2024; Crosschecked

Abstract: Although collaborative edge computing (CEC) systems are beneficial in enhancing the performance of mobile edge computing (MEC), the issue of user privacy leakage becomes prominent during task offloading. To address this issue, we designed a privacy preservation-aware delay optimization task-offloading algorithm (PPDO) in a CEC system. By considering location and usage-pattern privacy protection, we established a privacy task model to interfere with the edge server and ensure user privacy. To address the extra delay arising from privacy protection, we subsequently leveraged a Markov decision processing (MDP) policy-iteration-based algorithm to minimize delays without compromising privacy. To simultaneously accelerate the MDP operation, we developed an extension that improves the PPDO by optimizing the action set. Finally, a comprehensive simulation was conducted using the EUA dataset. The results demonstrated that PPDO achieves the optimal trade-off between privacy protection and delay with a minimal delay compared with that of existing algorithms. Moreover, we examined the advantages and disadvantages of improving PPDO.

unedited

Key words: Collaborative edge computing; Task offloading; Privacy protection; Markov decision process
https://doi.org/10.1631/FITEE.2300741 **CLC number:**

1 Introduction

With the rapid growth of mobile networks, intelligence-based mobile devices have become widely used in numerous fields. Owing to the many requirements of resource-intensive applications (Ouyang et al., 2019), there is a significant increase in data volume and demand for less delay (Wang et al., 2017), especially for applications that require a high quality of service (QoS) and are vulnerable to delays (Taleb et al., 2017) (Chen et al., 2019), such as mobile gaming (Yang et al., 2020), virtual reality (VR) (Lin et al., 2021), image video processing (Mach and Becvar, 2017), and speech recognition (Yousaf et al., 2018). However, the resources of mobile cloud computing (MCC) are usually located far from the mobile devices, with high latency and energy consumption for data transmission owing to network congestion and speed limitations (Gao et al., 2023b). Fortunately, these issues can be addressed by the emerging technology of mobile edge computing (MEC) (Abbas et al., 2017). By situating servers at the edge of the network, edge computing can minimize latency (Dong et al., 2023).

Collaborative edge computing (CEC) systems have attracted significant attention owing to the constrained edge server (ES) resources of MEC as compared to MCC (Lee et al., 2021). CEC, characterized by the distribution of computational tasks among distributed edge nodes (Sahni et al., 2019), ensures

‡ Corresponding author

* Project supported by the National Natural Science Foundation of China (Nos. 62362018, 62341117, and 62262011), the Guangxi Key Research and Development Program (Nos. GUIKEAB23075116 and GUIKEAB23075175), and the Innovation Project of Guangxi Graduate Education (No. YCSW2023350)

ORCID: Chao JING, https://orcid.org/0000-0002-4695-8746

© Zhejiang University Press 2024

appropriate collaboration on ESs when faced with a risk of privacy leakage owing to a reduced communication overhead in selecting a closer ES. Moreover, when the network conditions are poor or the ES processing queues are congested, certain tasks can be processed by mobile devices instead of being offloaded to the ES. By appropriately offloading tasks, CEC can greatly improve the performance of MEC. Several studies have addressed the costs associated with task offloading in edge computing, which generally include execution delay and energy consumption. An algorithm based on Lyapunov optimization was proposed to solve the cost optimization problem in edge computing systems with limited computational resources (Mao et al., 2017). Similarly, (Cao et al., 2021) adopted the Lyapunov optimization framework to minimize user QoS losses stemming from network delay and low frame rates under cost constraints. In (Chu et al., 2023) a distributed online version of the mechanism was proposed to maximize user QoS in MEC by jointly optimizing service caching, resource allocation, and task offloading decisions. Qin et al. (Qin et al., 2021) devised a threshold-based distributed task offloading algorithm that allows MEC users to update their thresholds based on their cost functions. Another study (Ren et al., 2019) considered collaboration between cloud and edge computing, with a joint communication resource and computation resource allocation problem formulated to minimize the latency of mobile devices. Wang et al. (Wang et al., 2021a) designed a resource allocation scheme that rationally assigns computational and network resources under changeable edge-computing conditions. However, most of these studies have focused on cost optimization, ignoring the importance of privacy protection during offloading.

When ESs are located closer to mobile devices, the QoS of users can benefit from the advantages of edge computing. However, edge computing raises the severe issue of privacy leakage. In certain cases, ESs and mobile edge-computing environments cannot be trusted. For example, the private information of users may be exposed during task offloading (He et al., 2017b) (Wang et al., 2019) (Qian et al., 2021) (Wang et al., 2021b). This creates an opportunity for malicious users to leverage private information for tracking, fraud, attacks, and illegal exploitation (He et al., 2017a) (Mao et al., 2022) (He et al., 2019). The process of task offloading has previously been associated with privacy leaks (Bai et al., 2020). Because MEC servers may attempt to obtain user locations and usage patterns (Min et al., 2019), it is essential to determine the trade-off between latency optimization and privacy protection.

Numerous studies have addressed the privacy protection issues inherent to edge computing. One recent study (Wang et al., 2023) advocated a privacy preservation-aware task-offloading framework based on a location perturbation scheme that provides differential privacy guarantees. Gao(Gao et al., 2023a) et al. introduced a deep reinforcement learning method to minimize the risk of location privacy leakage during task offloading. Another study (Wang et al., 2020)employed a location privacy preservation-aware migration framework and defined the total system cost as a combination of the risk of location privacy leakage and delay of task offloading. Wei et al (Hua et al., 2023) computed risk associated with location leakage and applied it as a constraint in task offloading. In (Zhao et al., 2023), privacy-preserving models were designed by hiding user offloading decisions and intentionally offloading redundant tasks. Nonetheless, most of the aforementioned studies were only concerned with protecting location privacy while ignoring usage pattern. Although (Zhao et al., 2023) protected both location and usage pattern privacy by offloading redundant tasks, these tasks were derived from the historical tasks of the user's device that extended the latency.

In contrast to the aforementioned methods, we propose a privacy-preservation-aware offloading algorithm in CEC (PPDO) with the objective of optimizing delay under privacy considerations. In the CEC framework, tasks associated with mobile users can be offloaded to the edge server or processed by a local device. To ensure privacy during offloading, we enhanced the protection of both location and usage pattern by integrating the concept of privacy tasks. The main function of a privacy task is to create interference wherein the ES cannot analyze the userâĂŹs location or usage pattern information. Although the traditional MDP-based (Ksentini et al., 2014) approach effectively optimizes delays during task offloading, the direct use of MDP is time-consuming owing to the selection of optimal actions. We therefore improved MDP by eliminating the redundant action set. Thus, the proposed algorithm efficiently

unedited

optimizes latency when tasks are offloaded, achieving a balance between privacy protection and latency optimization. The main contributions of this work are summarized as follows.

- We established a multi-user CEC that minimizes the delay of task offloading without compromising privacy preservation. Specifically, we enhanced the privacy protection scheme with respect to location and usage pattern. We also defined privacy tasks to interfere with ES operations to prevent privacy leaks.

- We developed an improved MDP-based privacy preservation-aware task-offloading algorithm in CEC under the enhanced privacy protection scheme. The improved MDP model embraces a state set, probabilistic transfer function, reward function, and action set. Because the policy iteration of MDP may be highly time-intensive, we reduced the exploration of the action set. Specifically, given a sufficiently small number of tasks, or a sufficiently large CPU processing capability of the ES, our algorithm bypasses the local execution of the task in the action set. Moreover, given a sufficient CPU processing capability of the local device, ESs that are situated far away from the user are omitted from the action set. Combining these principles with the privacy protection scheme, the proposed algorithm successfully minimizes the latency.

