Privacy-Preserving Online Task Assignment in Spatial Crowdsourcing: A Graph-based Approach

Hengzhi Wang[1,2], En Wang*[,1,2], Yongjian Yang[1,2], Jie Wu[3], and Falko Dressler[4]

[1]*College of Computer Science and Technology, Jilin University, China*
[2]*Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, China*
[3]*Department of Computer and Information Sciences, Temple University, USA*
[4]*School of Electrical Engineering and Computer Science, TU Berlin, German*
Emails: wanghz17@mails.jlu.edu.cn; {wangen, yyj}@jlu.edu.cn; jiewu@temple.edu; dressler@ccs-labs.org

***Abstract*—Recently, the growing popularity of Spatial Crowdsourcing (SC), allowing untrusted platforms to obtain a great quantity of information about workers and tasks' locations, has raised numerous privacy concerns. In this paper, we investigate the privacy-preserving task assignment in the online scenario, where workers and tasks arrive at the platform in real time and tasks should be assigned to workers immediately. Traditional online task assignments usually make a benchmark to decide the following task assignment. However, when location privacy is considered, the benchmark does not work anymore. Hence, how to assign tasks in real time based on workers and tasks' obfuscated locations is a challenging problem. Especially when many tasks could be assigned to one worker, path planning should be considered, making the assignment more challenging. To this end, we propose a Planar Laplace distribution based Privacy mechanism (PLP) to obfuscate real locations of workers and tasks, where the obfuscation does not change the ranking of these locations' relative distances. Furthermore, we design a Threshold-based Online task Assignment mechanism (TOA), which could deal with the one-worker-many-tasks assignment and achieve a satisfactory competitive ratio. Simulations based on two real-world datasets show that the proposed algorithm consistently outperforms the state-of-the-art approach.**

***Index Terms*—Spatial crowdsourcing, privacy protection, online task assignment, one-worker-many-tasks assignment.**

I. INTRODUCTION

Recently, the increasing availability of mobile Internet and portable devices has led to a promising paradigm, generally referred to as Spatial Crowdsourcing (SC) [1]–[3]. Many SC applications, such as the taxi-calling application Uber and the product placing application Gigwalk, are emerging in our daily life. As one of the most foundational issues in SC, task assignment has attracted much attention in the last decade [4]–[8], which mainly includes three parts: the platform, workers, and tasks. Specifically, the platform collects the locations of workers and tasks, and then assigns tasks to nearby workers for minimizing the cost (i.e., the distance) of performing tasks.

Compared with the offline task assignment [9]–[11] where full information of workers and tasks is known in advance, a more realistic scenario should be the online task assignment [12]–[15] where tasks arrive at the platform in real time and need to be assigned immediately. For example, taxi-calling

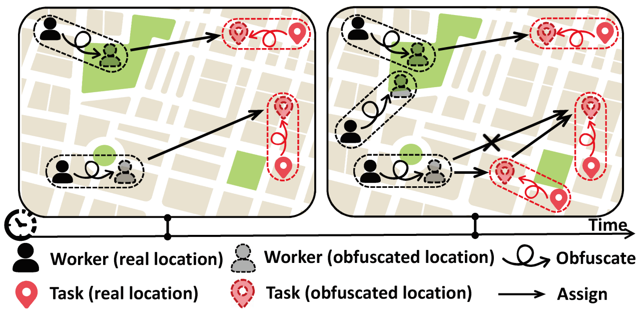


Fig. 1: Privacy-preserving online task assignment: one-worker-many-tasks assignment.

tasks of passengers arrive at the Uber application in real time, and should be immediately assigned to the nearby drivers. However, during task assignment, some sensitive information (e.g., locations) of workers and tasks has to be exposed to the platform, which raises serious privacy issues since the platform may be untrusted [16]. Motivated by this, we focus on the privacy-preserving online task assignment in this paper. Fig. 1 (left) shows such an online scenario, where both workers and tasks arrive one by one in real time and report their obfuscated locations to the platform. Given a set of workers and tasks, we prefer to assign tasks to nearby workers to minimize the total distance of workers to perform the task. However, in the online scenario, workers and tasks' locations are invisible to the platform until they arrive. Hence, when an online task arrives, it is difficult to assign the task to a suitable worker.

To this end, some existing works utilize the optimal stopping problem [13] or the Hierarchically well-Separated Tree (HST) [12] to solve the online task assignment. However, they ignore the privacy issues, that is, when the workers and tasks report the obfuscated locations to the platform, the assignment benchmark [13] or the measurement tree [12] is no longer accurate based on the obfuscated locations. Hence, their approaches can not work anymore. Actually, there is a tradeoff between privacy protection and online task assignment, i.e., high-level privacy protection results in inaccurate task assignment. Hence, how to *protect location privacy as well as ensure the availability of obfuscated locations* is the first challenge.

To this end, some works [5], [17] investigate the privacy-

*The corresponding author is En Wang.

preserving online task assignment. However, they only focus on the one-worker-one-task assignment, where a worker can only perform a task. But in most realistic scenarios, each worker can perform many tasks, such as the ride-sharing service [18]. Inspired by this, we consider a more general assignment scenario, i.e., the one-worker-many-tasks assignment as depicted in Fig. 1 (right), illustrating that when a new task arrives and is assigned to a worker who has already been assigned with other tasks, the worker dynamically adjusts its path plan and performs these two tasks in sequence. Considering both the task assignment and plan planning makes the one-worker-many-tasks assignment more challenging. Hence, the second challenge is how to *propose an efficient approach to conduct the one-worker-many-tasks online assignment*.

More importantly, when considering both privacy protection and one-worker-many-tasks assignment in the online scenario, it is difficult to provide the assignment performance bound to the offline solution. Hence, how to *prove the effectiveness of the proposed online task assignment mechanism, i.e., the competitive ratio* is the third challenge.

To overcome the above challenges, we propose a privacy-preserving online task assignment mechanism. Specifically, for *the first challenge*, we design a Planar Laplace distribution based Privacy mechanism (PLP). Based on PLP, workers and tasks obfuscate their real locations to other locations around them by adding the noise. PLP is proven to achieve ϵ-Geo-Indistinguishability and ensure the availability of obfuscated locations, i.e., the obfuscation does not change the ranking of these locations' relative distances. For *the second challenge*, we design a Threshold-based Online task Assignment algorithm (PLP-TOA) by leveraging the obfuscated locations to solve the one-worker-many-tasks assignment in two steps: (1) the platform estimates a threshold using a graph-based approach according to the historical data. Specifically, we construct an extended network graph and formulate the one-worker-many-tasks assignment as the Extended Minimum-Cost Flow (EMCF) problem, and then estimate the threshold; (2) when tasks arrive, the platform filters inappropriate tasks based on the estimated threshold and then assigns tasks efficiently based on EMCF in the online scenario. Regarding *the third challenge*, we first offer the competitive ratio of the no-privacy version algorithm of PLP-TOA. Then, by combining properties of the privacy mechanism PLP, we further prove that PLP-TOA achieves a competitive ratio related to the privacy budget and historical data. Finally, extensive simulations based on two real-world datasets demonstrate that PLP-TOA consistently outperforms the state-of-the-art approach.

- We design a privacy mechanism PLP based on the planar Laplace distribution to balance the tradeoff between privacy protection and task assignment in the online SC scenario. PLP is proven to achieve ϵ-Geo-Indistinguishability and ensure the availability of the obfuscated locations.
- To the best of our knowledge, we are the first to solve the one-worker-many-tasks assignment in the online scenario. Specifically, based on the historical data, we formulate the online assignment problem as an extended minimum-cost flow (EMCF) problem and leverage the graph-based approach to estimate a threshold. Based on the threshold and EMCF, the platform assigns tasks to suitable workers in the online scenario.
- We offer the competitive ratio of the no-privacy version algorithm of PLP-TOA. Then, by combining the privacy mechanism PLP, we prove that PLP-TOA achieves the competitive ratio $O(1/(\bar{d} \cdot \epsilon^2))$, where $\bar{d}$ is the average distance of performing a task in the offline solution using the historical data, and ϵ is the privacy budget.
- We conduct extensive simulations based on two real-world datasets to evaluate the performance of the proposed algorithm, and the results illustrate that our algorithm outperforms the state-of-the-art approach.

