Hindawi
Wireless Communications and Mobile Computing
Volume 2019, Article ID 3154917, 13 pages
https://doi.org/10.1155/2019/3154917

Research Article

A Novel Task Allocation Algorithm in Mobile Crowdsensing with Spatial Privacy Preservation

Wenyi Tang,[1] Qi Jin,[1,2] Xu Zheng,[1] Guangchun Luo,[3] Guiduo Duan,[1] and Aiguo Chen[1]

[1]*The School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*
[2]*Chengdu Municipal Public Security Bureau, Chengdu 610017, China*
[3]*The School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China*

Correspondence should be addressed to Guangchun Luo; gcluo.uestc@gmail.com

Received 31 January 2019; Accepted 6 March 2019; Published 1 April 2019

Guest Editor: Wei Cheng

The Internet of Things (IoT) has attracted the interests of both academia and industry and enables various real-world applications. The acquirement of large amounts of sensing data is a fundamental issue in IoT. An efficient way is obtaining sufficient data by the mobile crowdsensing. It is a promising paradigm which leverages the sensing capacity of portable mobile devices. The crowdsensing platform is the key entity who allocates tasks to participants in a mobile crowdsensing system. The strategy of task allocating is crucial for the crowdsensing platform, since it affects the data requester's confidence, the participant's confidence, and its own benefit. Traditional allocating algorithms regard the privacy preservation, which may lose the confidence of participants. In this paper, we propose a novel three-step algorithm which allocates tasks to participants with privacy consideration. It maximizes the benefit of the crowdsensing platform and meanwhile preserves the privacy of participants. Evaluation results on both benefit and privacy aspects show the effectiveness of our proposed algorithm.

1. Introduction

The IoT is an efficient network that connects various devices on the Internet. It often consists of sensor-equipped devices that can sense, communicate, and react to environmental variations [1]. The developments of IoT in both academia and industry are rapid, because of its promising market value [2, 3]. A large number of IoT applications have been developed and utilized in the society, such as the smart city [4, 5], smart grid [6, 7], and smart traffic [8, 9]. Most of the IoT applications require large amounts of sensing data for monitoring and computing. Therefore, the methods of acquiring sensing data are fundamental in IoT. An efficient way to acquire large amounts of sensing data is using the mobile crowdsensing. It is a promising sensing paradigm which encourages crowds to use mobile devices to collect sensing data. Since small-sized portable mobile devices become extremely prevalent in modern society, the mobile crowdsensing reveals its high performance on the collection of sensing data [10–12]. There is a wide range of IoT applications based on mobile crowdsensing, such as environmental monitoring [13, 14], healthcare [15], and smart cities [16, 17].

A mobile crowdsensing system typically consists of a crowdsensing platform (CSP), a set of data requesters, and a set of participants. Data requesters publish requirements of the sensing data to the CSP. The CSP segments tasks allocate segmented tasks to suitable participants and release the uploaded data from participants to requesters. The goals of the CSP are maintaining sufficient participants and maximizing its own benefit. The strategy of task allocating is crucial for the CSP, since it affects the data requester's confidence, the candidate participant's confidence and its own benefit. Specifically, participants care about their workloads and compensations. They like to collaborate with the CSP that always assigns proper sensing tasks and offers fair compensations. If a CSP always assigns improper tasks to participants, i.e., assigns participants to go far away from their

daily active regions, the CSP will lose many participants. This may decrease the CSP's capability of acquiring sensing data, lose the data requester's confidence, and reduce the CSP's benefit finally. Thus, the CSP should allocate proper tasks to participants for above reasons.

Moreover, the privacy preservation becomes important for the CSP, because the crowds care about the disclosure of their sensitive information in recent years [18–21]. Only the CSP who preserves the privacy can maintain sufficient participants. In this study, we treat the CSP as a trusted entity for participants. However, data requesters are untrusted, and we treat them as potential adversaries. They may be curious about the sensitive information of participants. For example, the requested data always associates with locations in crowdsensing. Adversaries can extract movement patterns of participants from acquired data. The movement patterns are sensitive since adversaries may be able to identify the addresses of participants' homes, schools, or working places [22]. Thus, adversaries usually choose the participants with abnormal profiles as vulnerable users and very likely execute further attacks, so that the privacy preservation is important for the task allocation as well.

Therefore, we investigate the strategy of task allocation with basic considerations and the privacy consideration. Specifically, we first formulate the problem of task allocation. This formulation carefully considers the utility of CSP and the privacy disclosure of participants. A task allocation algorithm with privacy preservation (TAPP) is proposed. It consists of three phases, allocating tasks without privacy preservation, modifying allocations with privacy consideration, and merging the allocations. Furthermore, a series of evaluations show that the proposed algorithm achieves outstanding performance on many aspects.

(1) We first formulate the problem of task allocation with privacy preservation on the CSP's site. We utilize the relative entropy to formulate the privacy disclosure of the participants. The problem formulation is based on a series of assumptions, such as limits of the participant's total time cost and privacy disclosure.

(2) A three-step algorithm, named TAPP, is proposed to allocate proper tasks to participants. The output allocating strategy gain a high benefit for the CSP meanwhile preserves the privacy of participants.

(3) Extensive evaluations are executed based on real-world crowdsensing datasets. The evaluation results show TAPP performs well on maximizing the CSP's benefit and preserving the privacy of participants simultaneously.

The remaining of the paper is organized as follows. The related works are introduced in Section 2. The problem formulation is presented in Section 3. Section 4 discusses the complexity of the formulated problem and introduces the proposed algorithm. Section 5 validates the effectiveness of the algorithm on several aspects. Section 6 concludes the paper.

2. Related Work

A large number of researchers concentrate on the task allocation in mobile crowdsensing. Traditional methods of task allocation lack privacy considerations. They make the allocation strategies according to some basic metrics, such as the quality of sensing data [23–25], the incentive cost [26, 27], the energy consumption [14, 28], and the travel distance [29–31]. The methods which focus on the quality of sensing data are mostly designed for monitoring the environment. They measure the quality of sensing data by a certain metric and attempt to maximize the data quality. The methods focus on the incentive cost allocating tasks on the site of the CSP. Xiong et al. [26] propose an incentive mechanism which minimizes the total budget of the CSP. In their study, the CSP pays according to the participant number. Zhang et al. [27] design a different incentive mechanism which assumes the CSP pays according to tasks. Xiong et al. [14] propose an energy-saving technique, named, piggybacking. It is an optimal collaborative data sensing and uploading scheme which reduces the energy consumption. The travel distance is widely considered in previous studies as well. The methods [29–31] measure the travel distances of participants by numerical values. They contain the same object to minimize the overall travel distance for all sensing tasks.

The crowds care more about the disclosure of their sensitive information in recent years [32]. The privacy preservation in mobile crowdsensing has attracted increasing research interests. Numerous preservation methods are proposed regarding to the spatial privacy, one of the important privacies. Some traditional methods [33–36] based on spatial cloaking are suitable for preserving privacy in mobile crowdsensing. These methods hide the participant spatial information by spatial transformations, generalization, or a set of dummy locations to preserve privacy. Kazemi et al. [37] propose a privacy protection method which directly applies to mobile crowdsensing. This method considers the CSP untrusted and adjusts the spatial information of a participant group. To et al. [38] adopt the differential privacy mechanism and propose a method for spatial crowdsensing task allocation. This method sets a trusted third party entity to aggregate the spatial information of participants. Wang et el. [39] propose a truthful incentive mechanism which preserves the privacy based on differential privacy and auction theory. Duan et al. [40] introduce the reverse auction to task allocation and design allocating algorithm in a novel respective.

A closely related work to ours is presented by Wang et al. [41]. They first preserve the spatial privacy on the participant's site and then allocate the tasks. This adds an extra procedure to the participants. The preservation mechanism is based on the differential privacy. In contrast, our study preserves the privacy on the CSP's site, which does not bother the participants.