- We conducted a simulation using the EUA dataset (Lai et al., 2018). Several competitive algorithms were selected for comparative purposes, including the random offloading algorithm, policy-iteration-based algorithm (Wang et al., 2020), and genetic algorithm (Wang et al., 2023). The results demonstrate that the proposed algorithm outperforms comparative algorithms in minimizing delay under the consideration of privacy preservation. Subsequently, we further discussed the potential improvements of PPDO.

The remainder of this paper is organized as follows. Section 2 describes the system model and formulates the problem. The proposed PPDO algorithm is described in detail in Section 3. In Section 4, we present a simulation and compare the proposed algorithm's performance with that of existing algorithms. Subsequently, we present a further discussion of the improvements yielded by PPDO. Finally, we conclude the paper and consider future directions of research in Section 5.

2 Problem formulation

2.1 System model

Fig.1. depicts the overall model of a CEC system, where $M = \{1, 2, ..., m\}$ and $N = \{1, 2, ..., n\}$ denote the number of mobile users (MUs) and ESs, respectively. Each mobile terminal generates multiple computing tasks $U = \{u_1, u_2, ..., u_k\}$. where each attribute of a task consists of a tuple $u_i = \{c_i, s_i\}(i \in U)$. Here s_i is the task's data volume, which remains constant during task processing, and c_i is the number of CPU cycles required for the task. Letting δ_i be the number of cycles per data unit, c_i can be computed as $s_i \times \delta_i$. The MU connects to the ES through the network and uses a binary offloading mode; thus, each task can only be processed locally by the MU or offloaded to an ES. The notation used is listed in Table 1.

Table 1 Explanation for the notations

Notation	Definition
M	Number of MU*
N	Number of ES*
U	Number of task
f^{local}	Local CPU capability
s^{local}	Mobile device storage capacity
f^{ES}	ES CPU capability
s^{ES}	ES storage capacity
s_i	Data size of task i
c_i	CPU cycle number of task i
δ_i	CPU cycle number per unit data size of task i
d_{ij}	Distance between user i and ES j
$s_j^{privacy}$	Data size of privacy task j
$c_j^{privacy}$	CPU cycle number of privacy task j
${\delta_j}^{privacy}$	CPU cycle number per unit data size of privacy task j
X	Offloading decision
B_w	Bandwidth
p	transmission power
σ	Noise power
A_s	Small-scale attenuation factor
B	Channel fading factor
h_i	Channel gain

* ES and MU denote edge servers and mobile users, respectively.

unedited

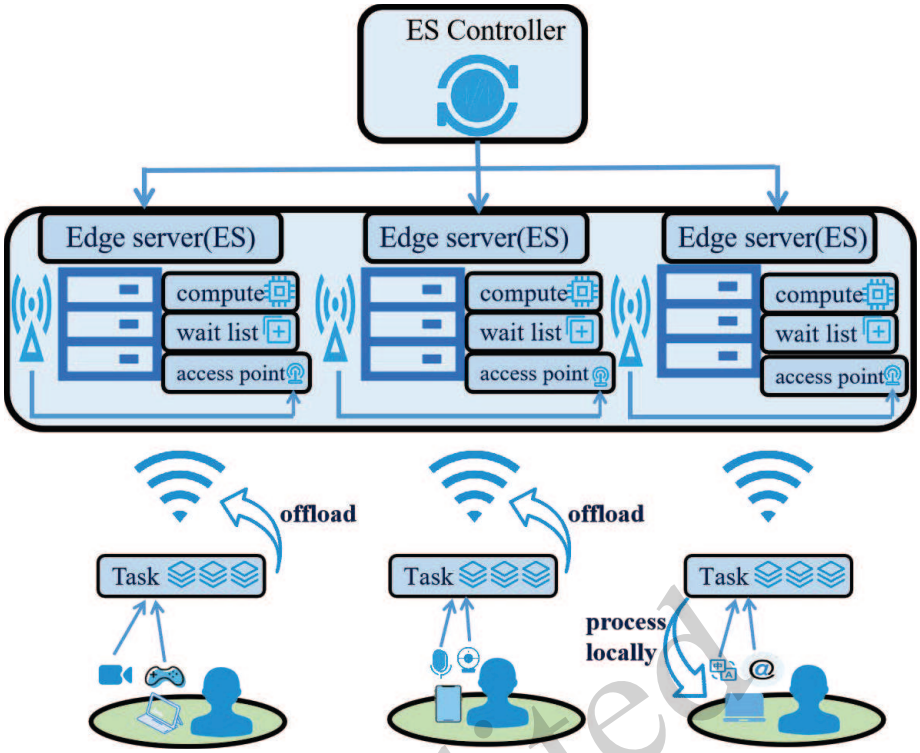


Fig. 1 Illustration of CEC system model with multiple users.

unedited

2.1.1 Local computing model

Tasks generated by terminal devices of the MU int CEC can be divided into local execution and offload to ES. However, tasks cannot be split during execution.

The set of offloading decision variables for the task is $X = \{x_{a_i} | i = 1, 2, ..., |U|\}$, where $x_{a_i} \in \{0, 1\}$. If $x_{a_i} = 0$, the task is processed at the local device; otherwise, it is offloaded to the ES for processing.

Locally executed tasks are only subject to delays from computation and queuing, as no transmission delay is incurred. The CPU frequency of the local device is $f_{m_i}^{local}(m_i \in M)$ and the is incurred storage capacity is $s_{m_i}^{local}$. The delay incurred by local execution can be derived from Equation (1):

$$t_{m_i}^{local} = \frac{c_i}{f_{m_i}^{local}} \quad (1)$$
$$s.t. \sum\nolimits_{i \in m} (1 - x_{a_i}) s_i \leq s_{m_i}^{local}$$

The constraint indicates that the sum of locally executed task data is less than the device storage capacity. When this threshold is exceeded, any subsequent tasks must be queued. The queuing delay is the sum of the execution delays from the previous task, calculated by Equation (2):

$$t_{q_i}^{local} = \sum_{i \in m} t_{m_i}^{local} \quad (2)$$

Therefore, the total delay in local task execution can be computed by Equation (3):

$$t_{sum}^{local} = \sum_{i=1}^{m} (t_{m_i}^{local} + t_{q_i}^{local}) \quad (3)$$

2.1.2 Edge server computing model

User devices connect to the ES through the network, and the upload processes of tasks are considered during network transmission. In the model specifications, B_w denotes the channel bandwidth of the network transmission, p indicates the transmission power, σ denotes the noise of power, A_s is used as a small-scale attenuation factor, B represents the channel fading factor, and d_{ij} denotes the communication distance between the user device i and the

ES j.Thus, the channel gain between i and j can be computed using Equation (4):

$$h_i = A_s d_{ij}^{-B} \tag{4}$$

Let $ph_{i'}$ represent the interference generated by other MUs in the network on MU i. Based on the Shannon formula, the user device i and transmission rate of ES j can be calculated by Equation (5):

$$\underset{i\in m, j\in n}{v_{ij}} = B_w \log_2(1 + \frac{ph_i}{\sum\limits_{\substack{i'\in m,\\ i'\neq i}} ph_{i'} + \sigma^2}) \tag{5}$$

Therefore, if the tasks generated by the user device are offloaded to a certain ES j, the task data must be transmitted to the server through the network. The transmission delay is defined by Equation (6):

$$t_{n_i}^{tran} = \sum\nolimits_{i\in m} x_{a_i} \frac{s_i}{v_{ij}} \tag{6}$$

Following the data transfer, the ES begins to compute the task, where $f_j^{ES}(j \in N)$ and s_j^{ES} denote the processing and storage capacities of the ES, respectively. The computational delay of the task offloaded to the ES can be obtained by Equation (7):

$$t_i^{ES} = \frac{c_i}{f_j} \tag{7}$$
$$s.t. \sum\nolimits_{i\in m} x_{a_i} s_i \leq s_j^{ES}$$