II. Related Work

Regarding the privacy-preserving task assignment, Wang *et al.* [4] propose a geo-obfuscated task allocation framework to protect workers' location privacy and formulate the task assignment as a mixed-integer non-linear programming problem. Then, they solve the problem using the genetic algorithm. Different from the location privacy, Ren *et al.* [16] focus on the cost privacy of workers, and formulate the task assignment with the obfuscated costs as an integer linear programming problem. Then, they propose a Hungarian method and a greedy method in two different scenarios, respectively. In addition, Xiao *et al.* [19] focus on protecting the privacy of both workers' quality and cost when assigning tasks to workers for minimizing the total cost. Based on the secret sharing scheme, they propose two secure task assignment protocols, which are based on the greedy method and proven to achieve an approximation ratio. However, all of [4], [16], [19] focus on the offline scenario, i.e., the information of workers and tasks are known in advance to the platform. To this end, Shahabi *et al.* [5] investigate the privacy-preserving task assignment in the online scenario and propose a heuristic approach, which estimates the reachability probability between a worker and a task, and assigns tasks greedily based on the probability. Unfortunately, the performance of the proposed heuristic approach cannot be guaranteed theoretically.

Furthermore, Tao *et al.* [17] propose a privacy-preserving online task assignment framework called TBF through constructing a hierarchically well-separated tree, which achieves a performance guarantee, i.e., the competitive ratio. Based on the tree, they confuse the locations of workers and tasks and assign the online tasks greedily. It is worth noting that all the works mentioned above only focus on the one-worker-one-task assignment. However, when a worker is able to perform multiple tasks, these approaches cannot work. To this end, given a worker, Demiryurek *et al.* [20] study the one-worker-many-tasks assignment, and propose a series of algorithms based on dynamic programming and branch-and-bound strategies. However, when there are multiple workers, the algorithms cannot work. Inspired by this, Deng [21] further investigate the one-worker-many-tasks assignment for multiple workers and propose a bisection-based framework to divide

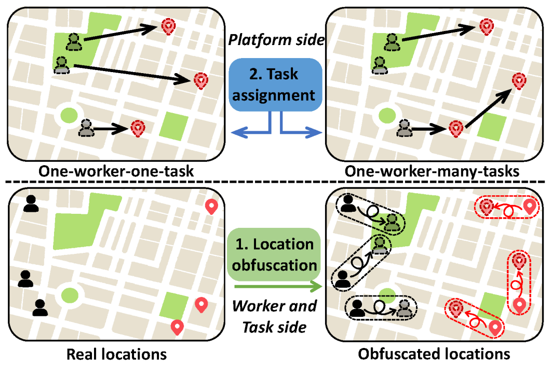


Fig. 2: Workflow of privacy-preserving task assignment.

workers and tasks into different partitions and assign tasks. However, both of [20], [21] consider the offline scenario and ignore privacy issues. Once in the privacy-preserving online scenario, their approaches fail to assign tasks effectively.

III. System model and Problem formulation

A. System Model

We consider a privacy-preserving online spatial crowdsourcing system model consisting of a platform, a large number of workers and tasks. The system evolves in a time-slotted fashion over T time slots, and temporary measures (e.g., seconds, minutes, or even days) can be used to represent the length of a time slot. Then, we introduce the three main parties and the workflow of the system model as follows:

(1) **Platform**. Without loss of generality, we assume that the platform is *honest-but-curious* [22], meaning that the platform will honestly carry out the task assignment but is curious about the private information (e.g., locations) of workers and tasks.

(2) **Worker**. As mentioned above, workers denoted by $\mathcal{W}$ dynamically arrive one by one in the online manner. Each worker $w_i \in \mathcal{W}$ is specified by a set of attributes $[\tau_a, \tau_l, \mathcal{C}, l_x, l_y]$, where $w_i.\tau_a, w_i.\tau_l$ represent the time slots in which w_i arrives and leaves. $w_i.\mathcal{C}$ represents the capacity (i.e., the number of tasks that can be assigned to w_i). In addition, $w_i.l_x, w_i.l_y$ represent the coordinates of w_i's location. For short, we use l to denote the location in the subsequent sections.

(3) **Task**. Similarly, tasks denoted by $\mathcal{T}$ also arrive dynamically one by one in the online manner, and each task $t_j \in \mathcal{T}$ is specified by a set of attributes $[\tau_a, \tau_l, l_x, l_y]$. Note that the task does not have the capacity attribute.

(4) **Workflow**. The workflow of the privacy-preserving task assignment is shown in Fig. 2. At first, workers $\mathcal{W}$ and tasks $\mathcal{T}$ arrive one by one in the online manner. Considering the privacy issues, both workers and tasks obfuscate their real locations to the obfuscated locations, by leveraging a privacy mechanism, to be introduced in Section IV. Fig. 2 (lower part) offers the location obfuscation process. For example, a worker or task's real location specified by (l_x, l_y) is changed to an obfuscated location specified by (l^*_x, l^*_y) ($l^* \in \mathbb{R}^2$ for short). Afterward, workers and tasks report their obfuscated locations to the platform for the subsequent task assignment. When workers or tasks arrive and report their obfuscated locations, the online privacy-preserving task assignment mechanism is invoked. As demonstrated in Fig. 2 (upper part), the platform assigns tasks according to the obfuscated locations. The assignment results are different in the one-worker-one-task and one-worker-many-tasks scenarios since a task could be assigned to a near worker even though the worker has been assigned with other tasks. In this paper, we focus on the one-worker-many-tasks assignment. Let $d(l, l')$ denote the distance between arbitrary two locations (workers and tasks' locations) l and l'. A worker w_i can be assigned multiple tasks denoted by $\mathcal{T}_i$ in a specific order, e.g., tasks $\mathcal{T}_i = \{t_1, t_2, t_3\}$ are assigned to w_i, then w_i's total travel distance is represented as $d_i = d(i, t_1) + d(t_1, t_2) + d(t_2, t_3)$. Note that a task can only be assigned once, i.e., $\mathcal{T}_i \cap \mathcal{T}_{i'} = \emptyset, \forall w_i, w'_i \in \mathcal{W}$. The notations used throughout the paper are described in Table I.

TABLE I: Notation Description

Notations	Description
$T, \mathcal{W}, \mathcal{T}$	Number of time slots, the worker set, the task set
w_i, t_j	Worker $w_i \in \mathcal{W}$, task $t_j \in \mathcal{T}$
τ_a, τ_l, l_x, l_y	Arrival time, leaving time, and location coordinates
$\mathcal{C}$	Number of tasks that a worker can perform
δ	Cardinality constraint to the number of assigned tasks
$\mathcal{M}$	Privacy mechanism
$d(l, l')$	Distance between any two locations l and l'
d_i	Total travel distance of worker i
$\mathcal{T}_i$	Tasks assigned to worker i
l, l^*, ϵ	Real and obfuscated location, the privacy budget
G, V, A	Network graph, the vertex set, the arc set
s, t	Origin vertex and destination vertex of the flow in G
$\mathcal{P}(G, s, t)$	Path in G from the source s to destination t
$\hat{d}(s, v)$	Distance of the shortest path from s to vertex v in G

B. Privacy Model

To prevent the platform from private information, we propose a privacy mechanism $\mathcal{M}$ to help workers and tasks obfuscate their real locations by adding the noise. Then, we offer the definitions regarding the privacy mechanism, privacy budget, and geo-indistinguishability as follows:

Definition 1 (Privacy mechanism)**.** *Given a metric space $\mathcal{L}$, a privacy mechanism $\mathcal{M}$ can map a real location $l \in \mathcal{L}$ to an obfuscated one $l^* \in \mathcal{L}$.*

Definition 2 (Privacy budget)**.** *Privacy budget ϵ indicates the privacy level that the privacy mechanism $\mathcal{M}$ can provide. A smaller ϵ denotes a stronger privacy level.*

Definition 3 (ϵ-Geo-Indistinguishability [23])**.** *A privacy mechanism $\mathcal{M}$ is said to satisfy Geo-Indistinguishability iff*

$$\mathcal{M}(l)(\mathcal{Z}) \le e^{\epsilon d(l,l')} \mathcal{M}(l')(\mathcal{Z}), \tag{1}$$

where $\mathcal{M}(l)(\mathcal{Z})$ and $\mathcal{M}(l')(\mathcal{Z})$ denote the probabilities that the obfuscated locations belong to $\mathcal{Z}$ when real locations are l and l', and $\mathcal{Z} \subseteq \mathcal{L}$ is the set of possible obfuscated locations. Note that $d(l, l') \le r$ denotes the Euclidean distance between arbitrary two locations l and l'.