Table 1: List of main notations.

Notation	Explanation
$\{L_1, L_2, \ldots, L_N\}$	the subregions
$\{W_1, W_2, \ldots, W_H\}$	the participants
$\{J_1, J_2, \ldots, J_N\}$	the task workloads
$\{t_{i1}, t_{i2}, \ldots, t_{iN}\}$	the allocations of W_i
$P_i = \{p_{i1}, p_{i2}, \ldots, p_{iN}\}$	the actual profile of W_i
$P_i' = \{p_{i1}', p_{i2}', \ldots, p_{iN}'\}$	the observed profile of W_i

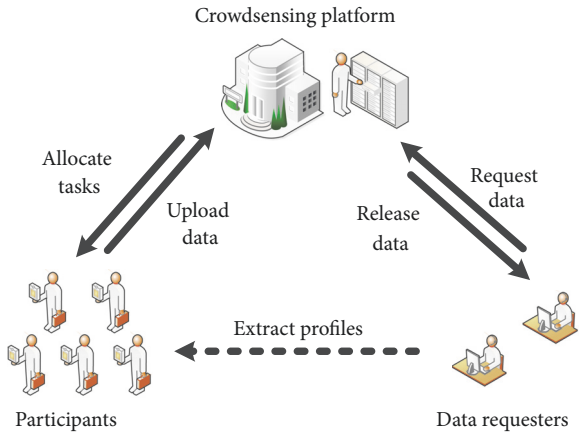

Figure 1: A general system model of crowdsensing.

3. Problem Formulation

This section introduces the general system model, system input, the definition of utility, and the requirements of privacy preservation. The list of main notations is shown in Table 1.

3.1. System Model. A general mobile crowdsensing system consists of three main entities: participants, the CSP, and data requesters, as shown in Figure 1. In this study, we consider the task allocation problem under several specific assumptions of these entities, described as follows.

Participants. The participants receive assigned crowdsensing tasks from the CSP. Each participant actively finishes assigned tasks in time, if the tasks are not overmuch and their privacy is protected. After that, they get a reward from the CSP paid daily or monthly.

Crowdsensing Platform (CSP). The CSP is trusted by participants and knows sensitive information of participants. The CSP receives the crowdsensing requirements from the data requesters and assigns tasks to the suitable participants. The CSP releases the requested data to data requesters and gets rewards from them.

Data Requesters. The data requesters acquire data from the CSP. They may extract the sensitive information of participants from the acquired data, which leads to a privacy leakage.

Note the payment assumption of participants, we consider the scenario that each participant who finishes all assigned tasks is paid daily or monthly [42, 43]. This payment setting is helpful for the quality of acquired sensing data, since it assigns tasks regularly to fixed participants.

3.2. System Input. Assume the crowdsensing system executes the crowdsensing works in a fixed area, which consists of N subregions $\{L_1, L_2, \ldots, L_N\}$. The crowdsensing system has H participants $\{W_1, W_2, \ldots, W_H\}$. Each participant W_i has a personal movement pattern in real-life. We call this pattern participant's *actual profile*, defined as follows:

$$P_i = \{p_{i1}, p_{i2}, \ldots, p_{iN}\}, \quad (1)$$

where $p_{ij} \in [\delta, 1)$, δ is an infinitely small quantity, and $\sum_{j=1}^{N} p_{ij} = 1$. We use δ instead of zero as the lower bound, because this avoids the condition 0/0 happening when we calculate the privacy disclosure. We call L_j is the inactive subregion of W_i if $p_{ij} = \delta$, which means the participant never goes to the subregion L_j. Otherwise, we call L_j is the active subregion of W_i. The CSP acquires the actual profile of each participant, by requiring this information in the register procedure.

When the CSP receives original tasks from the data requesters, the CSP divides the original tasks into unit tasks. Each unit task requires the same time cost and associates with a subregion L_i, $i \in \{1, 2, \ldots, N\}$. Set the time cost of one unit task as one for simplicity. The CSP collects all unit tasks according to associated regions; thus the workload (required time cost) in each subregion is $\{J_1, J_2, \ldots, J_N\}$. $J_i = 1$ means there is only one unit task required in L_i.

The CSP allocates the unit tasks to the participants, denoted as $t_i = \{t_{i1}, t_{i2}, \ldots, t_{iN}\}$. For example, t_{ij} means participant W_i needs to cost t_{ij} time to finish the allocated tasks in region L_j. Each participant has a time threshold τ_i. Assume the participants care about both the workloads and work regions. Therefore, the CSP should avoid following situations to maintain sufficient participants.

(1) Total time cost: if the total time cost $\sum_{j=1}^{N} t_{ij}$ exceeds the time threshold τ_i of the participant, he/she will quit the crowdsensing system. This means the participants are assigned too much works with insufficient rewards.

(2) Work regions: if the participant is assigned to a location where he/she never goes, i.e., $t_{ij} > \delta$, while $p_{ij} \leq \delta$, he/she will quit the crowdsensing system. This means the participants are assigned unsuitable works regarding to their actual profile.

The CSP collects the data generated by participants and sends the data to the data requester for rewards. Then, the data requesters can extract a movement pattern of each

participant from acquired data. We define this pattern as *observed profile* of each participant W_i,

$$P_i' = \left\{p_{i1}', p_{i2}', \ldots, p_{iN}'\right\}, \tag{2}$$

where $p_{ij}' \in [\delta, 1)$, and $\sum_{j=1}^{N} p_{ij}' = 1$. Each p_{ij}' is calculated as follows:

$$p_{ij}' = \frac{t_{ij}}{\sum_{k=1}^{N} t_{ik}}. \tag{3}$$

3.3. Utility. The utility of the CSP is its benefit. Recall the payment assumption that if a participant finishes all assigned tasks, the participant gets payment daily or monthly. If a participant has no assigned task, the participant gets no payment. The CSP gets a reward B when all the requested tasks are finished; meanwhile it pays each recruited participant $Cost$.

The utility of CSP is

$$Utility = B - Cost \left| \left\{ W_i \mid \sum_{j=1}^{N} t_{ij} > \delta \right\} \right|. \tag{4}$$

3.4. Privacy. Since the data requesters are curious and untrusted, they may be the adversaries. They infer the movement patterns of participants, which is the concerned sensitive information, from the observed profiles. The adversaries usually choose the participants with abnormal profiles as vulnerable users and very likely execute further attacks. Thus, we define the difference between an individual and its community as the privacy disclosure in this study.

Assume participant W_i is in community C_i with several other participants. The average profile of the community is

$$\overline{P}_{C_i} = \left\{\overline{p}_{C_i1}, \overline{p}_{C_i2}, \ldots, \overline{p}_{C_iN}\right\}, \tag{5}$$

where

$$\overline{p}_{C_ij} = \sum_{W_m \in C_i} \frac{p_{mj}}{|C_i|}. \tag{6}$$

Then we define the privacy disclosure of $W_i \in C_i$ as the relative entropy between P_i and $\overline{P}_{C_i}$

$$D\left(P_i' \parallel \overline{P}_{C_i}\right) = \sum_{i=1}^{N} p_{ij}' \ln \frac{p_{ij}'}{\overline{p}_{C_ij}}. \tag{7}$$

Furthermore, each participant may set a privacy threshold on this divergence, noted as θ_i. When CSP allocates tasks to participants, the relative entropy for each participant should be bounded by the privacy threshold to guarantee adversaries cannot learn significantly private information from the observed profiles.