When the amount of task data offloaded to the ES exceeds its current storage capacity, any forthcoming tasks must be queued. The queuing delay, represented by the sum of the task execution delays of other users who have already entered the processing queue, is obtained by Equation (8):

$$t_{q_i}^{ES} = \sum_{i\in m} t_i^{ES} \tag{8}$$

The total delay from task offloading to ES execution can then be calculated as Equation (9):

$$t_{sum}^{ES} = \sum_{i=1}^{n} (t_{n_i}^{tran} + t_i^{ES} + t_{q_i}^{ES}) \tag{9}$$

2.1.3 Privacy information

A. Risk of privacy leakage

The privacy of location relates to the distance between users and servers. Generally, for the further distance, the risk of userâĂŹs location privacy leakage will be lower (Hua et al., 2023). According to (Wang et al., 2020), we can obtain the risk of user's location privacy leakage by Equation (10):

$$l = \begin{cases} 0 & , d_{ij} > d_{thr}, \\ a_l + b_l h^{d_{ij}} & , d_{ij} \leq d_{thr}. \end{cases} \tag{10}$$

where a_l , b_l and h are constants. According to Equation (10), l relates to d_{ij} and d_{thr}. To highlight these key factors, we reformulate Equation (10) by integrating $\frac{d_{ij}}{d_{thr}}$; thus, the risk of location privacy leakage can be redefined as Equation (11) in this work:

$$l_l = \begin{cases} 0 & , d_{ij} > d_{thr}, \\ a_l + b_l h^{\frac{d_{ij}}{d_{thr}}} & , d_{ij} \leq d_{thr}. \end{cases} \tag{11}$$

In CEC, it is equally important to protect usage pattern privacy, users exhibit different usage patterns when running applications on their devices (Zhao et al., 2018). According to the previous equations, users who strictly satisfy $d_{ij} > d_{thr}$, their location privacy risk is 0. It means that there is no location privacy leakage problem. For these users it is imperative to prioritize the privacy of usage pattern. Accordingly, the risk of usage pattern privacy can be defined by Equation (12):

$$l_u = \begin{cases} a_l + b_l h^{\frac{d_{ij}}{d_{thr}}} & , d_{ij} > d_{thr}, \\ 0 & , d_{ij} \leq d_{thr}. \end{cases} \tag{12}$$

Conversely, we prioritize location privacy for any users satisfying $d_{ij} \leq d_{thr}$.

B. Privacy task

In our framework, all generated privacy tasks are offloaded. The number of privacy tasks is determined by the privacy risks l_l and l_u. Each privacy task has the same attributes (e.g., data size and number of cycles) as a normal task on the user's device; however, the value of each attribute is less than that in a normal task. Because each privacy

unedited

task interferes with the ES, higher attribute values may significantly impact delay.

To protect location privacy, mobile devices deliberately offload tasks under poor network conditions (Zhao et al., 2023). Therefore, the user's device is added to the privacy task queue and the tasks are offloaded to the previous ES when the user switches service. To protect privacy of usage pattern, privacy tasks are added to the user's normal task queue to perturb the presumption of the ES regarding the usage pattern. Fig.2. illustrates the process of privacy tasks from addition to offloading. The specific privacy-preserving scheme is described in Section 3.2.

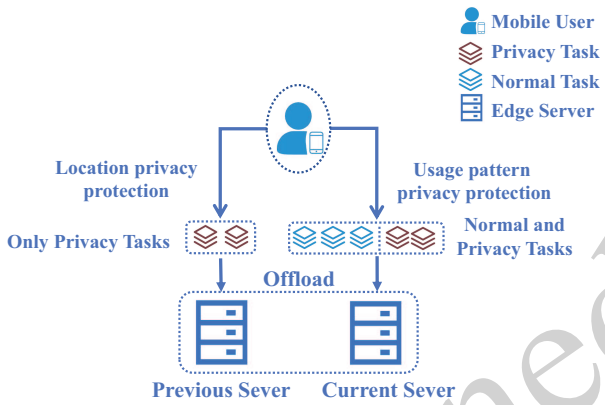


Fig. 2 Illustration of the privacy task

Let $s^{privacy}$ and $c^{privacy}$ denote the data size and number of cycles required by the privacy task, respectively. The transmission delay of the privacy task can be obtained using Equation (13):

$$t_{n_i}^{privacy_tran} = \frac{s^{privacy}}{v_{ij}} \tag{13}$$

All privacy tasks are offloaded to the ES for execution, and the processing delay is defined by Equation (14):

$$t_{n_i}^{privacy} = \frac{c_i^{privacy}}{f_j} \tag{14}$$
$$s.t. \sum\nolimits_{i\in m} s_i^{privacy} \le s_j^{ES}$$

When the task data received by the ES exceed the current storage capacity, privacy task processing must be queued. The corresponding queuing delay can be computed by Equation (15):

$$t_{q_i}^{privacy} = \sum_{i\in m} t_i^{privacy} \tag{15}$$

The cumulative delay incurred by privacy tasks is obtained by Equation (16).

$$t_{sum}^{privacy} = \sum_{i=1}^{n} (t_{n_i}^{privacy_tran} + t_{n_i}^{privacy} + t_{q_i}^{privacy}) \tag{16}$$

2.2 Problem statement

Combining the established CEC model, with the processing models for normal and privacy tasks, the total delay becomes the sum of all task delays, including those associated with execution processes and privacy tasks. The objective is to minimize the total delay, as defined by Equation (17).

$$P1: \min(T_{sum}) = t_{sum}^{local} + t_{sum}^{ES} + t_{sum}^{privacy} \tag{17}$$
$$s.t. \sum\nolimits_{i\in m} (1 - x_{a_i}) s_i \le s_{m_i}^{local},$$
$$\sum\nolimits_{i\in m} x_{a_i} s_i \le s_j^{ES}.$$

The first constraint denotes the tasks to execute locally while the local capacity is sufficient. The second constraint indicates the tasks to be offloaded to a certain ES if it has sufficient capacity.

3 Algorithm design

3.1 Overview

The following subsections describe the algorithm for integrating privacy protection and latency optimization in multi-user CEC systems. The privacy protection component offloads privacy tasks encompassing location and usage pattern privacy, whereas the proposed policy-iteration algorithm optimizes latency.

3.2 Privacy task-based privacy preservation

To protect the users' location and usage pattern privacy during task offloading in CEC, we implemented privacy tasks, which are offloaded to interfere with the ES responsible for task execution, inhibiting it from analyzing privacy information. Fig.3. illustrates the risk of location and usage pattern privacy leakage.

unedited

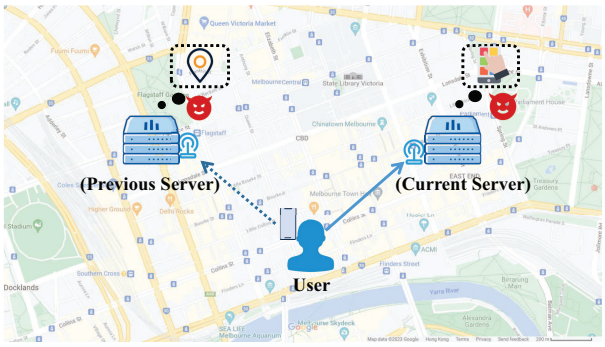


Fig. 3 The location and usage pattern privacy leakage Risks

- **Location privacy protection**: ESs can infer user location information through offloaded tasks, as mobile users in MEC may switch servers or execute tasks locally under poor network conditions (Min et al., 2019) (Bai et al., 2020). To solve this problem, our algorithm generates a list of all privacy tasks, with their number determined by the privacy leakage risk l_l and number of normal tasks u_k, denoted as $\lfloor l_l \times u_k \rfloor$. Consequently, the ES continues to function as if it is receiving offloaded tasks from the user, and therefore cannot infer the user's location information. However, the user is subject to the delay stemming from the privacy tasks.