C. Problem Formulation

Privacy-preserving Online Task Assignment (POTA) Problem: Given a set of workers $\mathcal{W}$ and tasks $\mathcal{T}$ that arrive one by one in real time and an untrusted platform,

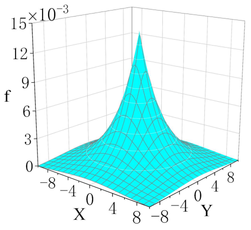


Fig. 3: The planar Laplace distribution centered at l_0.

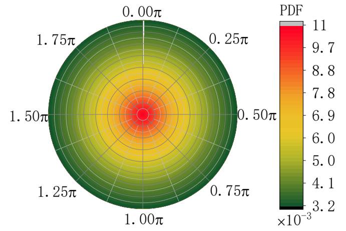


Fig. 4: The polar Laplace distribution with origin in l_0.

POTA aims to design a mechanism including two parts: (1) design a privacy mechanism to obfuscate workers and tasks' real locations, satisfying ϵ-Geo-Indistinguishability; (2) design an online one-worker-many-tasks assignment mechanism to assign tasks to minimize the total travel distance of workers under the assigned task cardinality constraint.

$$\text{Minimize} \quad \sum\nolimits_{w_i \in \mathcal{W}} d_i \tag{2}$$

$$\text{Subject to} \quad \sum\nolimits_{w_i \in \mathcal{W}} |\mathcal{T}_i| \geq \delta \tag{3}$$

$$|\mathcal{T}_i| \leq w_i.\mathcal{C}, \ \forall w_i \in \mathcal{W}. \tag{4}$$

As mentioned above, the goal of POTA is to minimize the total travel distance of all workers, as shown in Eq. (2). Note that $d_i = 0$ if no task is assigned to worker w_i, i.e., $|\mathcal{T}_i| = 0$. The first constraint indicates that the number of all assigned tasks is not less than δ, called the cardinality constraint. The second constraint indicates the number of assigned tasks for w_i should be limited by w_i's capacity, i.e., the capacity constraint. More importantly, the platform has to use the obfuscated locations of workers and tasks generated by the privacy mechanism $\mathcal{M}$ to assign tasks, called the privacy constraint.

IV. Mechanism Design

In this section, we introduce the privacy mechanism and online privacy-preserving task assignment mechanism in detail.

A. The privacy mechanism

In this section, we design the privacy mechanism in POTA. To protect the location privacy of workers and tasks, we prefer to hide their real locations and use the obfuscated locations to conduct the online task assignment. However, it will inevitably affect the assignment effect, i.e., there is a tradeoff between privacy protection and online task assignment. To balance the tradeoff, we propose a privacy mechanism called PLP to obfuscate workers and tasks' real locations based on the planar Laplace distribution (PLD) [23]–[26]. In the subsequent sections, PLP is proven to achieve the privacy guarantee and availability of obfuscated locations. Intuitively, the idea of PLP is that for each real location $l \in \mathbb{R}^2$, which can also be represented as (l_x, l_y), PLP randomly generates an obfuscated location $l^* \in \mathbb{R}^2$ around l by adding the noise generated by PLD. Specifically, given the privacy budget $\epsilon \in \mathbb{R}$ and real location l, we offer PLD's probability density function (pdf):

$$f(l, l^*, \epsilon) = \frac{\epsilon^2}{2\pi} e^{-\epsilon \cdot d(l, l^*)}. \tag{5}$$

Fig. 3 shows an example of the pdf centered at the real location $l_0 = (0, 0)$. We can discover that a standard Laplace distribution can be projected through the plane passing l_0. Afterward, we introduce how to draw the noise based on PLD. For convenience, we switch Eq. (5) to the polar coordinate system with the origin of l_0 in Fig. 4 and get Eq. (6).

Algorithm 1: PLP

```
    Input: Privacy budget $\epsilon$, real location $l = (l_x, l_y)$
    Output: Obfuscated location $l^* = (l_x^*, l_y^*)$
1   Draw $p \in [0,1]$ uniformly, $\rho = F^{-1}(p)$;
2   Draw $\theta \in [0, 2\pi]$ uniformly;
3   $l_x^* = l_x + \rho\cos(\theta)$, $l_y^* = l_y + \rho\sin(\theta)$;
4   return $l^* = (l_x^*, l_y^*)$
```

$$f(\rho, \theta, \epsilon) = \frac{\epsilon^2}{2\pi} e^{-\epsilon \cdot \rho}, \tag{6}$$

where ρ represents the distance between l and l^*, and θ represents the angle of the vector ll^* from the horizontal axis of the Cartesian coordinate system. Hence, the key to draw the noise from PLD is to determine the values of ρ and θ. To this end, we offer the cumulative probability function (cpf):

$$F(\rho, \epsilon) = \int_0^{\rho} \int_0^{2\pi} \frac{\epsilon^2}{2\pi} e^{-\epsilon\rho} d\theta d\rho = 1 - (1 + \epsilon\rho) e^{-\epsilon\rho}. \tag{7}$$

Based on Eq. (7), we determine the values of ρ and θ. For θ, since θ is independent with the cpf, we draw the value of θ randomly from $[0, 2\pi]$. For ρ, given $p = F(\rho, \epsilon)$ denoting the probability that $d(l, l^*) \leq \rho$, we offer the inverse function F^{-1} of cpf such that $\rho = F^{-1}(p, \epsilon)$. Since each probability p corresponds to a distance d, we randomly generate a value of $p \in [0, 1]$, and find the corresponding value of ρ based on the inverse function. The process is shown in Alg. 1. After getting ρ and θ, we can specify an obfuscated location l^* through adding the noise to the real location $l = (l_x, l_y)$ such that $l^* = (l_x + \Delta x, l_y + \Delta y) = (l_x + \rho\cos(\theta), l_y + \rho\sin(\theta))$. Next, we offer some properties of PLP as follows:

Theorem 1. *PLP achieves ϵ-Geo-Indistinguishability.*

Proof. Given arbitrary two real locations $l_1, l_2 \in \mathcal{L}$, and any an obfuscated point $l^* \in \mathcal{Z}$, we get

$$\begin{aligned} &\mathcal{M}(l_1)(l^*)/\mathcal{M}(l_2)(l^*) = f(l_1, l^*, \epsilon)/f(l_2, l^*, \epsilon) \\ &= e^{\epsilon \cdot (d(l_2, l^*) - d(l_1, l^*))} \leq e^{\epsilon \cdot d(l_1, d_2)}. \end{aligned} \tag{8}$$

The inequality holds because $d(l_2, l^*) - d(l_1, l^*) \leq d(l_1, l_2)$. Also, for each location in $\mathcal{Z}$, we can get the above property. Thus, we have $\mathcal{M}(l_1)(\mathcal{Z})/\mathcal{M}(l_2)(\mathcal{Z}) \leq e^{\epsilon \cdot d(l_1, d_2)}$. According to Definition 3, ϵ-Geo-Indistinguishability is achieved. □

B. Privacy-preserving online task assignment

In this section, we design the online task assignment mechanism in POTA. In the online scenario, workers and tasks arrive one by one in real time, so their locations are invisible to the platform until they arrive. Due to the lack of information about future workers and tasks, when an online task arrives, the platform has no assignment benchmark to decide whether to assign the task. Hence, we need to estimate a benchmark or threshold to help with the online task assignment. In addition, the online task assignment in POTA is the one-worker-many-tasks assignment. So, after deciding whether to assign the task based on the threshold, we still need to decide which worker to assign the task with considering the path planning of workers. To this end, we design a Threshold-based Online task Assignment mechanism (TOA). When TOA uses the obfuscated locations generated by PLP to assign tasks, we call it PLP-TOA. At first, PLP-TOA finds the optimal solution to the offline one-worker-many-tasks assignment based on the historical spatial-temporal data to estimate a threshold. Then, PLP-TOA conducts the online one-worker-many-tasks assignment based on the threshold.