3.5. Design Object. Our object is to derive an allocation scheme for the CSP, so that sufficient participants are maintained by allocating suitable tasks and preserving their privacy, meanwhile maximizing the benefit for the CSP. We formalize the problem as follows:

$$\max \quad B - Cost \left| \left\{ W_i \mid \sum_{j=1}^{N} t_{ij} > \delta \right\} \right| \tag{8}$$

$$s.t. \quad D\left(P_i' \parallel \overline{P}_{C_i}\right) \le \theta_i \quad \forall i \in \{1, 2, \ldots, H\} \tag{9}$$

$$\sum_{j=1}^{N} t_{ij} \le \tau_i \quad \forall i \in \{1, 2, \ldots, H\} \tag{10}$$

$$t_{ij} = \delta \quad \forall p_{ij} = \delta \tag{11}$$

$$\sum_{i=1}^{H} t_{ij} \ge J_j \quad \forall j \in \{1, 2, \ldots, N\} \tag{12}$$

4. Task Allocation Algorithm

In this section, we first analyze the complexity of the formulated problem. Then we introduce overview of the proposed algorithm. In the remaining parts, the main phases of the whole algorithm are presented.

4.1. Complexity. The problem of achieving the maximum benefit for the CSP following constraints (9), (10), and (11) is NP-hard.

Consider an arbitrary instance of the minimum set cover problem, consisting of an universal set $U = \{e_1, e_2, \ldots, e_n\}$, series of subsets $\{S_1, S_2, \ldots, S_m\}$. We construct the instance of the maximum benefit problem corresponding to the instance of the minimum set cover problem. We set the privacy threshold $\theta_i = \infty$ and time threshold $\tau_i = \sum_{j=1}^{N} |\{p_{ij} \mid p_{ij} > \delta\}|$ for each W_i, which means there is no privacy constraint and the time threshold equals to the number of subregions where the participant goes. We construct the workload the crowdsensing regions $L = \{L_1, L_2, \ldots, L_n\}$, $J = \{J_1, J_2, \ldots, J_n\}$, $\forall |J_i| = 1$ regarding the universal set U and participants $\{W_1, W_2, \ldots, W_m\}$ corresponding to the subsets. Each W_i corresponding to S_i associates with the actual profile $P_i = \{p_{i1}, p_{i2}, \ldots, p_{in}\}$. Set $p_{ij} = \delta$, $\forall e_j \notin S_i$; otherwise $p_{ij} = 1/|S_i|$, where $|S_i|$ is the element number of S_i. We should set the allocation of each W_i as $\{t_{i1}, t_{i2}, \ldots, t_{in}\}$, $\forall p_{ij} = \delta, t_{ij} = 0$; otherwise $t_{ij} = 1$ for maximizing the benefit, for example, following our construction, given $J = \{1, 1, 1, 1, 1\}$ and a W_i with the actual profile $\{\delta, 1/3, 1/3, 1/3, \delta\}$ and $\tau_i = 3$. Since using less participants increases the benefit of the CSP, the allocation $\{0, 1, 1, 1, 0\}$ is more likely to maximize the benefit than $\{0, 3, 0, 0, 0\}$.

Thus, finding the minimum set cover equals to finding an allocation which covers all the tasks and selects the minimum number of participants, i.e., achieving the maximum benefit. As the reduction shown above, the formulated problem is NP-hard.

Input: participants $\{W_1, W_2, \ldots, W_H\}$, workloads $\{J_1, J_2, \ldots, J_N\}$, each community profile $\{\overline{p}_{C_i1}, \overline{p}_{C_i2}, \ldots, \overline{p}_{C_iN}\}$, each actual profile $\{p_{i1}, p_{i2}, \ldots, p_{iN}\}$, each time treshold τ_i
Output: the allocation $\{t_{i1}, t_{i2}, \ldots, t_{iN}\}$ for each W_i
1: **for** each W_i **do**
2: set each $t_{ij} = 0$, $j \in \{1, 2, \ldots, N\}$, $\Delta\tau_i = 0$;
3: compute $A_i = \{a_{i1}, a_{i2}, \ldots, a_{iN}\}$ by P_i;
4: **for** each J_i **do**
5: **if** $J_i \leq 0$ **then** $can_i = \delta'$;
6: **else** $can_i = \sum_{j=1}^{H} a_{ij} * \tau_j$;
7: **for** each can_i **do**
8: **if** $can_i < J_i$ **then fails**;
9: **while** $\sum_{i=1}^{N} J_i > 0$ **do**
10: update $Can = \{can_1, can_2, \ldots, can_N\}$;
11: $Prio^{task} = \{can_1 - J_1, can_2 - J_2, \ldots, can_N - J_N\}$;
12: $k = \arg\min\ prio_i^{task}$, $prio_i^{task} \in Prio^{task}$;
13: **for** each W_i **do**
14: **if** $a_{ik} == 1$ **then** $\{W_i\} \longrightarrow Prio_k^{part}$;
15: **if** $Prio_k^{part} = \emptyset$ **then fails**;
16: choose $W_l \in Prio_k^{part}$ with $a_{lk} = 1$, minimum $\sum_{j=1}^{N} a_{lj}$, and maximum $\tau_l - \Delta\tau_l$;
17: $t_{lk} = t_{lk} + 1$, $\Delta\tau_l = \Delta\tau_l + 1$, $J_k = J_k - 1$;
18: **if** $\Delta\tau_l == \tau_l$ **then** each $a_{lk} = 0$;
19: **if** $\forall W_i, \Delta\tau_i == \tau_i$ and $\exists J_i > 0$ **then fails**;

Algorithm 1: Task allocation without privacy preservation.

4.2. Overview of the Algorithm. Our task allocation algorithm with privacy preservation, called TAPP, runs on the site of the CSP. The CSP first receives original tasks from the data requesters. It divides the original tasks into unit tasks and merges all unit tasks according to associated subregions. Then the framework analyzes the uploaded information from participants. Then it acquires time thresholds, privacy thresholds, actual profiles, and community profiles. By collecting all the inputs, the framework makes an allocation strategy by following three phases: (i) allocating tasks without privacy preservation, as shown in Algorithm 1; (ii) modifying allocations with the privacy consideration, as shown in Algorithm 2; (iii) reducing allocated participants by merging tasks of two participants, as shown in Algorithm 3.

4.3. Task Allocation Phase. In this part, we introduce the first phase of TAPP algorithm. It iteratively picks a unit task according to the task priority and allocates the unit task to a participant according to the participant priority. The algorithm only considers the time constraints of participants and ignores the privacy constraints. Generally, the algorithm properly allocates all tasks to participants with no consideration of the privacy preservation.

Combined with Algorithm 1, the algorithm first initializes the allocations $\{t_{i1}, t_{i2}, \ldots, t_{iN}\}$, the total workload $\Delta\tau_i$, and the set $A_i = \{a_{i1}, a_{i2}, \ldots, a_{iN}\}$ for each W_i. Each $a_{ij} \in A$ indicates the active or inactive subregion of W_i; i.e., if $p_{ij} > \delta, a_{ij} = 1$; otherwise $a_{ij} = 0$. Then the algorithm computes the set $Can = \{can_1, can_2, \ldots, can_N\}$ in Lines 4~6, where δ' is a very large number. Each can_i denotes the total available time of participants who are active in subregion L_i. It checks the worst case that the tasks are unfinished, even if all participants cost all of their time in one subregion (Lines 7 and 8). After this, it allocates one union task to a participant step by step, until all tasks are allocated.

In each step, the algorithm chooses a subregion and a participant for task allocation. Specifically, it first updates the set *Can* by the method shown in Lines 4~6. Then it computes the priority of tasks associated with different subregions. The priority set is $Prio^{task} = \{prio_1^{task}, prio_2^{task}, \ldots, prio_N^{task}\}$, where $prio_i^{task} = can_i - J_i$. We choose $prio_i^{task}$ indicates the priority, since the tasks in a subregion with minimum excess should first be assigned. For example, assume some $J_i = 10$. The subregion L_i is the active subregion only for participants W_j and W_k, which means only $a_{ji} = 1$ and $a_{ki} = 1$. The time threshold of W_j and W_k are 5, respectively. Thus we can only allocate them spending all their time to finish tasks in L_i for a feasible allocation. So that the algorithm chooses $prio_k^{task}$ associated with L_k, the minimum one in $Prio^{task}$. For each participant who contains active subregion L_k, it chooses the participant W_l that contains minimum number of active regions and the maximum rest time. Then, the algorithm allocates a unit task from J_k to W_l (Line 17). It sets all $a_{ij} \in A_i$ for W_i, when the workload for W_i equals the time threshold. This makes the participant who has no rest time never be chosen anymore.