- **Usage pattern privacy protection**: The leakage of usage pattern privacy can be primarily attributed to the ES evaluating data associated with offloaded tasks (Min et al., 2019) (Zhao et al., 2023). To protect the privacy of the usage pattern, we added privacy tasks to the task list offloaded to the ES, with the number of privacy tasks denoted as $\lfloor l_u \times u_k \rfloor$. These privacy tasks interfere with the ES during the evaluation of task data, thereby protecting the usage pattern. Similarly, the user is subject to delay stemming from the privacy tasks.

Fig.4. illustrates the general privacy protection procedure for mobile users. Although both types of privacy are preserved by offloading privacy tasks, the specific details differ between the two. The process is described in Algorithm 1.

3.3 MDP-based offloading algorithm

The following subsections describe the integration of the transformed CEC model into an MDP model, as well as problem design under this model, followed by solving the latency optimization problem stemming from task execution using the policy iteration algorithm.

Algorithm 1 privacy-task-based privacy preservation-aware scheme

Initialize $u_k, l_l, l_u,$ for all i =1,2,...,$m(i \in M)$ // initialize privacy leakage risk $l_l, l_u,$, task number u_k
/***Location privacy preservation***/
loop
  for all $i \in M$ **do**
    if $l_l > 0$ **then**
      Generate of $\lfloor l_l \times u_k \rfloor$ privacy tasks
      Set new task list by $\lfloor l_l \times u_k \rfloor$ privacy tasks
    end if
  end for
  if user $i(i \in M)$ switched the ES **then**
    Offload privacy tasks to previous ES
  end if
end loop
/***Usage pattern privacy preservation***/
loop
  for all $i \in M$ **do**
    if $l_u > 0$ **then**
      Generate of $\lfloor l_u \times u_k \rfloor$ privacy tasks
      Add $\lfloor l_u \times u_k \rfloor$ privacy tasks to task list
      Offload task list to current ES
    end if
  end for
end loop

3.3.1 MDP model

An MDP model typically includes a state set, action set, system transfer probability, and reward function. We used an iteration strategy to determine the best value function that solves the MDP problem and obtains the user's offloading decision. For the state set, the state space is defined as: $S = \{s|s = (d, q)\}$, where d denotes the distance between the user device and each ES, and q denotes the task processing queue of the ES. The CEC system defines actions associated with the users' selection of ES nodes, with the action set defined as $A = \{a|a_1, a_2, a_3, ..., a_n\}$, where $a(a \in A)$, For example, $a = 1$, indicates that the user offloads a task to the ES labeled 1. Each policy corresponds to an action in action set A, denoted by $\pi(s) \in A(s)$.

The state transfer probability of the CEC system is related to distance d and queue q in the state set. The distance transfer function is shown in Equa-

unedited

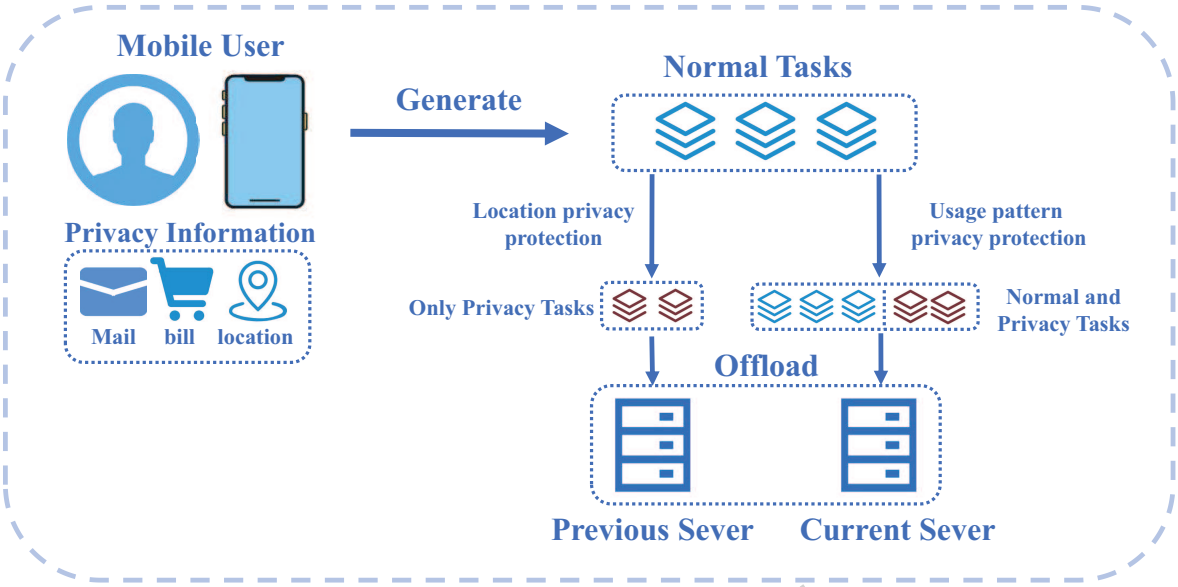


Fig. 4 Illustration of general procedure on privacy task based privacy preservation

unedited

tion (18):

$$P(d|d',a) = \frac{c_1}{nd'} \tag{18}$$

where n is the number of ESs, d' denotes the distance between the ES to be selected and the user device, and c_1 is a constant. For CEC systems with a fixed number of ESs, a smaller distance between the user device and ES, indicates a higher transfer probability.

The queue transfer function, which relates to the queue state of the edge device, is shown in Equation (19):

$$P(q|q',a) = \frac{c_2}{q_t} \tag{19}$$

where q_t is the number of tasks queued by the edge device and c_2 is a constant. Similarly, a smaller number of tasks in the queue ensures a greater probability of queue transfer. Letting s' be the state immediately following state s and d' be equivalent to the distance d between the user device and ES, if d and the transfer probability of the task processing queue q of the ES are mutually independent, the transfer probability of the system is obtained by Equation (20):

$$P(s|s',a) = P(d|d',a)P(q|q',a) \tag{20}$$

In the MDP model developed in this study, the reward function is defined as the weighted sum of the time delays generated following the selection of an ES expressed as Equation (21):

$$r = \omega \times t_{sum} \tag{21}$$

where $\omega > 0$ is a weighting factor that takes values within [1,10] and t_{sum} is the total delay incurred by task execution.

In summary, the value function of the task under the MDP model is formulated as Equation (22):

$$V(s,a) = r + \gamma \sum_{s'} P(s|s',a)V(s',a) \tag{22}$$

where $\gamma \in [0,1]$ is the discount factor that indicates the degree of influence of the future state on the current state. Because the reward r is proportional to the task delay, an optimal solution to this problem must minimize the value function, as in Equation (23).

$$P2: \min_{a\in A, s\in S} V(s,a) \tag{23}$$

Observing P2, for the transformed MDP problem, an algorithm must be used to solve the minimum value function $V(s,a)$, where $V(s,a)$ converges to the policy $\pi(s)$ and then remains constant, and

its action $a(s)$ is the node identified as the optimal offloading decision for the user in the CEC.

3.3.2 Policy-iteration-based privacy preservation-aware offloading algorithm

In this subsection, we propose the PPDO privacy privacy-preservation-aware algorithm, which generates offloading decisions for users with the objective of reducing the latency of task offloading. We identified the delay incurred by user task offloading and privacy protection in CEC systems. The algorithm first initializes the selection of the value function $V(s)$ and policy $\pi(s)$ for each user in the context of conversion to MDP, using a random policy for the initial action selection, and the value function uses the reward r for the initial action in the initialization phase. Subsequently, the policy evaluation phase initiates to compute the value function $V(s)$ for a given strategy with the objective of converging $V(s)$. Finally, there is the policy improvement phase mainly focuses on the selection of $\pi(s)$ based on the difference in $V(s)$. Because r is proportional to the total delay of task offloading in this paper, smaller $V(s)$ is required in this phase. An optimal policy is generated when $\pi(s)$ no longer changes in this phase, and the algorithm terminates. The overall process is presented in Algorithm 2.