1) Offline one-worker-many-tasks assignment: In the offline scenario, where the platform has the entire spatial-temporal information of workers and tasks in advance [27], [28], we aim to find the optimal solution to the offline one-worker-many-tasks assignment. However, considering both the task assignment and path planning makes it difficult to find the optimal solution directly. Inspired by the fact that the one-worker-one-task assignment is a special and simple case, we try to solve the one-worker-one-task assignment at first, then extend the approach to solve the one-worker-many-tasks assignment. To solve the one-worker-one-task assignment, we introduce its equivalent problem, i.e., the well-known minimum bipartite matching (MBM) problem in Definition 4.

Definition 4 (Minimum bipartite matching (MBM) problem [12])**.** *Given a set of workers $\mathcal{W}$ and tasks $\mathcal{T}$, the MBM problem aims to find a maximum cardinality one-to-one matching to minimize the total distance between pairs.*

MBM problem can be reducible to the minimum-cost flow (MCF) problem (also called the minimum-cost circulations problem) [9], [29], [30] in the network graph and solved by Bellman-Ford algorithm. Therefore, to solve the offline one-worker-one-task assignment, we could construct a special network graph based on workers and tasks' information and find the minimum-cost flow. Specifically, given workers $\mathcal{W} = \{w_1, \cdots, w_n\}$ and tasks $\mathcal{T} = \{t_1, \cdots, t_m\}$, let $G = (V, A)$ be the network graph as shown in Fig. 5, where $V = \mathcal{W} \bigcup \mathcal{T}$ is the vertex set and A is the arc set. We call $\mathcal{W}$ the worker vertices and $\mathcal{T}$ the task vertices. Note that each arc $(v, v') \in A$ contains two properties: capacity $c(v, v')$ and cost $d(v, v')$ (i.e., distance), and the flow in each arc is limited by its capacity. Then, we add a source vertex s and a sink vertex t to V and corresponding arcs to A. Also, we assign different capacity and cost to each arc in A. Formally, we offer the specific network graph construction process as follows:

$$\begin{gathered} V = \mathcal{W} \cup \mathcal{T} \cup \{s, t\}, \\ A = \{(s, w_i) \cup (w_i, t_j) \cup (t_j, t) | \forall w_i \in \mathcal{W}, \forall t_j \in \mathcal{T}\}, \\ c(s, w_i) = 1, d(s, w_i) = 0, \forall w_i \in \mathcal{W}, \\ c(w_i, t_j) = 1, d(w_i, t_j) = d(w_i, t_j), \forall w_i \in \mathcal{W}, \forall t_j \in \mathcal{T}, \\ c(t_j, t) = 1, d(t_j, t) = 0, \forall t_j \in \mathcal{T}. \end{gathered} \tag{9}$$

Note that $d(w_i, t_j)$ is the actual distance between worker w_i and task t_j. Then, let s be the origin of the flow and t the destination to find the minimum-cost flow based on Bellman-Ford algorithm. The minimum-cost flow is proven

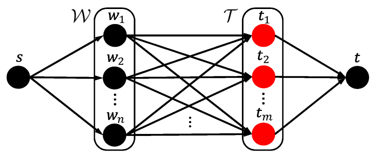


Fig. 5: Illustration of the MCF problem instance.

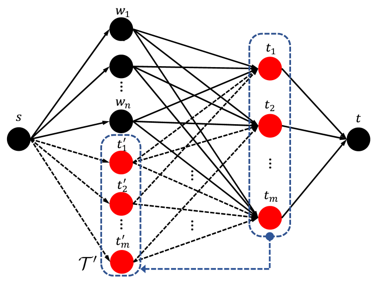


Fig. 6: Illustration of the EMCF problem instance.

to be the optimal solution to the MBM problem [31]. For instance, given a special MBM problem aiming to find three worker-task pairs while minimizing the total distance, we construct a network graph G as mentioned in Eq. (9) and solve the MCF problem to obtain the minimum-cost flow $\mathcal{F} = \{(s, w_1, t_2, t), (s, w_2, t_1, t), (s, w_3, t_3, t)\}$ with the cost of 4 and worker-task pair number of 3. Then, the optimal solution to the MBM problem are the worker-task pairs (w_1, t_2), (w_2, t_1) and (w_3, t_3), which are also the optimal solution to the offline one-worker-one-task assignment.

In summary, we construct a network graph and transfer the offline one-worker-one-task assignment to an MCF problem, then get the optimal solution to the offline one-worker-one-task assignment by finding the minimum-cost flow. However, as mentioned above, the approach cannot be used to solve the offline one-worker-one-task assignment because the platform has to consider not only the task assignment but also the path planning. For example, if two tasks t_1, t_2 are assigned to a same worker w_i, the total distances of different paths, such as $w_i \to t_1 \to t_2$ and $w_i \to t_2 \to t_1$, are different.

Therefore, to find the optimal solution to the one-worker-many-tasks assignment, we propose an extended network graph as shown in Fig. 6 by introducing the virtual worker vertices. The insight is that considering a worker's travel path such as $w_1 \to t_1 \to t_2$, it is easy to understand that a task actually becomes a new worker vertex if it is assigned to a worker. Specifically, given workers $\mathcal{W}$ and tasks $\mathcal{T}$, we copy the task vertices $\mathcal{T} = \{t_1, \cdots, t_m\}$ as the virtual worker vertices $\mathcal{T}' = \{t'_1, \cdots, t'_m\}$ as shown by the blue arrow in Fig. 6, then construct an extended network graph $G' = (V', A')$ based on G and add some new vertices and arcs to G':

$$\begin{gathered} V' = V \cup \mathcal{T}', \\ A' = A \cup \{(s, t'_i) | \forall t'_i \in \mathcal{T}'\} \cup \{(t'_i, t_j) | \forall t'_i \in \mathcal{T}', t_j \in \mathcal{T}\}, \\ c(s, t'_i) = 1, d(s, t'_i) = +\infty, \forall t'_i \in \mathcal{T}', \\ c(t'_i, t_j) = 1, d(t'_i, t_j) = d(t'_i, t_j), \forall t'_i \in \mathcal{T}', t_j \in \mathcal{T}, i \neq j, \\ c(t'_i, t_j) = 0, d(t'_i, t_j) = +\infty, \forall t'_i \in \mathcal{T}', t_j \in \mathcal{T}, i = j. \end{gathered} \tag{10}$$

In the initial G', the costs of the arcs between s and new

Algorithm 2: Extened minimum-cost flow (EMCF)

```
Input: Workers $\mathcal{W}$, tasks $\mathcal{T}$, cardinality constraint $\delta$
Output: The minimum-cost flow $\mathcal{F}$, total cost $\mathcal{D}$
1  Construct $G' = (V', A')$ as Eq. (10) based on $\mathcal{W}, \mathcal{T}$;
2  Initialize the flow $\mathcal{F} \leftarrow \emptyset$, total cost $\mathcal{D} \leftarrow 0$;
3  foreach $(w_i, t_j) \in A'$ do
4      if $w_i.t_l < t_j.t_a$ or $w_i.t_a > t_j.t_l$ then
5          Remove the arc $(w_i, t_j)$ from $A'$;
6  Find the minimum-cost augmenting path $\mathcal{P}(G', s, t)$;
7  while $\mathcal{P}(G', s, t)$ exists and $|\mathcal{F}| < \delta$ do
8      $\mathcal{F} \leftarrow \mathcal{F} \cup \mathcal{P}(G', s, t)$;
9      foreach arc $(v, v') \in \mathcal{P}(G', s, t)$ do
10         if arc $(v', v) \notin A'$ then
11             $A' \leftarrow A' \cup \{(v', v)\}$;
12             $d(v', v) \leftarrow -d(v, v')$, $c(v', v) \leftarrow 0$;
13         $c(v, v') \leftarrow c(v, v') - 1$, $c(v', v) \leftarrow c(v', v) + 1$;
14         $\mathcal{D} \leftarrow \mathcal{D} + d(v, v')$;
15         if $v' = t_j, \forall t_j \in \mathcal{T}$ then
16             $d(s, t'_j) \leftarrow 0$;
17     Find $\mathcal{P}(G', s, t)$ again based on the current $G'$;
18 return $\mathcal{F}$, $\mathcal{D}$
```

vertices $\mathcal{T}'$ are set to $+\infty$, meaning a path from s to t cannot pass through this arc at the beginning. Next, inspired by Bellman-Ford algorithm, we propose an Extended Minimum-Cost Flow (EMCF) algorithm to solve both task assignment and path planning in Alg. 2. To describe Alg. 2 clearly, we offer the definition of the augmenting path in Definition 5.