The allocation iterates until all the tasks are allocated. Otherwise, it fails (Line 19). The allocation loop is the main part of this algorithm, which costs $O(H \cdot \sum_{i=1}^{N} J_i)$ or $O(N \cdot$

Input: subregions $\{L_1, L_2, \ldots, L_N\}$, participants $\{W_1, W_2, \ldots, W_H\}$, each community profile $\{\overline{p}_{C_i1}, \overline{p}_{C_i2}, \ldots, \overline{p}_{C_iN}\}$, each workload $\Delta\tau_i$
Output: the allocation $\{t_{i1}, t_{i2}, \ldots, t_{iN}\}$ for each W_i
update each $A_i = \{a_{i1}, a_{i2}, \ldots, a_{iN}\}$ by P_i;
for each W_i that $\Delta\tau_i > 0$ **do**
  if $D(P_i' \parallel \overline{P}_{C_i}) > \theta_i$ **then** $\{W_i\} \longrightarrow Danger$;
each $\{W_i \mid \Delta\tau_i < \tau_i$ and $W_i \notin Danger\} \longrightarrow Safe$;
sort $W_i \in Danger$ by $\Delta\tau_i$ in descending order;
sort $W_j \in Safe$ by $\tau_j - \Delta\tau_j$ in ascending order;
for each $W_i \in Danger$ **do**
  compute $case_i^- = \{c_{i1}^-, c_{i2}^-, \ldots, c_{iN}^-\}$;
  while $\exists c_{ik}^- < 0$ **do**
    $l = \arg\min c_{ik}^-, S_1 = \emptyset, S_2 = \emptyset$;
    for each $W_j \in Safe$ **do**
      if $a_{jl} > 0$ and $\Delta\tau_j < \tau_j$
        then compute c_{jl}^+;
      else continue;
      if $c_{jl}^+ \le 0$ **then** $\{W_j\} \longrightarrow S_1$;
      else if $D(P_j' \parallel \overline{P}_{C_j}) + c_{jl}^+ \le \theta_i$
        then $\{W_j\} \longrightarrow S_2$;
    if $S_1 \neq \emptyset$ **then**
      $q = \arg\max(\tau_j - \Delta\tau_j), W_j \in S_1$;
    else if $S_2 \neq \emptyset$ **then**
      $q = \arg\max(\tau_j - \Delta\tau_j), W_j \in S_2$;
    else fails;
    $t_{ql} = t_{ql} + 1, \Delta\tau_q = \Delta\tau_q + 1$;
    $t_{il} = t_{il} - 1, \Delta\tau_i = \Delta\tau_i - 1$;
    if $\Delta\tau_q == \tau_q$ **then** $Safe$ deletes W_q;
    if $D(P_i' \parallel \overline{P}_{C_i}) \le \theta_i$ or $\Delta\tau_i = 0$ **then break**;
    compute $case_i^- = \{c_{i1}^-, c_{i2}^-, \ldots, c_{iN}^-\}$;

Algorithm 2: Allocation modification.

Input: subregions $\{L_1, L_2, \ldots, L_N\}$, participants $\{W_1, W_2, \ldots, W_H\}$, workloads $\{J_1, J_2, \ldots, J_N\}$, each community profile $\{\overline{p}_{C_i1}, \overline{p}_{C_i2}, \ldots, \overline{p}_{C_iN}\}$, each actual profile $\{p_{i1}, p_{i2}, \ldots, p_{iN}\}$
Output: the allocation $\{t_{i1}, t_{i2}, \ldots, t_{iN}\}$ for each W_i
update each $A_i = \{a_{i1}, a_{i2}, \ldots, a_{iN}\}$ by P_i;
for each W_i **do**
  for each W_j that $j \neq i$ **do**
    if $\tau_j - \Delta\tau_j > \Delta\tau_i$ and A_j covers A_i **then**
      $tmp = \{t_{i1} + t_{j1}, t_{i2} + t_{j2}, \ldots, t_{iN} + t_{jN}\}$;
      compute P_{tmp}' by tmp;
      if $D(P_{tmp}' \parallel \overline{P}_{C_j}) \le \theta_j$ **then**
        $t_j = tmp, \Delta\tau_j = \Delta\tau_j + \Delta\tau_i$;
        each $t_{ik} = 0,\ k \in \{1, 2, \ldots, N\}, \Delta\tau_j = 0$;
for each W_i that $\Delta\tau_i > 0$ **do**
  if $D(P_i' \parallel \overline{P}_{C_i}) > \theta_i$ **then fails**;

Algorithm 3: Allocation mergence.

$\sum_{i=1}^{N} J_i$) time. It finally costs $O(H \cdot \sum_{i=1}^{N} J_i)$ time in total, since H is always much bigger than N in practice.

4.4. Allocation Modification Phase. We introduce the second phase of TAPP in this part. We call the participant whose privacy leakage is bigger than the privacy threshold as a dangerous participant. In this phase, the algorithm modifies the allocations among participants in order to reduce dangerous participants. Specifically, the algorithm transfers some workloads from dangerous participants to safe participants, in order to make all participants safe.

Combined with Algorithm 2, the algorithm first updates set A by the actual profiles. Then it checks all allocated participants and adds dangerous participants in set *Danger* (Line 2~3). The set *Safe* contains two kinds of participants: (i) the safe participants have some allocated tasks but their workloads are less than the time threshold; (ii) the participants have no allocated task. The algorithm sorts the participants in *Danger* regarding to their workloads in descending order. It sorts each participant W_j in *Safe* regarding $\tau_j - \Delta\tau_j$ in ascending order. This means the dangerous participants first choose safe participants with allocated tasks, when they search for safe participants to transfer their workloads. After these, the algorithm transfers some allocations from dangerous participants to safe participants.

For each dangerous participant, the algorithm first computes the set $case_i^- = \{c_{i1}^-, c_{i2}^-, \ldots, c_{iN}^-\}$. Each c_{ij}^- denotes the variation of the privacy leakage if W_i reduces a unit task associated with L_j. Moreover, we set $c_{ij}^- = \delta'$ if there is no candidate safe participants to allocate an unit task in L_j, where δ' is a very large number. Thus, $c_{ij}^- < 0$ means the privacy leakage will reduce if we reduce an unit task in L_j for W_i. The algorithm iteratively transfers a unit task associated with minimum c_{ij}^- to the candidate safe participant, until the privacy leakage of the dangerous participant is less than the threshold or its workload is zero (Lines 9~27). Specifically, a candidate safe participant W_j should satisfy $a_{jl} > 0$ and $\Delta\tau_j < \tau_j$, given the transferred task associated with L_l. Then it computes each c_{jl}^+ (Lines 12~14), which is similar to c_{jl}^-. The difference is that c_{jl}^+ is the variation of the privacy leakage if W_j adds an unit task associated with L_l. The algorithm chooses the candidate W_q according to the rest workload (Lines 18~22). Then the algorithm transfers an unit task in L_l from dangerous W_i to safe W_q.

The algorithm attempts to modify the allocations of all dangerous participants by the above transferring method. The best case is that there is no dangerous participant after these modifications. The time complexity of this phase is $O(H \cdot N \cdot \sum_{i=1}^{N} J_i)$ in total.