3.3.3 Analysis of time complexity

In Algorithm 2, the convergence of PPDO relates to the value function $V(s)$. During the policy evaluation process, the computation of $V(s)$ in each state requires the system transfer probability and value functions to be mutually independent. If the algorithm satisfies $\Delta<\theta$ within a smaller range of constants, the optimal time complexity of this process is $O(n^2)$. For policy improvement, it is necessary to select the action and compute $V(s)$. Generally, the time complexity of the process is $O(n^3)$. Overall, the time complexity of the algorithm 2 is $O(n^3)$.

3.4 Improvement of PPDO

In the MDP model established above, because the CEC system must engage in the collaborative computation of local devices and ESs, we implemented privacy tasks to guarantee the protection of privacy. This leads to exploration with a high temporal overhead during policy iteration.

Algorithm 2 privacy preservation-aware delay optimization task-offloading algorithm

1: **Input:** S, A, P // initialize state set S,action set A and probability set P
2: **Output:** $\pi(s)$ // Output optimal strategy $\pi(s)$
3: Randomly set $\pi(s)$ and $V(s)$ for all $s \in S$
4: **/*Policy evaluation*/**
5: **loop**
6: $\Delta \leftarrow 0$
7: **for** each $s \in S$ **do**
8: $v \leftarrow V(s)$
9: $V(s) \leftarrow r + \gamma \sum_{s'} P(s|s',a)V(s',a)$
10: $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
11: **end for**
12: **end loop** ($\Delta > \theta$ (a small positive number))
13: **/*Policy improvement*/**
14: policy_flag $\leftarrow$ true
15: **for** each $s \in S$ **do**
16: last_action $\leftarrow \pi(s)$
17: $\pi(s) \leftarrow \arg\min_a [r + \gamma \sum_{s'} P(s|s',a)V(s',a)]$
18: **if** last_action $\neq \pi(s)$ **then**
19: policy_flag $\leftarrow$ false
20: **end if**
21: **if** policy_flag **then**
22: **return** $V \leftarrow v_*, \pi \leftarrow \pi_*$
23: **else**
24: go to **4**(Policy evaluation)
25: **end if**
26: **end for**

To address this issue, we accelerated the policy iteration by reducing the action set A. Specifically, given a sufficiently small number of tasks, the tasks of the user devices can be completely offloaded to the ES for processing, and the selection of local devices in A can be reduced. Similarly, when the CPU processing capability of the ES is sufficiently large, many tasks are offloaded to the ES for processing. Thus, the selection of local devices in A decreases. In addition, when the CPU processing capability of the local device increases, the local device and ES adopt collaborative computation to execute tasks, while some farther distant ESs are not selected by the user. In this case, the ESs farther from this user can be removed from A. As the size of A decreases, the time complexity of the algorithm is reduced to $O(n^2)$.

The optimization component of PPDO is detailed in Algorithm 3. In lines 5-6, if the number of tasks is sufficiently small or the CPU processing capability of the ES is sufficiently large, the action

unedited

Algorithm 3 improved policy iteration algorithm

1: **Input** action set A, task number u_k, edge CPU capability f^{ES}, local CPU capability f^{local}.
2: **Output** A // Output updated action set A
3: **loop**
4: **for** all $a \in A$ **do**
5: **if** $u_k <= u_{thr}$ or $f^{ES} >= f^{ES}_{thr}$ **then**
6: Remove action a for local execution of tasks
7: **end if**
8: **if** $f^{local} >= f^{local}_{thr}$ **then**
9: Remove action a for $d_{ij} > d_{thr}$
10: **end if**
11: **end for**
12: **return** $A \leftarrow A^*$
13: **end loop**

a that executes the task locally is removed from A. Likewise, in lines 8-9, when the CPU processing capability of the local device is sufficiently large, actions are removed from A as the corresponding tasks are offloaded. Simultaneously, the reduced A^* is updated to action set A. According to Equation (12), if the risk of usage pattern privacy leakage satisfies $l_u = 0$, privacy tasks associated with usage pattern privacy can be omitted. Therefore, Algorithm 3 can be implemented within Algorithm 2 to ensure the efficiency of selection operations.

4 Performance evaluation

4.1 Simulation setup and data preparation

The simulation was implemented in Java (based on JDK1.8) and the IntelliJ IDEA software, version IntelliJ IDEA 2019.3.3. The number of user devices (i.e., local devices) was alternated between 50 and 100. In addition, 30 edge servers were used in the experiment. To verify the effectiveness of PPDO, we considered the total delay metric, which encompasses the delay associated with both execution and privacy protection.

The parameters used in the simulations are listed in Table 2, where f^{local} is the CPU processing capability of the MU, ranging from 0.5GHz to 2GHz, and s^{local} is the storage capacity of the MU, set to 1000KB. The processing capability f^{ES} and storage capacity s^{ES} of the ES were set to (2GHz,4.5GHz) and (5000KB,10000KB) respectively. s_i is the data size of task i, set within (10KB,100KB).

c_i is the number of CPU cycles for task i, taking the value of $s_i \times \delta_i$, where δ_i is the number of cycles per data unit for a normal task i and $\delta_i = 0.8$. For the privacy task, the data size $s^{privacy}$ falls within (5KB,50KB) and the number of CPU cycles is computed as $c^{privacy} = s^{privacy} \times \delta^{privacy}$. To guarantee the effectiveness of privacy task, we tuned $\delta^{privacy}$ to 0.6, which was less than δ_i, to avoid augmenting the relative importance of privacy tasks. The network parameters are B_w, p, σ, A_s, and B.

Moreover, θ is a small positive number in Algorithm 2, and a_l, b_l, and h are constants in Equation (11). The location information of local devices and ESs was selected from the EUA dataset (Lai et al., 2018), containing the location data of mobile users and edge servers in Melbourne City CBD.

Table 2 Parameters setting

Parameters	Value
f^{local}	(0.5GHz,2.5GHz)
s^{local}	1000KB
f^{ES}	(2.5GHz,4.5GHz)
s^{ES}	(5000KB,10000KB)
s_i	(10KB,100KB)
δ_i	0.8 (Cycle number per unit data size)
c_i	$s_i \times \delta_i$
$s^{privacy}$	(5KB,50KB)
$\delta^{privacy}$	0.6 (Cycle number per unit data size)
$c^{privacy}$	$s^{privacy} \times \delta^{privacy}$
B_w	500MHz
p	200mW
σ	-100mW
A_s	2.5
B	0.5
θ	1.6
a_l	0
b_l	2
h	0.5

4.2 Comparative algorithms

The proposed PPDO algorithm was compared with the following existing algorithms:

- **Random offloading algorithm**: Without considering location distance, the algorithm randomly generates offloading decisions for user devices and then traverses all ES nodes to assign the offloading or local execution of tasks. The time complexity of this algorithm is $O(mn)$.
- **Genetic algorithm**: The genetic algorithm uses a chromosome structure to represent the offloading strategy for tasks in MEC. The algorithm generates chromosomes and subsequently

unedited

initiates a crossover mutation process. We set the chromosome set the chromosome size to 200, the crossover probability to 0.7, and the mutation probability to 0.2 (Wang et al., 2023). The time overhead relates to the population size. In general, the time complexity is bounded by $O(n^2)$.

- **Policy iteration based algorithm**: This algorithm solves the MDP problem using a policy iteration that generates offloading decisions for the user. It does not account for collaborative computing between the ES and local device in terms of a full offloading of tasks to the ES (Wang et al., 2020). Note that the two latter algorithms were combined with the proposed privacy-preserving scheme for comparison.