Definition 5 (Augmenting path). *A path $\mathcal{P}(G', s, t)$ from the source s to destination t in G' is called augmenting path if for any arc $(v, v') \in A'$, its current capacity $c(v, v') > 0$.*

In Alg. 2, given the worker set $\mathcal{W}$ and task set $\mathcal{T}$, we construct the extended network graph G' as shown in Eq. (10). Next, we set the initial values of the flow $\mathcal{F}$ as $\emptyset$ and the total distance $\mathcal{D}$ as 0 in line 2. The flow can be regarded as the set of paths from s to t. In lines 3-5, due to the scenario where both workers and tasks arrive one by one in real time, we eliminate the arcs in G' between workers and tasks that cannot exist at the same time, e.g., a task that exists during $8:00$~$9:00$ cannot be assigned to the worker that exists during $10:00$~$11:30$. After that, we find the augmenting path $\mathcal{P}(G', s, t)$ with minimum cost by leveraging the Shortest Path Faster Algorithm (SPFA) [32] in lines 6 and 17. If $\mathcal{P}(G', s, t)$ exists and the current assigned task number does not exceed the cardinality constraint δ in Eq. (3), Alg. 2 enters the loop part (lines 7-17). In this part, we first add the path $\mathcal{P}(G', s, t)$ to the flow. Then, for any two vertices v, v', if (v, v') exists in $\mathcal{P}(G', s, t)$, connect v' and v and add the reverse arc (v', v) to G'. In addition, we assign specific capacity and cost to the reverse arc. The reverse arc enables Alg. 2 to change the previous decision when the current path is chosen, so as to obtain the optimal solution effectively. Note that in lines 15-16, we change the cost of the

Algorithm 3: Threshold online assignment (TOA)

```
Input: $\mathcal{W}$, $\mathcal{T}$, $\delta$, $T$, $\phi$
Output: $\mathcal{F}$
1  $\mathcal{F} \leftarrow \emptyset$, $t \leftarrow 0$ ;
2  while $t \leq T$ and $|\mathcal{F}| < \delta$ do
3      $\mathcal{W}_t, \mathcal{T}_t \leftarrow$ the current workers and tasks in $t$;
4      $\mathcal{F}_t, \mathcal{D}_t = \text{EMCF}(\mathcal{W}_t, \mathcal{T}_t, \delta)$;
5      foreach $\mathcal{P} \in \mathcal{F}_t$ do
6          if $d(\mathcal{P}) \leq \kappa$ and the pre-path of $\mathcal{P}$ is in $\mathcal{F}$ then
7              $\mathcal{F} \leftarrow \mathcal{F} \cup \{\mathcal{P}\}$;
8      $t \leftarrow t + 1$;
9  return $\mathcal{F}$
```

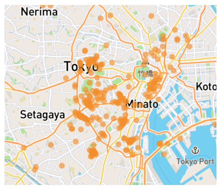
arc (s, t'_j) if the vertex t_j is in the path $\mathcal{P}(G', s, t)$, meaning that the arc (s, t'_j) is now passable since a worker needs to travel to t_j first, and then travel to other tasks from t_j. Note that introducing the passable arc (s, t'_j) may result in negative-cost circulations and we should cut these negative-cost circulations. Finally, Alg. 2 returns the flow $\mathcal{F}$ and total cost $\mathcal{D}$. Actually, the task assignment solution is $\mathcal{F}$, e.g., Alg. 3 returns a flow $\mathcal{F} = \{(s, w_1, t_2, t), (s, t'_2, t_1, t), (s, w_2, t_3, t)\}$, then the assignment solution is $w_1 \to t_2 \to t_1$ and $w_2 \to t_3$ with the total cost $\mathcal{D}$.

Theorem 2 (Optimality). *The flow returned by Alg. 2 is the minimum-cost flow with the cardinality constraint δ.*

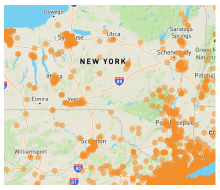
Proof. To prove the optimality, we offer the optimal condition of Alg. 2, that is, if there are no negative-cost paths from s to t in G', the flow returned by Alg. 2 is the minimum-cost flow [31]. Next, we prove that there are no negative-cost paths in G' when Alg. 2 terminates. First, we call a path $\mathcal{P}(G', s, t)$ the negative-cost path if its total cost $\sum_{(v,v')\in\mathcal{P}} d(v, v') < 0$. Note that in line 6 of Alg. 2, we utilize SPFA to find the current shortest path. Actually, SPFA will calculate the cost of the shortest path from the source s to an arbitrary vertex $v \in V'$ as the intermediate results, denoted as $\hat{d}(s, v), \forall v \in V'$. Thus, for each arc $(v, v') \in A'$ at the end of Alg. 2, we get $\hat{d}(s, v') \leq \hat{d}(s, v) + d(v, v')$. This is because if $\hat{d}(s, v') > \hat{d}(s, v) + d(v, v')$, indicating that $\hat{d}(s, v')$ is not the cost of the shortest path from s to v', then SPFA will not terminate. In other words, if SPFA terminates, we always have $\hat{d}(s, v') \leq \hat{d}(s, v) + d(v, v')$. In addition, for an arbitrary path $\mathcal{P}(G', s, t)$, the cost is calculated as

$$\sum\nolimits_{(v,v')\in\mathcal{P}} d(v, v') \geq \sum\nolimits_{(v,v')\in\mathcal{P}} (\hat{d}(s, v') - \hat{d}(s, v)) = \hat{d}(s, t).$$

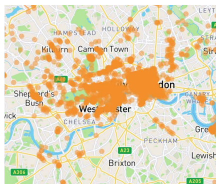
Clearly, $\hat{d}(s, t) \geq 0$. Hence, there are no negative-cost paths in G' at the end of Alg. 2. Also, the cost of every s-t path is greater than $\hat{d}(s, t)$. It is worth noting that adding the task vertices $\mathcal{T}'$ in Eq. (10) and changing the costs of the arcs between s and each vertex in $\mathcal{T}'$ do not affect the optimality of Alg. 2 due to the following two reasons: (1) The change on G' is considered for the previous augmenting paths. Reverse arcs provide Alg. 2 with opportunities to change the previously chosen augmenting paths. For example, in the current iteration, we have the flow $\mathcal{F} = \{(s, w_1, t_1, t), (s, t'_1, t_2, t)\}$, meaning



(a) Tokyo



(b) New York



(c) London

Fig. 7: The spatial-temporal distributions of the three cities.

that w_1 travels to t_1, t_2 in sequence. In the next iteration, we have $\mathcal{F} = \{(s, w_1, t_3, t), (s, t_1', t_2, t), (s, w_2, t_1, t)\}$, meaning that w_2 replaces w_1 to travel to t_1, t_2, and w_1 travels to a new task t_3. (2) The change on G' is considered for the afterward augmenting paths. As mentioned above, it has been proven that at the end of Alg. 2, we get the minimum-cost flow with new task vertices $\mathcal{T}'$. Above all, the optimality is proved. $\square$

Based on the flow $\mathcal{F}$ obtained by Alg. 2, we can estimate the threshold $\kappa = min\{d(\mathcal{P})\}, \forall \mathcal{P} \in \mathcal{F}$, where $\mathcal{P}$ is the s-t path, and $d(\mathcal{P}) = \sum_{(v,v')\in\mathcal{P}} d(v, v')$ is the total cost of $\mathcal{P}$, e.g., if $\mathcal{P} = \{s, w_1, t_2, t\}$, then $d(\mathcal{P}) = d(s, w_1) + d(w_1, t_2) + d(t_2, t) = d(w_1, t_2)$ since $d(s, w_1) = d(t_2, t) = 0$.