4.5. Allocation Mergence Phase. After the modification phase, we introduce the allocation mergence phase of TAPP in this part. The basic idea of this phase is that we can transfer all the allocations of a W_i to a W_j, if the time and privacy constraints of W_j are still satisfied. This procedure reduces the number of allocated participants; thus it increases the utility of the CSP.

Table 2: Statistics for each city.

City	# Active Users
Cleveland, OH, USA	332
Tempe, AZ, USA	342
Calgary, AB, Canada	428

Combined with Algorithm 3, the algorithm first updates each set A_i by actual profiles. Then it iteratively checks each participant pair (W_i, W_j), $i \neq j$ whether all their allocations can be merged and allocated to W_j (Lines 2~9). For each participant pair (W_i, W_j), the algorithm first check the time constraint of W_j, where A_j covers A_i means $\nexists a_{jk} = 0, a_{ik} = 1$ (Line 4). The set tmp is the mergence of all allocations (Line 5). Then the algorithm checks the privacy constraint of W_j, if we allocate the tmp to W_j. The algorithm allocates the mergence to W_j if the time and privacy constraints are satisfied and allocates no task to W_i (Line 7~9).

The algorithm checks all allocated participants at the end. If there still are some dangerous participants, the algorithm fails. The time complexity of this phase is $O(N \cdot H^2)$ in total.

5. Evaluation

We evaluate the performance of TAPP towards a real-world dataset from Yelp (https://www.yelp.com/dataset/challenge). Yelp is a location-based service system where reviewers publish reviews and comments for nearby businesses. In our evaluation, we consider reviewers as participants, and reviews as tasks. A review associates with a business, and the business associates with a location.

Three cities are considered in our evaluation: Cleveland, OH, USA; Tempe, AZ, USA; Calgary, AB, Canada. The user activities in each city reflect different real-world situations. Thus, these three cities are representative for evaluations. Specifically, we focus on the active participants with more than 30 reviews. The numbers of active participants in each city are shown in Table 2. The area of each city is divided into 3 by 3 grids. Since each review has a corresponding grid, the participant's actual profile is the ratio of reviews located in each grid. We consider the participants of a city belong to a community. The community profile for each city is the average value evaluated from all the participants' actual profiles. The cost for finishing a unit task is set to 1.

This evaluation focuses on two metrics: the number of dangerous participants, and the utility of the CSP. A dangerous participant is the one who has a privacy leakage more than its privacy threshold at the end of allocation. The number of dangerous participant is bigger than zero means the allocation is unfeasible. However, this metric helps us analyze the effectiveness of the allocation algorithm. Thus,

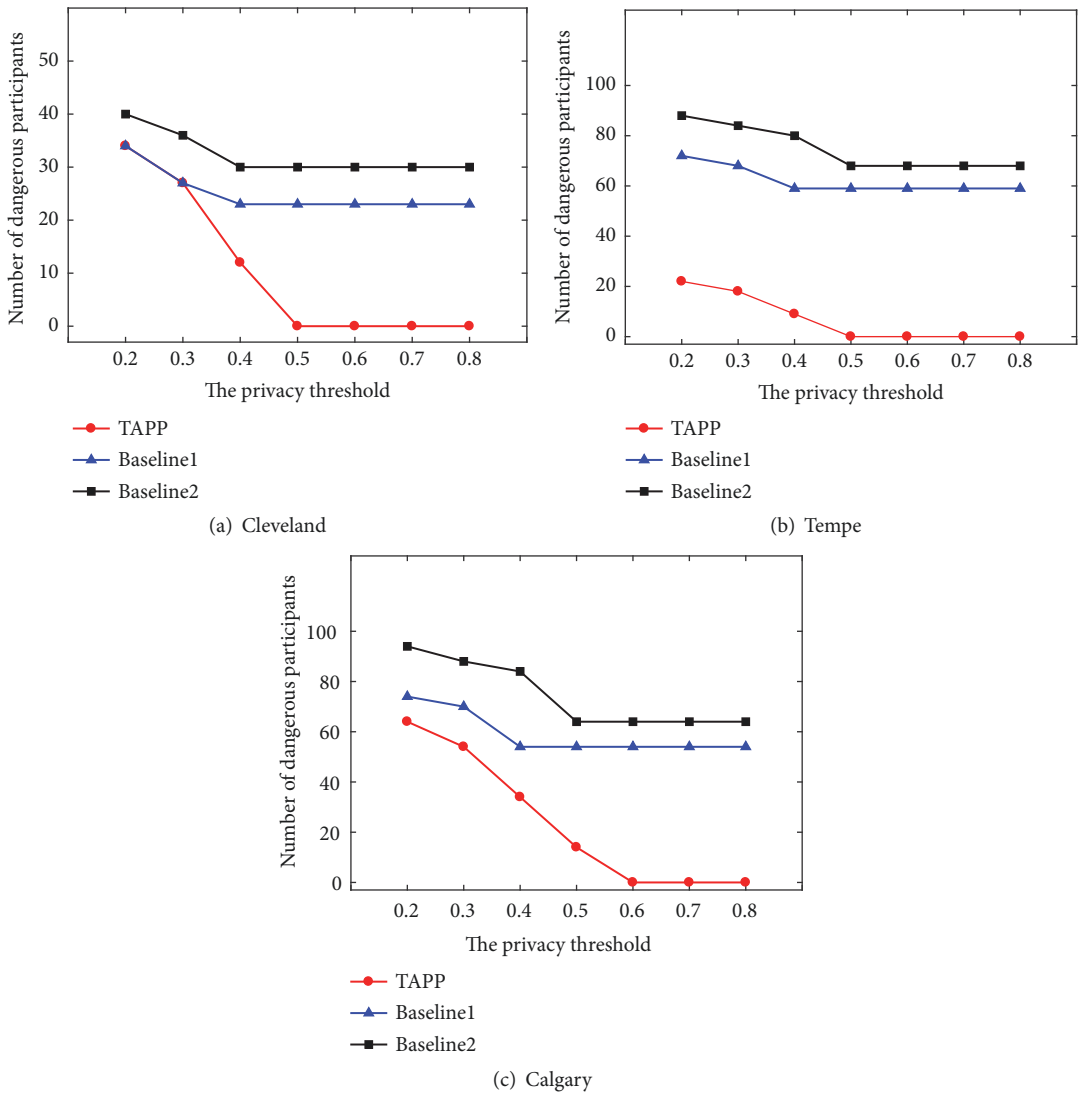

FIGURE 2: The number of dangerous participants among different allocating methods.

we change Line 22 to "else break" in Algorithm 2 for the evaluation.

The TAPP is compared with two baseline allocating methods. Baseline1 is the first phase of TAPP. Baseline2 is a greedy allocating method. It prefers to allocate the maximum J_i to participants who have the active subregion L_i and the maximum time threshold.

5.1. General Performance. We validate the effectiveness of TAPP in this part. The time threshold and the privacy threshold are set as the same value for each participant, respectively. Moreover, the privacy threshold ranges from 0.2 to 0.8.

Figure 2 shows the numbers of dangerous participants in each city. As we can see, when the privacy threshold is small, three algorithms suffer large numbers of dangerous participants. However, the number of dangerous participants in TAPP is averagely 76.23% less than Baseline1 and Baseline2, when the privacy threshold ranges from 0.2 to 0.4, because the second and third phases of TAPP help to reduce the dangerous participants. The TAPP acquires feasible allocating solutions when the privacy threshold grows. Specifically, TAPP gets feasible solutions after the privacy threshold achieves 0.5, 0.5, and 0.6, respectively. Note that TAPP first acquires a feasible solution in Calgary with bigger privacy threshold than the other cities. It is because that the city with more population, the profiles are more heterogeneous. Then the algorithm performs relatively worse.