4.3 Comparison results

To verify the effectiveness of the proposed algorithm, we initially conducted a comparison in terms of the task number, edge CPU, and local device CPU to minimize the total delay. Next, we investigated the effect of the distance threshold on the risk of privacy leakage. Subsequently, a comparative simulation of the impact of privacy protection was conducted. Finally, the impact of the discount factor on algorithm performance was examined via simulation.

4.3.1 Evaluation of total delay

We compared the total delay incurred by each algorithm after implementing our privacy protection scheme. Three groups of comparison simulations were set up in terms of the number of tasks, CPU processing capability of the ES, and CPU processing capability of the local device. Fig.5 illustrates the variation of total delay in the CEC under different conditions.

From Figs.5a, we see that as the number of tasks generated by the mobile device increases, the total delay grows in all four algorithms, as random offloading cannot be rationalized in a collaborative computing system. The total delay of the GA algorithm was less than that of the randomized algorithm and greater than that of both MDP-based algorithms. Furthermore, the total delay under the PPDO algorithm became lower than that under (Wang et al., 2020) as the number of tasks increased. This can be attributed to the ability of PPDO to utilize local devices for collaborative computation.

As shown in Fig.5b, following 30 tasks, the total delay incurred by the four algorithms decreased as the CPU processing capability of the ES increased. For higher processing capabilities, the policy-iteration-based algorithm (Wang et al., 2020) and PPDO exhibited similar performance. The PPDO offloads most tasks to the ES for execution when the CPU processing capability of the ES is sufficiently high. The genetic algorithm utilizes collaborative computation between the local device and ES when the CPU of the ES is sufficiently small, incurring a lower total delay. Owing to its random characteristics, the random offloading algorithm obtained the worst results.

Figs.5c illustrates the performance results for 30 tasks with respect to the CPU processing power of the local device. The total delay exhibited a decreasing trend for the random offloading, genetic, and PPDO algorithms. In contrast, the algorithm of (Wang et al., 2020) could not take advantage of the local CPU processing capabilities, as it offloaded all tasks to the ES. When the chromosome of the genetic algorithm is set to zero, the tasks are executed on the local device (Wang et al., 2023). Similarly, the PPDO algorithm may allow users to execute tasks at the local device to realize collaborative computation. Consequently, the total delay incurred by PPDO is less than that incurred by the genetic algorithm.

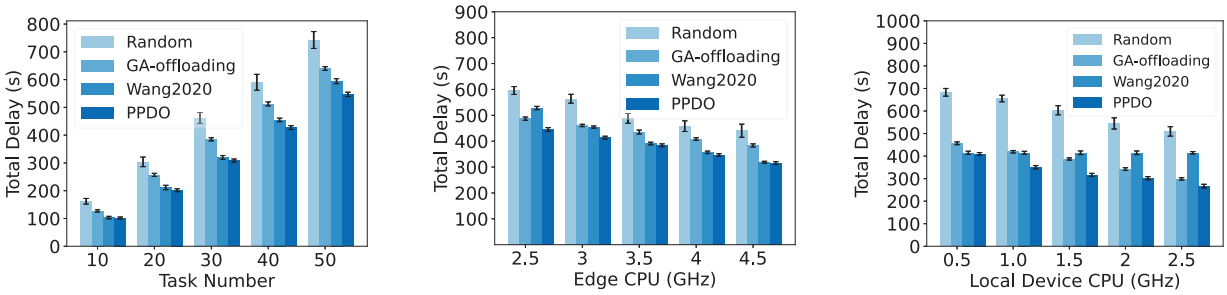
Overall, the MDP-based policy iteration algorithm outperformed all other algorithms with respect to the number of tasks, the CPU processing capability of the ES, and processing capability of the local device. The PPDO algorithm achieves optimal performance by taking advantage of collaborative computation, thereby minimizing total delay.

4.3.2 Impact on risk of privacy leakage

The privacy protection scheme proposed in this study offloads privacy tasks to protect user privacy, with the number of privacy tasks determined by the risk of privacy leakage. According to subsection 2.1.3, the distance threshold d_{thr} which affects the risk of privacy leakage, can be set to the average distance d_{avg} between the mobile device and all ESs, the closest distance d_{min}, or the farthest distance d_{max}. Fig.6 depicts the risks of privacy leakage calculated according to Equations (11) and (12) under different values of d_{thr}.

Fig.6a shows the leakage risk of usage pattern

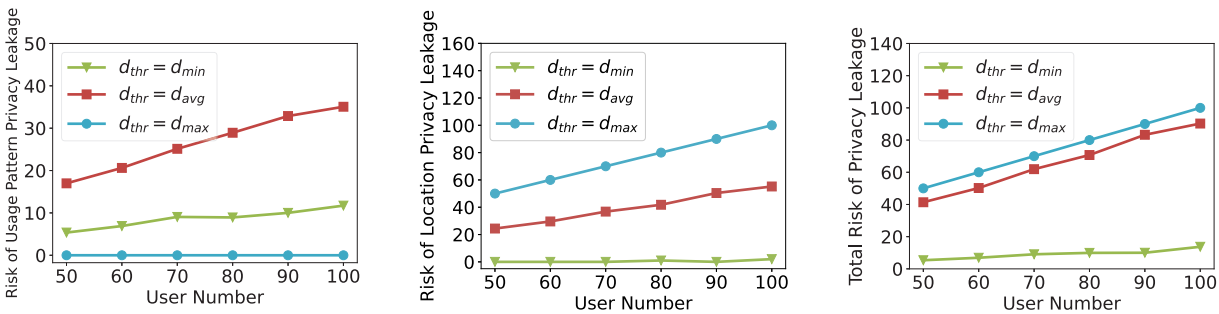
unedited



(a) (b) (c)

Fig. 5 The impact of the number of tasks, CPU of the ES, and CPU of the local device on total delay, second(s)

unedited



(a) (b) (c)

Fig. 6 Risks of privacy leakage under different d_{thr}

privacy l_u for different d_{thr}. As the number of users increases, l_u is observed to increase for both $d_{thr} = d_{min}$ and $d_{thr} = d_{avg}$. According to Equation (12), l_u is always 0 at $d_{thr} = d_{max}$. The risk of location privacy leakage l_l for different d_{thr} is shown in Figs.6b. For $d_{thr} = d_{avg}$ and $d_{thr} = d_{max}$, l_l increases with the number of users, and l_l reaches its maximum (i.e., 100%) for all users at $d_{thr} = d_{max}$, for d_{min}, most of the risk values are close to 0. It is because the risk of location privacy exists only for individual users with d_{ij}=d_{min}. Fig.6c illustrates the total risk value of privacy leakage. When $d_{thr} = d_{max}$, all users have only location privacy leakage risk, while $d_{thr} = d_{min}$, the leakage risk is mostly location privacy.

In general, the value of d_{thr} affects the risk of privacy leakage for both location and user pattern. To simultaneously investigate the leakage risks for both types of privacy, we adopted $d_{thr} = d_{avg}$ for the simulations.

4.3.3 Impact of privacy-preservation

To examine the impact of privacy protection on the total delay, we considered a scenario that considers the protection of each type of privacy alone under different numbers of tasks. To avoid the impact of device parameters on results, we set the CPU capabilities of the local device and ES at maxima of 2.5GHz and 4.5GHz, respectively.

Table 3 presents the total delay comparison of the three algorithms with location privacy protection, showing that the PPDO algorithm incurs the smallest total delay. Local devices only consider the risk of location privacy leakage l_l and add additional privacy task queues for mobile devices accordingly. Although the total delay incurred by each algorithm increased with the number of tasks, it remained less than that incurred when both privacy protection schemes were implemented, as fewer privacy tasks were generated.