2) Online one-worker-many-tasks assignment: After determining the threshold κ, we propose a Threshold-based Online task Assignment (TOA) algorithm in Alg. 3 to assign tasks in the online manner. We input workers $\mathcal{W}$, tasks $\mathcal{T}$, cardinality constraint δ, total time slot number T and threshold κ, and get the final task assignment solution $\mathcal{F}$. Specifically, in each time slot t, we get the current workers and tasks, including those who have just arrived or not yet left, and conduct EMCF algorithm (i.e., Alg. 2) to obtain the current minimum-cost flow $\mathcal{F}_t$. After that, we filter the paths in $\mathcal{F}_i$ and add specific paths to $\mathcal{F}$ based on the threshold in lines 5-7. Note that the pre-path in line 6 means that only when a path (s, w_1, t_1, t) is contained in $\mathcal{F}$, then the path (s, t_1', t_2, t) can be added. At the end of Alg. 3, the flow $\mathcal{F}$ returned is actually the task assignment solution. Furthermore, the theoretical guarantee of the online assignment algorithms is usually measured by Competitive Ratio (CR), and the definition of CR is given as follows according to [30]:

Definition 6 (CR)**.** *The competitive ratio of an online algorithm $\mathcal{A}$ in the random order model is defined as*

$$CR = E(\mathcal{A})/OPT, \tag{11}$$

where $E(\mathcal{A})$ is the expected total distance produced by the online algorithm $\mathcal{A}$ over all possible arrival orders, and OPT is the total distance produced by the offline algorithm.

We claim that TOA achieves a performance guarantee and offer the CR of TOA in Theorem 3.

Theorem 3. *TOA achieves the CR as $\frac{\phi\cdot\delta}{d(\mathcal{O}')-\sigma}$.*

Proof. Let $\mathcal{O}$, $\mathcal{O}'$ denote the optimal solutions to the online task assignment shown in Alg. 3 and offline task assignment based on historical spatial-temporal data in Alg. 2, respectively. Obviously, the costs of $\mathcal{O}$, $\mathcal{O}'$ have the same distribution [17], [30]. According to Chebyshev inequality, we have

$$Pr[|d(\mathcal{O}) - E(d(\mathcal{O}'))| \leq \varepsilon] \geq 1 - \sigma^2/\varepsilon^2, \tag{12}$$

where $E(d(\mathcal{O}'))$ and σ represents the expectation and standard deviation of $d(\mathcal{O})$, and $\varepsilon \in \mathbb{R}^+$. Hence, we get

$$\begin{aligned} CR &= E(TOA)/OPT = E(\delta\cdot\Delta)/d(\mathcal{O}) = \delta E(\Delta)/d(\mathcal{O}) \\ &\leq \phi\cdot\delta/d(\mathcal{O}) \leq \phi\cdot\delta/(1-\sigma^2/\varepsilon^2)\cdot(d(\mathcal{O}')-\varepsilon). \end{aligned} \tag{13}$$

Then, $\lim_{\varepsilon\to\sigma} CR = \phi\cdot\delta/(d(\mathcal{O}')-\sigma)$, proved. $\square$

When the platform uses TOA to assign tasks based on the obfuscated locations generated by PLP, we call the algorithm PLP-TOA. While protecting the privacy of workers and tasks, we prove that PLP-TOA still achieves a performance guarantee. At first, we give the following lemma according to [26].

Lemma 1 (λ-Distance Aggregation Error)**.** *Given the privacy budget ϵ and a real $\lambda \in \mathbb{R}^+$, we claim that the privacy mechanism PLP achieves λ-distance aggregation error, i.e.,*

$$Pr[|d(\tilde{\mathcal{F}}) - d(\mathcal{F})| \geq \lambda] \geq \frac{6}{\lambda^2\cdot\epsilon^2}, \tag{14}$$

where $d(\tilde{\mathcal{F}}), d(\mathcal{F})$ are the aggregated distances returned by PLP-TOA and TOA, respectively.

Furthermore, we give the CR of PLP-TOA in Theorem 4.

Theorem 4. *PLP-TOA achieves the CR of $\frac{6(\kappa\cdot\delta+\lambda)}{\lambda^2\epsilon^2(d(\mathcal{O}')-\sigma)}$.*

Proof. Given Theorem 3 and Lemma 1, we have

$$CR = d(\tilde{\mathcal{F}})/OPT \leq \frac{6(d(\mathcal{F})+\lambda)}{d(\mathcal{O})\cdot\lambda^2\epsilon^2} \leq \frac{6(\kappa\cdot\delta+\lambda)}{\lambda^2\epsilon^2(d(\mathcal{O}')-\sigma)},$$

where κ is the threshold determined by Alg. 2, δ the cardinality constraint, $\lambda \in \mathbb{R}^+$ the distance aggregation error, ϵ the privacy budget and σ the standard deviation of $d(\mathcal{O})$. Hence, the CR can also be represented as $O(1/(\bar{d}\cdot\epsilon^2))$, where $\bar{d} = d(\mathcal{O}')/\delta$ is the average cost of performing a task in $\mathcal{O}'$, meaning that lower average cost and privacy level lead to a better CR. $\square$

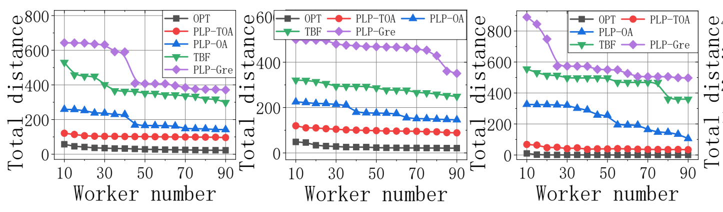
TABLE II: Simulation Parameters

Parameters	Value
Time length T	100
Worker number $\lvert\mathcal{W}\rvert$	$10 \sim 90$
Task number $\lvert\mathcal{T}\rvert$	$50 \sim 250$
Privacy budget ϵ_1	$0.1 \sim 2.1$
Cardinality δ	$5 \sim 45$
Capacity $\mathcal{C}$	$1 \sim 10$

V. Simulations

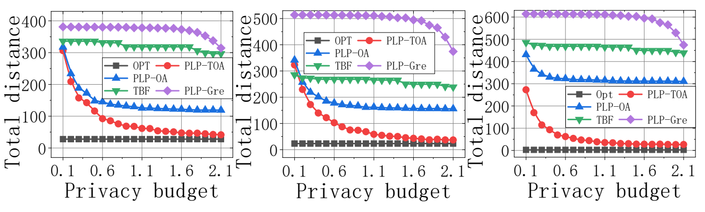
A. Dataset and Settings

In the simulations, we adopt two widely-used real-world datasets, Gowalla [33] and Foursquare [34]. The datasets contain a large amount of check-in data of workers over years such as worker id, check-in location, and check-in time. From datasets, we mainly select three cities: Tokyo, New York, and London as shown in Fig. 7, where the orange nodes represent workers' real check-in locations. Then, we determine the simulation settings as follows. For workers, we select some workers $\mathcal{W}$ randomly from the datasets and $|\mathcal{W}| \in [10, 90]$. We take a period of time in datasets and normalize it to $T = 100$ time slots, each of which lasts approximately one day. Then, each worker's arrival time τ_a is determined by the check-in



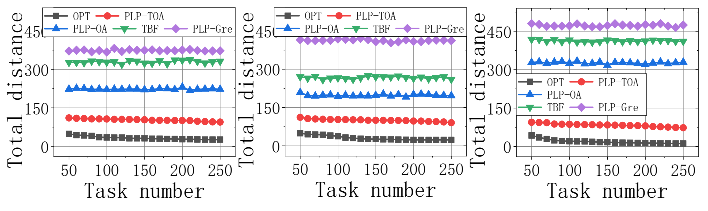
(a) Tokyo (b) New York (c) London

Fig. 8: Total distance vs. Worker number.



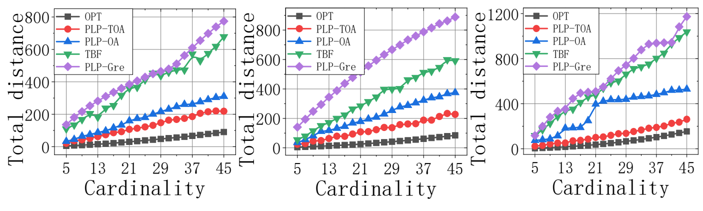
(a) Tokyo (b) New York (c) London

Fig. 10: Total distance vs. Privacy budget.



(a) Tokyo (b) New York (c) London

Fig. 9: Total distance vs. Task number.