Figure 3 shows the utility of the CSP in each city. We treat an allocating solution as a feasible solution for comparison, even if it is unfeasible. Specifically, given H participants in total and H' participants who have allocated tasks, the utility of the CSP is $H - H'$ in this evaluation section. Because Baseline2 is based on the greedy strategy, the results of Baseline2 are close to optimal if there is no privacy constraint. The TAPP gets close to the results of Baseline2, when the privacy threshold grows bigger. By further analysis between

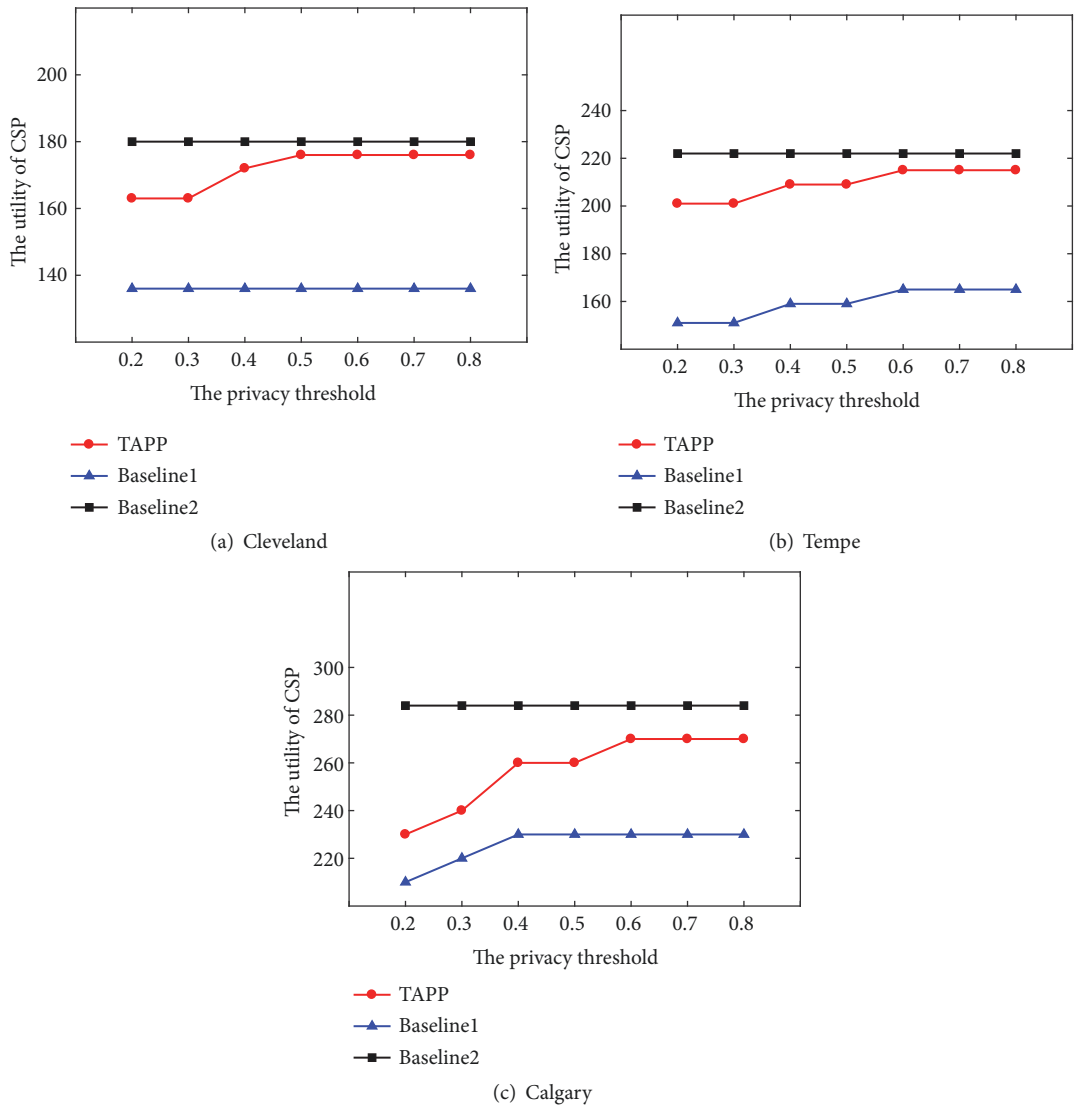

Figure 3: The utility of the CSP among different allocating methods.

TAPP and Baseline1, the second and third phases of TAPP averagely increase the utility 24.76% in three cities. Focusing on the results in Calgary, the heterogeneity of profiles affects the utility as well, since it affects the allocating solution.

5.2. Performance for Different Cases. In this part, we investigate the performance of TAPP for different types of task distributions. The results indicate the effectiveness of our algorithm under different task workloads. Specifically, we set the requested tasks $J^c = \{J_1^c, J_2^c, \ldots, J_N^c\}$ following the distribution of the community profile, and the requested tasks $J^u = \{J_1^u, J_2^u, \ldots, J_N^u\}$ following the uniform distribution. J^c and J^u satisfy $\sum_{i=1}^{N} J_i^c = \sum_{i=1}^{N} J_i^u$. The results under J^c and J^u are denoted as com-distribution and uni-distribution, respectively. The rest settings are as the same as in Section 5.1.

Figure 4 shows the numbers of dangerous participants under these two distributions. The privacy threshold under uni-distribution is bigger than com-distribution, when TAPP first acquires a feasible solution. This is caused by the privacy constraint. The privacy constraint is based on the relative entropy between the observed profile and the community profile. Since J^c follows the distribution of the community profile, the algorithm is easier to acquire a solution which satisfies the privacy constraint. Meanwhile, TAPP acquires the first feasible solution under uni-distribution, when the privacy threshold is a little bigger than it under com-distribution. Comparing the results among three cities, the heterogeneity of profiles affects the dangerous numbers under different distributions as well.

Figure 5 shows the utilities of the CSP under these two distributions. The utilities under uni-distribution are less than com-distribution in all cases. This is because the task allocation is based on the privacy constraint, which is strongly related to the community profile. Since the com-distribution follows the same distribution with the community profile, the task allocation regarding the privacy constraint under