Table 4 presents a delay comparison of the three algorithms solely under usage pattern privacy protection. Mobile devices only take into account their risk of usage pattern privacy leakage l_u and add privacy tasks to the user's device task queue based on l_u. Similarly, although the total delay incurred by each algorithm increased with the number of tasks, it remained less than that incurred when both privacy protection schemes were implemented. The PPDO algorithm also obtains the least total delay when only utilizing the usage pattern privacy protection.

Table 3 Total delay with location privacy protection, second(s)

Task Number	Total Delay(s)		
	GA	Wang2020	PPDO
10	125.41	101.39	98.81
20	249.88	205.99	200.59
30	377.88	329.15	304.24
40	505.84	435.21	421.99
50	625.86	545.97	520.81

Table 4 Total delay with usage pattern privacy protection, second(s)

Task Number	Total Delay(s)		
	GA	Wang2020	PPDO
10	122.91	98.57	95.36
20	247.93	198.88	194.76
30	370.69	305.18	301.08
40	495.96	433.60	412.65
50	618.86	546.48	528.69

From the above results, we observe that the impact of the two privacy-protection schemes exert varying impacts on total delay. with location privacy protection resulting in greater total delay. However, the PPDO algorithm balances the total delay irrespective of the privacy protection scheme.

4.3.4 Impact of different γ

Because the PPDO algorithm was primarily built on the MDP model, the discount factor γ in Equation (22) is proportional to the impact of the current decision on the future state. Accordingly, we varied the setting of γ to evaluate its effect on algorithm performance. For this experiment, we set the task number to 30, CPU capability of the local device to 2.5GHz, and CPU capability of the ES to 4.5GHz.

As shown in Fig.7, as γ grows from 0 to 1, the total delay of the PPDO algorithm is minimized at $\gamma = 0.9$. This indicates that PPDO relies more on the future state than the current state when computing $V(s, a)$. In our simulation, the PPDO algorithm yields optimal performance at $\gamma = 0.9$.

4.4 Discussion of the improvement of PPDO

We previously mentioned that the proposed policy iteration algorithm may incur a high time overhead. To investigate the performance improvement in PPDO, we extended our simulation to the time

unedited

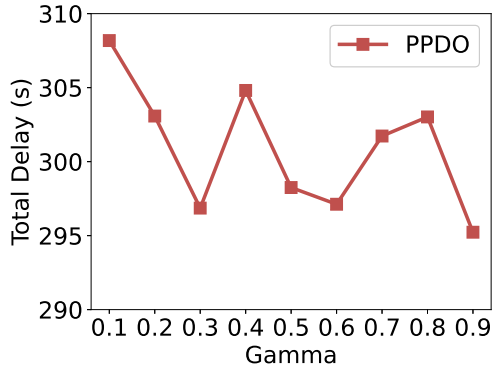


Fig. 7 Impact on different γ, second(s)

overhead in terms of running time. We set the user device number to 50, task number to 10, CPU processing capability of the ES to 4.5GHz, and local device CPU to 2.5GHz.

Figure 8 presents the running times obtained by the two algorithms. Owing to the reduction in action set A, the traversal of local devices and ESs was reduced during policy evaluation, thereby improving the iterative process. Under these conditions, there was a significant decrease in running time.

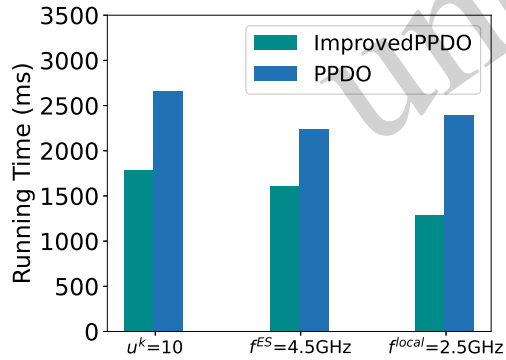


Fig. 8 Comparison results in terms of running time, millisecond(ms)

However, because the improved PPDO algorithm no longer includes the action of the local device, it becomes equivalent to that in (Wang et al., 2020), which offloads all tasks to the ES when the task number is sufficiently small (10) or the capability of the ES is sufficiently large (4.5GHz). Furthermore, if the local device CPU is maximized at 2.5GHz, the ESs that satisfy the $d_{ij} > d_{thr}$ distance requirement are removed from A, leaving fewer available ESs. Because the risk of privacy leakage of the user's usage pattern privacy $l_u = 0$ is calculated according to Equation (12), the number of privacy tasks is reduced, resulting in a change in the total delay.

Figure 9 presents a comparison of total delay between the improved PPDO and PPDO. Under certain conditions, when the number of tasks is small and the device's CPU processing capability is high, the improved PPDO has the enhancement in terms of running time compared to PPDO, while the total delay is slightly higher than that of PPDO.

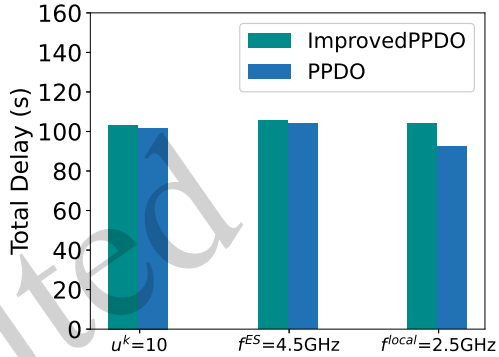


Fig. 9 Comparison of total delay between improved PPDO and PPDO, second(s)

unedited

5 Conclusion and future work

This study primarily investigated the balance between privacy protection and delay in CEC systems. In contrast to prior studies, we investigated the problem of privacy preservation with the objective of minimizing the total delay in task offloading. The preservation scheme calculates the privacy leakage risk and offloads privacy tasks accordingly. Subsequently, the issue of minimizing the delay incurred by task offloading was formulated as an MDP problem. We developed a policy-iteration-based privacy preservation-aware offloading algorithm to decrease the total latency of task offloading while protecting user privacy. Furthermore, we extended PPDO by decreasing the exploration time in the action set. The results show that the proposed algorithm minimizes the delays without compromising privacy protection while offloading the tasks. The complexity of the improved PPDO was compared and discussed to optimize performance in terms of running time.

In future work, we will consider the use of a

changeable network channel model in CEC. Furthermore, to fit into the real-time scenario, it is important to integrate the concept of adaptiveness into the current algorithm. This can be achieved by incorporating reinforcement learning to find a trade-off between privacy protection and delay.

Contributors

Chao JING guided the research, revised and finalized the paper. Jianwu XU designed the research, organized and drafted the paper.

Compliance with ethics guidelines

Chao JING and Jianwu XU declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are not openly available as the data also forms part of an ongoing study.

References

Abbas N, Zhang Y, Taherkordi A, et al., 2018. Mobile edge computing: A survey. *IEEE Internet Things J*, 5(1):450-465. https://doi.org/10.1109/JIOT.2017.2750180

Bai Y, Chen LX, Song LQ, et al., 2020. Risk-aware edge computation offloading using bayesian stackelberg game. *IEEE Trans Netw Serv Manage*, 17(2):1000-1012. https://doi.org/10.1109/TNSM.2020.2985080

Cao T, Qian ZZ, Wu K, et al., 2021. Service placement and bandwidth allocation for mec-enabled mobile cloud gaming. *Proc 22nd Int Symp on a World of Wireless, Mobile and Multimedia Networks*, p.179-188. https://doi.org/10.1109/WoWMoM51794.2021.00031

Chen SG, Zheng YM, Lu WF, et al., 2020. Energy-optimal dynamic computation offloading for industrial iot in fog computing. *IEEE Trans on Green Commun Netw*, 4(2):566-576. https://doi.org/10.1109/TGCN.2019.2960767

Chu WB, Jia XM, Yu ZW, et al., 2023. Joint service caching, resource allocation and task offloading for mec-based networks: A multi-layer optimization approach. *IEEE Trans on Mobile Comput*, in press. https://doi.org/10.1109/TMC.2023.3268048