(a) Tokyo (b) New York (c) London

Fig. 11: Total distance vs. Cardinality.

time. Hence, the leaving time $\tau_l = \tau_a + \varrho$. In addition, we take some popular check-in locations as tasks $|\mathcal{T}| \in [50, 250]$ and set the tasks' arrival time in a similar manner. The values of key simulation parameters are presented in Table II. The default values of key simulation parameters are as follows: $|\mathcal{W}| = 50$, $|\mathcal{T}| = 200$, $\epsilon = 0.6$, $\delta = 20$, and $\mathcal{C} = 1$.

B. Metrics and Baselines

Total distance and competitive ratio (CR) are the key metrics to evaluate the performance of online assignment algorithms. Then, we mainly introduce some baselines as follows:

(1) PLP-TOA, the proposed algorithm that uses PLP to protect the location privacy and uses TOA algorithm to assign tasks. Note that PLP-TOA is the one-worker-many-tasks algorithm, i.e., the worker has the capacity $\mathcal{C} \geq 1$.

(2) TBF [17], the state-of-the-art algorithm proposed to protect privacy and assign tasks based on a tree structure called HSTs. TBF is the one-worker-one-task algorithm.

(3) PLP-OA, the algorithm similar to PLP-TOA and designed to compare with TBF. When a task arrives, PLP-OA does not filter tasks based on the threshold, but only decides which worker to assign the task. We adjust PLP-OA to the one-worker-one-task algorithm, i.e., $\mathcal{C} = 1$.

(4) OPT, the algorithm to find the offline optimal solution without privacy protection, i.e., the offline-version TOA.

(5) PLP-Gre, the algorithm that uses PLP to protect the location privacy and assign tasks greedily, i.e., assign tasks to the nearest worker when online tasks arrive. Also, $\mathcal{C} = 1$.

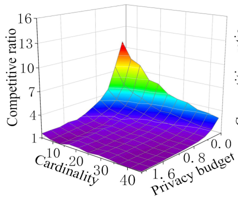
Note that the state-of-the-art algorithm TBF can only be applied in a restricted scenario where (1) tasks arrive dynamically but workers are fixed and known in advance, (2) a worker can only perform a task, and (3) when tasks arrive, TBF will not filter tasks and discard inappropriate ones, but assign each arrival task to workers until the cardinality constraint is met. Actually, the restricted scenario is a special case of the scenario considered in this paper. Hence, to verify the performance of our algorithm fairly, we should compare them with TBF in the same scenario by adjusting our algorithm. When our algorithm still outperforms TBF, the following holds: (1) our algorithm supports more general scenarios; (2) even though in the same restricted scenario, our algorithm is still better than TBF.

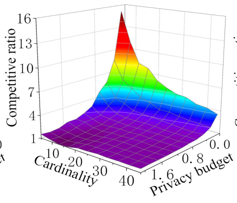
C. Evaluation Results

1) Total distance: At first, we illustrate the simulation results in terms of the main metric, i.e., the total distance as shown in Figs. 8~11. In Fig. 8, we observe the total distance with the worker number changing from 10 to 90 when other parameters remain default. The total distances of all algorithms gradually decrease when the worker number rises because the platform could assign the task to a more suitable worker when more workers are available. OPT outperforms best as it should be. Then, we discover that PLP-TOA outperforms better than PLP-OA because when tasks arrive, PLP-TOA will filter tasks based on the threshold and select specific tasks to assign, e.g., if a task t_j is far away from all workers, then PLP-TOA will abandon this task. However, PLP-OA will assign all tasks no matter the task is good or not. In addition, compared with PLP-OA, PLP-TOA is the one-worker-one-task assignment and can select a suitable worker to perform multiple tasks. Hence, PLP-TOA is better than PLP-OA. More importantly, in the same condition, it always follows that PLP-OA>TBF>PLP-Gre, showing that our algorithm outperforms the state-of-the-art algorithm, and both of them outperform the greedy algorithm.

In Fig. 9, we increase the task number from 50 to 250, and the total distances of OPT and PLP-TOA marginally decrease due to the similar reason as Fig. 8, i.e., more tasks lead to a higher probability to select good tasks. However, it shows that the total distances of other algorithms are basically unchanged because these algorithms only select the first δ tasks to assign, no matter how many tasks will come next. In addition, our algorithm PLP-OA still outperforms TBF.

In Fig. 10, given the privacy budget from 0.1 to 2.1, the total distances of all algorithms decrease except that OPT remains unchanged. This is because OPT does not consider privacy protection and uses the real locations to assign tasks, thus the total distance does not change. However, for other algorithms that consider privacy protection, as the privacy budget becomes larger, meaning that the level of privacy protection becomes


(a) Tokyo


(b) New York

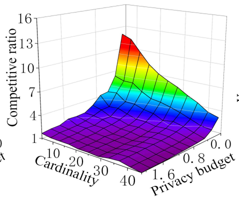
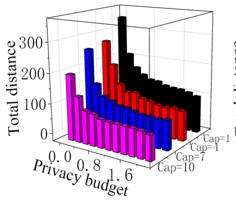
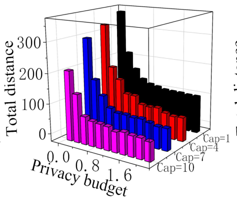

(c) London

Fig. 12: Competitive ratio.


(a) Tokyo


(b) New York

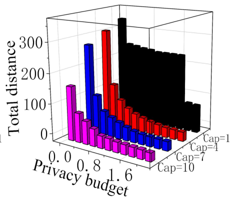

(c) London

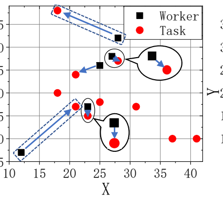
Fig. 13: Total distance vs. Capacity.

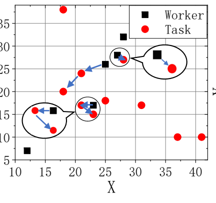
lower, the obfuscated locations are closer to the real locations according to Alg. 1, hence the task assignment effect is better.

Furthermore, we evaluate the effect of the cardinality constraint in Eq. (3) on the total distance as depicted in Fig. 11. With the increase of the cardinality, the total distances of all algorithms grow rapidly, indicating that the cardinality is the main factor that affects the performance. Also, our algorithms are still better than TBF, and greedy based algorithm has the worst performance. In summary, with the change of various parameters, the simulation of the total distance illustrate that (1) PLP-TOA and PLP-OA outperform the state-of-the-art algorithm TBF in the same scenario, and (2) in the general scenario, PLP-TOA achieves the close performance to OPT.

2) Competitive ratio: Next, we verify the performance in items of another essential metric, i.e., the competitive ratio (CR). In Fig. 12, we calculate the CR of our algorithm PLP-TOA based on the three cities according to Eq. (11). With the decline of the privacy budget from 1.6 to 0.1 and the cardinality from 40 to 10, the CR shows an upward trend. Note that when the cardinality increases, the average cost of the offline optimal solution, i.e., $\bar{d}$, decreases gradually because the distance function $d(\mathcal{O}')$ is actually non-submodular with respect to the number of assigned tasks, i.e., the cardinality. Therefore, we can say that CR shows an upward trend as the privacy budget ϵ and average cost $\bar{d}$ decrease, which is exactly consistent with the theoretical guarantee $O(1/(\bar{d}\cdot\epsilon^2))$ that has been proved in Theorem 4. In addition, the result also demonstrates when the privacy budget and cardinality are not too small, the value of CR is very close to 1, indicating the performance of PLP-TOA is close to the optimal solution.