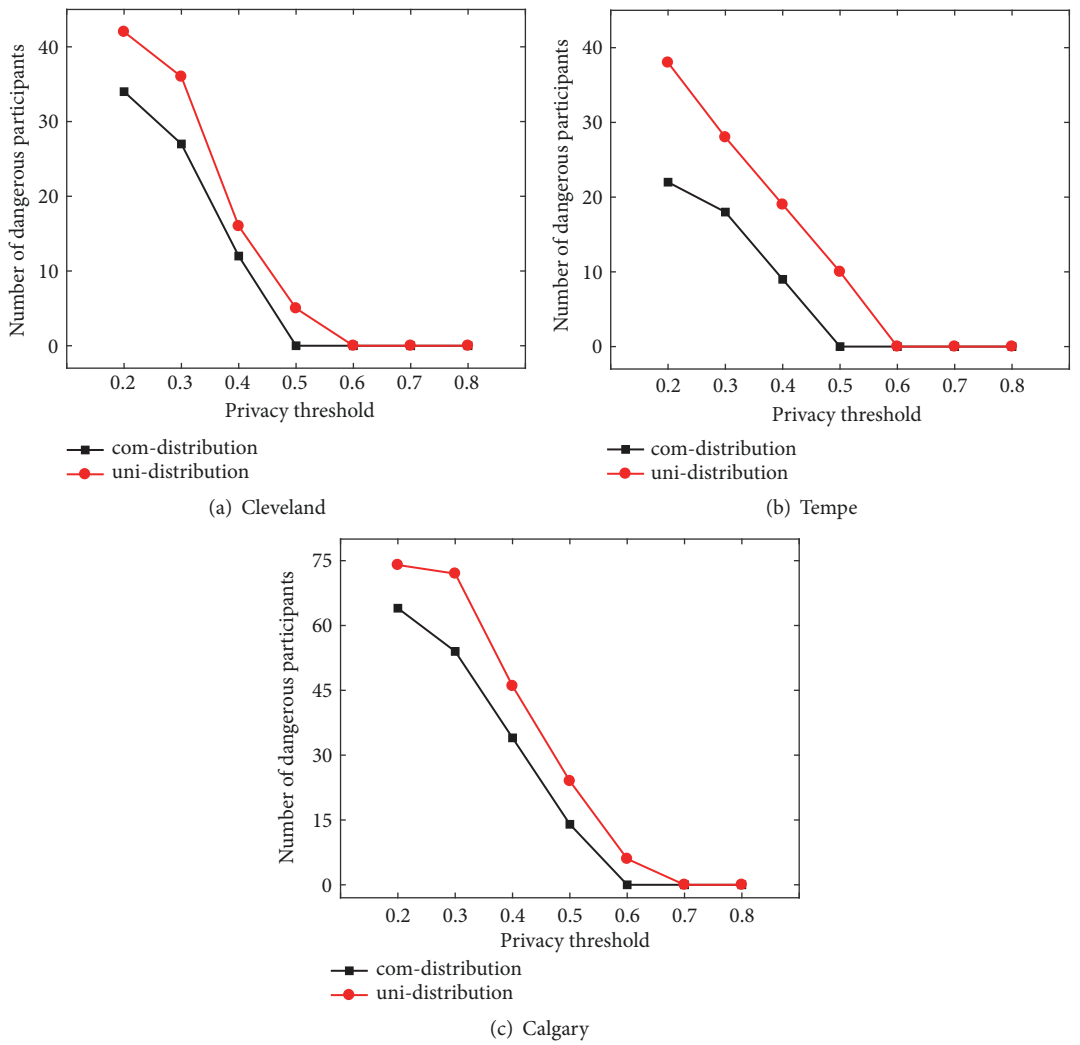

Figure 4: The number of dangerous participants under different task distributions.

the com-distribution is easier than the uni-distribution. The TAPP tries to make the solution satisfying the privacy constraint by allocating more participants, when given a more strict task distribution. The utility increases 6.53% and 6.15% under different distributions in Tempe and Calgary, respectively. Thus, the heterogeneity of profiles may not affect the utilities under different distributions.

6. Conclusion

Since the crowds care more about their privacy disclosure in recent years, the design of a task allocation algorithm should consider the privacy preservation. In this study, we investigate the algorithm of task allocation with basic considerations and the spatial privacy consideration. The problem formulation of task allocation is first presented. After that, we propose a task allocation algorithm on CSP's site with privacy preservation based on the formulation. It consists of three phases, allocating tasks without privacy preservation, modifying allocations with privacy consideration and merging the allocations. The algorithm maximizes the benefit of the CSP, but meanwhile preserves the special privacy of participants. Evaluation results on utility and privacy aspects show the effectiveness of our proposed algorithm.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research work is supported by the key research plan for State Commission of Science Technology of China (2018YFC0807501, 2018YFC0807503), by the Foundation of Science & Technology Department of Sichuan province

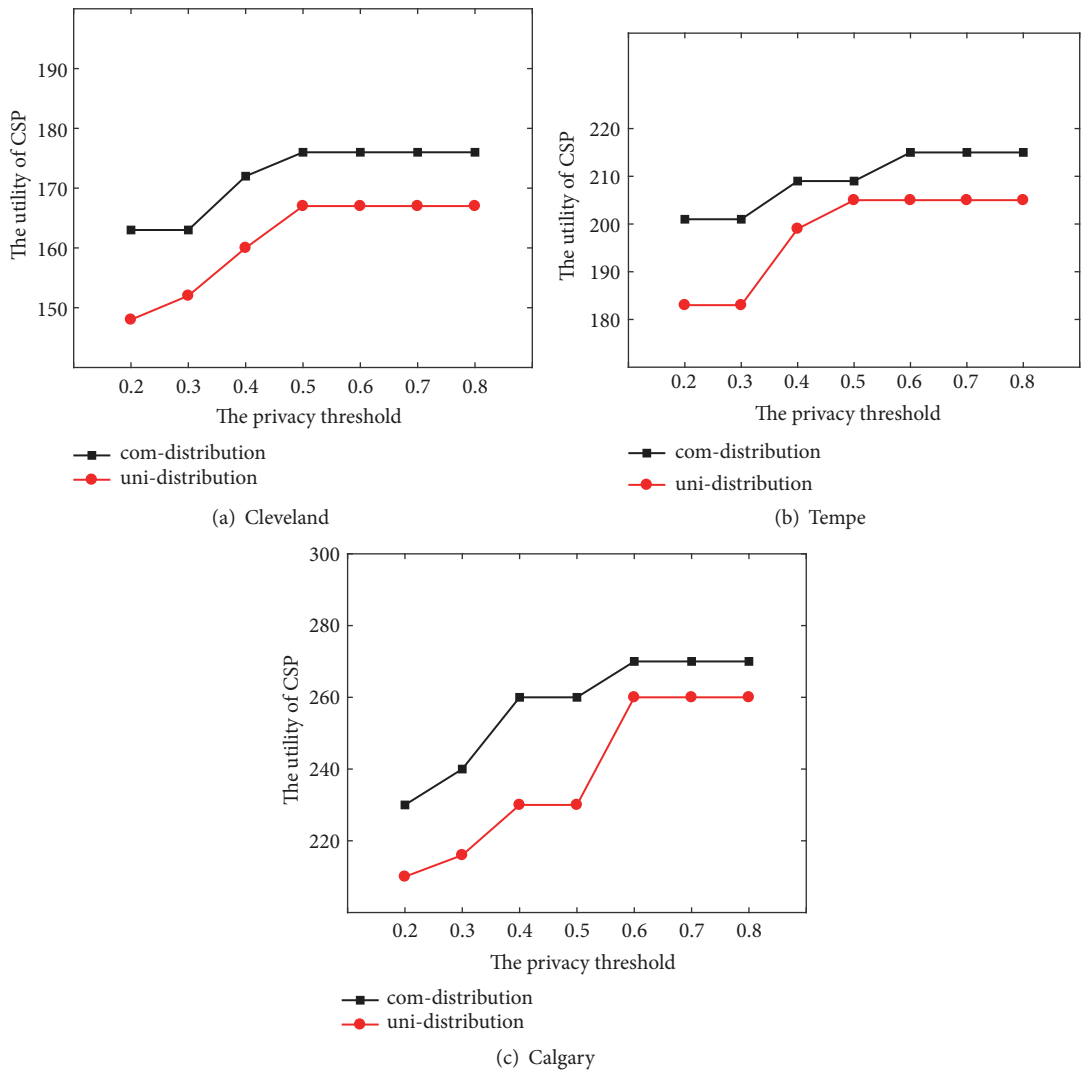

Figure 5: The utility of the CSP under different task distributions.

under Grants nos. 2017JY0027, 2017JY0007, 2018JY0067, 2017GFW0128, and 2016FZ0108, and by the Sichuan Provincial Economic and Information Commission (no. 2018DS010).

References

[1] M. Seliem, K. Elgazzar, and K. Khalil, "Towards privacy preserving iot environments: a survey," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1032761, 15 pages, 2018.

[2] *Worldwide Internet of Things Forecast*, 2017, https://www.idc.com/getdoc.jsp?containerId=IDC.

[3] R. Gartner, "Forecast: the internet of things, worldwide," in *The Internet of Things, Forecast*, 2017.

[4] D. C. Bogatinoska, R. Malekian, J. Trengoska, and W. A. Nyako, "Advanced sensing and internet of things in smart cities," in *Proceedings of the 39th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2016*, pp. 632–637, Croatia, June 2016.

[5] J. Jin, J. Gubbi, S. Marusic, and M. Palaniswami, "An information framework for creating a smart city through internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 2, pp. 112–121, 2014.