Dong LB, Gao HH, Wu WL, et al., 2023. Dependence-aware edge intelligent function offloading for 6G-based IoV. *IEEE Trans Intell Transport Syst*, 24(2):2265-2274. https://doi.org/10.1109/TITS.2022.3148229

Gao HH, Huang WQ, Liu T, et al., 2023b. PPO2: Location privacy-oriented task offloading to edge computing using reinforcement learning for intelligent autonomous transport systems. *IEEE Trans Intell Transport Syst*, 24(7):7599-7612. https://doi.org/10.1109/TITS.2022.3169421

Gao HH, Wang XJ, Wei W, et al., 2023a. Com-DDPG: Task offloading based on multiagent reinforcement learning for information-communication-enhanced mobile edge computing in the internet of vehicles. *IEEE Trans Veh Technol*, in press. https://doi.org/10.1109/TVT.2023.3309321

He T, Ciftcioglu EN, Wang SQ, et al., 2017a. Location privacy in mobile edge clouds. *Proc IEEE 37th Int Conf on Distributed Computing Systems,* p.2264-2269. https://doi.org/10.1109/ICDCS.2017.39

He XF, Liu J, Jin RC, et al., 2017b. Privacy-aware offloading in mobile-edge computing. *IEEE Global Communications Conf*, p.1-6. https://doi.org/10.1109/GLOCOM.2017.8253985

He XF, Jin RC, Dai HY, et al., 2020. Peace: Privacy-preserving and cost-efficient task offloading for mobile-edge computing. *IEEE Trans Wirel Commun*, 19(3):1814-1824. https://doi.org/10.1109/TWC.2019.2958091

Hua W, Zhou ZY, Huang LY, 2023. Location privacy-aware offloading for mec-enabled iot: Optimality and heuristics. *IEEE Internet Things J*, 10(21):19270-19281. https://doi.org/10.1109/JIOT.2023.3281609

Ksentini A, Taleb T, Chen M, 2014. A markov decision process-based service migration procedure for follow me cloud. *IEEE Int Conf on Communications,* p.1350-1354. https://doi.org/10.1109/ICC.2014.6883509

Lai P, He Q, Abdelrazek M, et al., 2018. Optimal edge user allocation in edge computing with variable sized vector bin packing. *Proc 16th Int Conf on Service-Oriented Computing*, p.230-245. https://doi.org/10.1007/9783030035969_15

Lee S, Lee S, Lee SS, 2021. Deadline-aware task scheduling for iot applications in collaborative edge computing. *IEEE Wirel Commun Lett*, 10(10):2175-2179. https://doi.org/10.1109/LWC.2021.3095496

Lin P, Song Q, Wang D, et al., 2021. Resource management for pervasive-edge-computing-assisted wireless VR streaming in industrial internet of things. *IEEE Trans Ind Inf*, 17(11):7607-7617. https://doi.org/10.1109/TII.2021.3061579

Mach P, Becvar Z, 2017. Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surv Tut*, 19(3):1628-1656. https://doi.org/10.1109/COMST.2017.2682318

Mao S, Liu L, Zhang N, et al., 2022. Reconfigurable intelligent surface-assisted secure mobile edge computing networks. *IEEE Trans Veh Technol*, 71(6):6647-6660. https://doi.org/10.1109/TVT.2022.3162044

Mao YY, Zhang J, Song SH, et al., 2017. Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems. *IEEE Trans on Wirel Commun*, 16(9):5994-6009. https://doi.org/10.1109/TWC.2017.2717986

Min MH, Wan XY, Xiao L, et al., 2019. Learning-based privacy-aware offloading for healthcare iot with energy harvesting. *IEEE Internet Things J*, 6(3):4307-4316. https://doi.org/10.1109/JIOT.2018.2875926

Ouyang T, Li R, Chen X, et al., 2019. Adaptive user-managed service placement for mobile edge computing: An online learning approach. *IEEE Conf on Computer*

unedited

Communications,p.1468-1476.
https://doi.org/10.1109/INFOCOM.2019.8737560

Qian LP, Wu WC, Lu WD, et al., 2021. Secrecy-based energy-efficient mobile edge computing via cooperative non-orthogonal multiple access transmission. *IEEE Trans Commun*, 69(7):4659-4677.
https://doi.org/10.1109/TCOMM.2021.3070620

Qin XD, Li B, Ying L, 2021. Distributed threshold-based offloading for large-scale mobile cloud computing. *IEEE Conf on Computer Communications,* p.1-10.
https://doi.org/10.1109/INFOCOM42981.2021.9488821

Ren JK, Yu GD, He YH, et al., 2019. Collaborative cloud and edge computing for latency minimization. *IEEE Trans Veh Technol*, 68(5):5031-5044.
https://doi.org/10.1109/TVT.2019.2904244

Sahni Y, Cao J, Yang L, 2019. Data-aware task allocation for achieving low latency in collaborative edge computing. *IEEE Internet Things J*, 6(2):3512-3524.
https://doi.org/10.1109/JIOT.2018.2886757

Taleb T, Dutta S, Ksentini A, et al., 2017. Mobile edge computing potential in making cities smarter. *IEEE Commun Mag*, 55(3):38-43.
https://doi.org/10.1109/MCOM.2017.1600249CM

Wang ZB, Pang XY, Chen YH, et al., 2019. Privacy-preserving crowd-sourced statistical data publishing with an untrusted server. *IEEE Trans Mobile Comput*, 18(6):1356-1367.
https://doi.org/10.1109/TMC.2018.2861765

Wang F, Xu J, Wang X, et al., 2018. Joint offloading and computing optimization in wireless powered mobile-edge computing systems. *IEEE Trans on Wirel Commun*, 17(3):1784-1797.
https://doi.org/10.1109/TWC.2017.2785305

Wang JD, Zhao L, Liu JJ, et al., 2021a. Smart resource allocation for mobile edge computing: A deep reinforcement learning approach. *IEEE Trans Emerg Top Comput*, 9(3):1529-1541.
https://doi.org/10.1109/TETC.2019.2902661

Wang TS, Li Y, Wu Y, 2021b. Energy-efficient UAV assisted secure relay transmission via cooperative computation offloading. *IEEE Trans Green Commun Netw*, 5(4):1669-1683.
https://doi.org/10.1109/TGCN.2021.3099523

Wang WX, Ge SX, Zhou XB, 2020. Location-privacy-aware service migration in mobile edge computing. *IEEE Wireless Communications and Networking Conf*,p.1-6.
https://doi.org/10.1109/WCNC45663.2020.9120551

Wang ZB, Sun YN, Liu DF, et al., 2023. Location privacy-aware task offloading in mobile edge computing. *IEEE Trans Mobile Comput*, in press.
https://doi.org/10.1109/TMC.2023.3254553

Yang XM, Luo H, Sun Y, et al., 2020. Energy-efficient collaborative offloading for multiplayer games with cache-aided MEC. *IEEE Int Conf on Communications,* p.1-7.
https://doi.org/10.1109/ICC40277.2020.9148751

Yousaf K, Mehmood Z, Saba T, et al., 2018. A novel technique for speech recognition and visualization based mobile application to support two-way communication between deaf-mute and normal peoples. *Wirel Commun Mobile Comput*, 2018:101323.
https://doi.org/10.1155/2018/1013234

Zhao P, Jiang HB, Lui JCS, et al., 2018. P^3-LOC: A privacy-preserving paradigm-driven framework for indoor localization. *IEEE/ACM Trans Netw,* 26(6):2856-2869.
https://doi.org/10.1109/TNET.2018.2879967

Zhao P, Tao JW, Lui K, et al., 2023. Deep reinforcement learning-based joint optimization of delay and privacy in multiple-user MEC systems. *IEEE Trans Cloud Comput*, 11(2):1487-1499.
https://doi.org/10.1109/TCC.2022.3140231

unedited