3) One-worker-many-tasks assignment: Compared with the one-worker-one-task assignment, the one-worker-many-tasks assignment is more realistic. To evaluate the difference between them, we first evaluate the effect of the worker capacity in Fig. 13. The results show that the total distance decline with the increase of the capacity from 1 to 10. Furthermore, we perform a specific simulation based on the check-in data in New York and offer specific examples with the capacity $\mathcal{C}$ from 1 to 3. In order to show the examples intuitively,


(a) Capacity=1


(b) Capacity=2

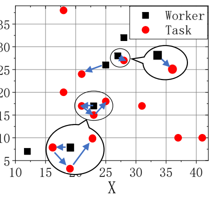

(c) Capacity=3

Fig. 14: Examples of the one-worker-many-tasks assignment.

we normalize and mesh the real locations of workers and tasks to $l_x, l_y \in [1, 100]$ and set the parameters as follows: task number $|\mathcal{T}| = 10$, worker number $|\mathcal{W}| = 5$, cardinality constraint $\delta = 5$. Then, the examples are illustrated in Fig. 14, we discover that when $\mathcal{C} = 1$, the case is actually the one-worker-one-task assignment scenario, where each worker only performs a task and the total distance is 33.0018. When $\mathcal{C} = 2$, some workers perform two tasks near them and the total distance is effectively reduced to 15.7147. Note that the two worker-task pairs in the dotted rectangle in Fig. 14a disappear in Fig. 14b because when $\mathcal{C} = 1$, to satisfy the cardinality constraint, tasks have to be assigned to the workers who are far away. But when $\mathcal{C} = 2$, tasks have the better worker to select even though the worker has been assigned. When $\mathcal{C} = 3$ in Fig. 14c, we discover that five tasks are assigned to three popular workers, i.e., those at the center of all tasks. In addition, we get the lowest total distance 14.3203. It is reasonable since when we relax the restriction on the capacity, our algorithm will assign tasks to the workers who have location advantages to minimize the total distance, illustrating its effectiveness in the one-worker-many-tasks assignment scenario.

VI. Conclusion

In this paper, we propose a privacy-preserving online task assignment framework to assign tasks to workers for minimizing the total travel distance under the assigned task cardinality constraint. Specifically, to protect privacy, we present a Planar Laplace distribution based Privacy mechanism (PLP), which achieves ϵ-Geo-Indistinguishability and ensures the availability of the obfuscated locations. Then, we propose a Threshold-based Online task Assignment (PLP-TOA) algorithm using the obfuscated locations to conduct the one-worker-many-tasks assignment. Specifically, at first, based on the historical data, we formulate the task assignment as an Extended Minimum-Cost Flow problem (EMCF) and estimate a threshold. Furthermore, based on the threshold and EMCF, the platform not only assigns tasks but also plans the path for workers in the online scenario. Finally, results of extensive simulations based on real-world datasets illustrate that our algorithm consistently outperforms the state-of-the-art approach.

Acknowledgment

This work is supported by the National Key R&D Program of China under Grant 2021ZD0112501 and Grant 2021ZD0112502, the National Natural Science Foundations of China under Grant No. 61772230, No. 61972450 and No. 62072209, CCF- Baidu Open Fund(NO.2021PP15002000), and in part by NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1629746, CNS 1651947, and CNS 1564128.

References

[1] L. Kazemi and C. Shahabi, "Geocrowd: enabling query answering with spatial crowdsourcing," in *Proc. ACM SIGSPATIAL*, 2012, pp. 189–198.

[2] Y. Tong, L. Chen, and C. Shahabi, "Spatial crowdsourcing: Challenges, techniques, and applications," *PVLDB*, vol. 10, no. 12, pp. 1988–1991, 2017.

[3] Y. Liu, B. Guo, H. Du, Z. Yu, D. Zhang, and C. Chen, "Poster: Foodnet: Optimized on demand take-out food delivery using spatial crowdsourcing," in *Proc. ACM MobiCom*, 2017, pp. 564–566.

[4] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma, "Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation," in *Proc. ACM WWW*, 2017, pp. 627–636.

[5] H. To, C. Shahabi, and L. Xiong, "Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server," in *Proc. IEEE ICDE*, 2018, pp. 833–844.

[6] Z. Wang, J. Li, J. Hu, J. Ren, and Y. Li, "Towards privacy-preserving incentive for mobile crowdsensing under an untrusted platform," in *Proc. IEEE INFOCOM*, 2019.

[7] J. Ni, K. Zhang, Q. Xia, X. Lin, and X. S. Shen, "Enabling strong privacy preservation and accurate task allocation for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1317–1331, 2020.

[8] B. Zhao, S. Tang, X. Liu, X. Zhang, and W. Chen, "itam: Bilateral privacy-preserving task assignment for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[9] L. Hou, L. Yiu, K. Mouratidis, and N. Mamoulis, "Capacity constrained assignment in spatial databases," in *Proc. ACM SIGMOD*, 2008, pp. 15–28.

[10] 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.

[11] C. Dai, X. Wang, K. Liu, D. Qi, W. Lin, and P. Zhou, "Stable task assignment for mobile crowdsensing with budget constraint," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[12] Y. Tong, J. She, B. Ding, L. Chen, T. Wo, and K. Xu, "Online minimum matching in real-time spatial data: Experiments and analysis," *PVLDB*, vol. 9, no. 12, pp. 1053–1064, 2016.

[13] W. Liu, Y. Yang, E. Wang, and J. Wu, "Dynamic user recruitment with truthful pricing for mobile crowdsensing," in *Proc. IEEE INFOCOM*, 2020, pp. 1113–1122.

[14] E. Wang, H. Wang, Y. Yang, and W. Liu, "Truthful incentive mechanism for budget-constrained online user selection in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[15] Z. Shi, S. Jiang, L. Zhang, Y. Du, and X.-Y. Li, "Crowdsourcing system for numerical tasks based on latent topic aware worker reliability," in *Proc. IEEE INFOCOM*, 2021.

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

[17] Q. Tao, Y. Tong, Z. Zhou, Y. Shi, L. Chen, and K. Xu, "Differentially private online task assignment in spatial crowdsourcing: A tree-based approach," in *Proc. IEEE ICDE*, 2020.

[18] A. O. Al-Abbasi, A. Ghosh, and V. Aggarwal, "Deeppool: Distributed model-free algorithm for ride-sharing using deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4714–4727, 2019.

[19] M. Xiao, G. Gao, J. Wu, S. Zhang, and L. Huang, "Privacy-preserving user recruitment protocol for mobile crowdsensing," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 519–532, 2020.

[20] D. Deng, C. Shahabi, and U. Demiryurek, "Maximizing the number of worker's self-selected tasks in spatial crowdsourcing," in *Proc. ACM SIGSPATIAL GIS*, 2013, pp. 314–323.

[21] D. Deng, C. Shahabi, and L. Zhu, "Task matching and scheduling for multiple workers in spatial crowdsourcing," in *Proc. ACM SIGSPATIAL GIS*, 2015, pp. 21:1–21:10.

[22] G. Zhuo, Q. Jia, L. Guo, M. Li, and P. Li, "Privacy-preserving verifiable data aggregation and analysis for cloud-assisted mobile crowdsourcing," in *Proc. IEEE INFOCOM*, 2016, pp. 1–9.

[23] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *Proc. ACM CCS*, 2013, pp. 901–914.

[24] K. Lange and J. S. Sinsheimer, "Normal/independent distributions and their applications in robust regression," *Journal of Computational and Graphical Statistics*, vol. 2, no. 2, pp. 175–198, 1993.

[25] H. To, G. Ghinita, and C. Shahabi, "A framework for protecting worker location privacy in spatial crowdsourcing," *PVLDB*, vol. 7, no. 10, pp. 919–930, 2014.

[26] Y. Liu, T. Feng, M. Peng, Z. Jiang, Z. Xu, J. Guan, and S. Yao, "COMP: online control mechanism for profit maximization in privacy- preserving crowdsensing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1614–1628, 2020.

[27] L. Hou, K. Mouratidis, L. Yiu, and N. Mamoulis, "Optimal matching between spatial datasets under capacity constraints," *ACM Transactions on Database Systems*, vol. 35, no. 2, pp. 9:1–9:44, 2010.

[28] C. Long, R. C. Wong, P. S. Yu, and M. Jiang, "On optimal worst-case matching," in *Proc. ACM SIGMOD*, 2013, pp. 845–856.

[29] A. V. Goldberg and R. E. Tarjan, "Finding minimum-cost circulations by successive approximation," *Mathematics of Operations Research*, vol. 15, no. 3, pp. 430–466, 1990.

[30] J. Liu and K. Xu, "Budget-aware online task assignment in spatial crowdsourcing," vol. 23, no. 1, 2020, pp. 289–311.

[31] R. Ahuja, T. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*, 2013.

[32] E. F. Moore, "The shortest path through a maze," in *Proc. Internat. Sympos. Switching Theory 1957, Part II*, 1959, pp. 285–292.

[33] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proc. ACM SIGKDD*, 2011, pp. 1082–1090.

[34] D. Yang, D. Zhang, V. W. Zheng, and Z. Yu, "Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 129–142, 2015.