[6] M. Yun and B. Yuxin, "Research on the architecture and key technology of Internet of Things (IoT) applied on smart grid," in *Proceedings of the International Conference on Advances in Energy Engineering (ICAEE '10)*, pp. 69–72, June 2010.

[7] N. Bui, A. P. Castellani, P. Casari, and M. Zorzi, "The internet of energy: a web-enabled smart grid system," *IEEE Network*, vol. 26, no. 4, pp. 39–45, 2012.

[8] S. Sahabiswas, S. Saha, P. Mitra et al., "Drunken driving detection and prevention models using Internet of Things," in *Proceedings of the 7th IEEE Annual Information Technology, Electronics and Mobile Communication Conference, IEEE IEMCON 2016*, Canada, October 2016.

[9] T. T. Thakur, A. Naik, S. Vatari, and M. Gogate, "Real time traffic management using Internet of Things," in *Proceedings of the 2016 International Conference on Communication and Signal Processing, ICCSP 2016*, pp. 1950–1953, India, April 2016.

[10] J. L. Cai, M. Yan, and Y. Li, "Using crowdsourced data in location-based social networks to explore influence maximization," in *Proceedings of the IEEE INFOCOM 2016 - IEEE Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

[11] J. Li, Z. Cai, J. Wang, M. Han, and Y. Li, "Truthful incentive mechanisms for geographical position conflicting mobile crowdsensing systems," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 324–334, 2018.

[12] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science & Engineering*, no. 99, pp. 1-1, 2018.

[13] H. To, L. Fan, L. Tran, and C. Shahabi, "Real-time task assignment in hyperlocal spatial crowdsourcing under budget constraints," in *Proceedings of the 14th IEEE International Conference on Pervasive Computing and Communications, PerCom 2016*, Australia, March 2016.

[14] H. Xiong, D. Zhang, L. Wang, and H. Chaouchi, "EMC3: energy-efficient data transfer in mobile crowdsensing under full coverage constraint," *IEEE Transactions on Mobile Computing*, vol. 14, no. 7, pp. 1355–1368, 2015.

[15] H. Zhang, J. Liu, and N. Kato, "Threshold tuning-based wearable sensor fault detection for reliable medical monitoring using bayesian network model," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1886–1896, 2018.

[16] M. Zhang, P. Yang, C. Tian et al., "Quality-aware sensing coverage in budget-constrained mobile crowdsensing networks," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 9, pp. 7698–7707, 2016.

[17] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 66, no. 5, pp. 810–819, 2017.

[18] X. Zheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Follow but no track: privacy preserved profile publishing in cyber-physical social systems," *IEEE Internet of Things Journal*, 2017.

[19] Y. Huo, C. Yong, and Y. Lu, "Re-adp: Real-time data aggregation with adaptive ω-event differential privacy for fog computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 6285719, 13 pages, 2018.

[20] Y. Huo, Y. Tian, L. Ma, X. Cheng, and T. Jing, "Jamming strategies for physical layer security," *IEEE Wireless Communications Magazine*, vol. 25, no. 1, pp. 148–153, 2018.

[21] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart internet of things systems: a consideration from a privacy perspective," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 55–61, 2018.

[22] S. Hayashida, D. Amagata, T. Hara, and X. Xie, "Dummy generation based on user-movement estimation for location privacy protection," *IEEE Access*, vol. 6, pp. 22958–22969, 2018.

[23] L. Wang, D. Zhang, A. Pathak et al., "CCS-TA: Quality-guaranteed online task allocation in compressive crowdsensing," in *Proceedings of the 3rd ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2015*, pp. 683–694, Japan, September 2015.

[24] L. Wang, D. Zhang, Y. Wang, C. Chen, X. Han, and A. M'hamed, "Sparse mobile crowdsensing: challenges and opportunities," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 161–167, 2016.

[25] Y. Zhu, Z. Li, H. Zhu, M. Li, and Q. Zhang, "A compressive sensing approach to urban traffic estimation with probe vehicles," *IEEE Transactions on Mobile Computing*, vol. 12, no. 11, pp. 2289–2302, 2013.

[26] H. Xiong, D. Zhang, G. Chen, L. Wang, and V. Gauthier, "CrowdTasker: maximizing coverage quality in piggyback crowdsensing under budget constraint," in *Proceedings of the 13th IEEE International Conference on Pervasive Computing and Communications, PerCom 2015*, pp. 55–62, USA, March 2015.

[27] D. Zhang, H. Xiong, L. Wang, and G. Chen, "CrowdRecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint," in *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 703–714, ACM, September 2014.

[28] X. Sheng, J. Tang, and W. Zhang, "Energy-efficient collaborative sensing with mobile phones," in *Proceedings of the IEEE Conference on Computer Communications, INFOCOM 2012*, pp. 1916–1924, USA, March 2012.

[29] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, "ActiveCrowd: a framework for optimized multitask allocation in mobile crowdsensing systems," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 392–403, 2017.

[30] S. He, D.-H. Shin, J. Zhang, and J. Chen, "Toward optimal allocation of location dependent tasks in crowdsensing," in *Proceedings of the 33rd IEEE Conference on Computer Communications (INFOCOM '14)*, pp. 745–753, IEEE, Toronto, Canada, May 2014.

[31] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, "TaskMe: multi-task allocation in mobile crowd sensing," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp 2016*, pp. 403–414, Germany, September 2016.

[32] G. Luo, K. Yan, X. Zheng, L. Tian, and Z. Cai, "Preserving adjustable path privacy for task acquisition in mobile crowdsensing systems," *Information Sciences*, 2018.

[33] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Proceedings of the International Symposium on Spatial and Temporal Databases*, pp. 239–257, Springer, 2007.

[34] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, "Preventing location-based identity inference in anonymous spatial queries," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 12, pp. 1719–1733, 2007.

[35] S. Wang and X. Sean Wang, "In-device spatial cloaking for mobile user privacy assisted by the cloud," in *Proceedings of the 11th IEEE International Conference on Mobile Data Management, MDM 2010*, pp. 381–386, USA, May 2010.

[36] H. Kido, Y. Yanagisawa, and T. Satoh, "An anonymous communication technique using dummies for location-based services," in *Proceedings of the 2nd International Conference on Pervasive Services (ICPS '05)*, pp. 88–97, IEEE Press, July 2005.

[37] L. Kazemi and C. Shahabi, "A privacy-aware framework for participatory sensing," *ACM SIGKDD Explorations Newsletter*, vol. 13, no. 1, p. 43, 2011.

[38] H. To, G. Ghinita, and C. Shahabi, "Framework for protecting worker location privacy in spatial crowdsourcing," *Proceedings of the VLDB Endowment*, vol. 7, no. 10, pp. 919–930, 2014.

[39] Y. Wang, Z. Cai, X. Tong, Y. Gao, and G. Yin, "Truthful incentive mechanism with location privacy-preserving for mobile crowdsourcing systems," *Computer Networks*, vol. 135, pp. 32–43, 2018.

[40] Z. Duan, W. Li, and Z. Cai, "Distributed auctions for task assignment and scheduling in mobile crowdsensing systems," in *Proceedings of the 37th IEEE International Conference on Distributed Computing Systems, ICDCS 2017*, pp. 635–644, USA, June 2017.

[41] L. Wang, T. Wang, D. Yang, D. Zhang, X. Han, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proceedings of the 26th International World Wide Web Conference, WWW 2017*, pp. 627–636, Australia, April 2017.

[42] E. Wang, Y. Yang, J. Wu, W. Liu, and X. Wang, "An efficient prediction-based user recruitment for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 16–28, 2018.

[43] Y. Yang, W. Liu, E. Wang, and H. Wang, "Beaconing control strategy based on game theory in mobile crowdsensing," *Future Generation Computer Systems*, vol. 86, pp. 222–233, 2018.