Information Sciences 619 (2023) 654–678

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

MAPP: An efficient multi-location task allocation framework with personalized location privacy-protecting in spatial crowdsourcing

Yu Fan [a,b], Liang Liu [a,b,*], Xingxing Zhang [a,b], Huibin Shi [a,b], Wenbin Zhai [a,b]

[a] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
[b] Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China, Tianjing, China

ARTICLE INFO

Article history:
Received 14 June 2022
Received in revised form 9 October 2022
Accepted 14 November 2022
Available online 21 November 2022

Keywords:
Spatial crowdsourcing
Task allocation
Privacy protection
Personalized location privacy protection
Multi-location task

ABSTRACT

Due to its wide coverage and strong scalability, spatial crowdsourcing (SC) has become a research hotspot in recent years. In order to assign tasks to closer workers, it is necessary for workers to provide accurate locations to the server. However, it will result in the leakage of the participants' location privacy. Existing works provide each worker with the same level of location privacy protection, which cannot meet the different privacy requirements of various workers. In addition, most works assume that the tasks are single-location tasks, and do not consider multi-location tasks. In this paper, we propose the Multi-location Task Allocation Problem with personalized location privacy protection (MLTAP). As far as we know, we are the first to study MLTAP. We propose a Multi-location task Allocation framework with Personalized location Privacy-protecting (MAPP). In order to allocate tasks efficiently, we use the R-tree to store workers and minimum bounding rectangle to represent multi-location tasks, thus filtering the unreachable workers for tasks. To better eliminate the adverse effect of location confusion, the SC server sorts candidate workers by the ranking metrics and allocates multi-location tasks efficiently. Finally, we conduct experiments to verify that MAPP has good performance in terms of utility.

1. Introduction

With the rapid development of wireless networks, as well as the increasing communication and computing capabilities of mobile devices, a new form of crowdsourcing – spatial crowdsourcing (SC) has emerged [1]. SC has been widely used in traffic monitoring, news reporting, and so on.

As shown in Fig. 1, after workers and task publishers register with the SC server, workers need to upload their real-time location information. Task publishers send some spatial tasks to the SC server with instructions, available resources, remuneration information, and so on. For example, the transportation authority publishes some tasks about collecting traffic data in different places. After receiving tasks, the SC server calculates the proximity between workers and each task, and then assigns it to the appropriate worker. The worker goes to the specified location and performs the task according to its requirement. When the task is completed, the worker sends the execution results to the task publisher by uploading them to the SC server. Finally, the task publisher pays the salary to the worker via the SC server [2].

* Corresponding author at: College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.
E-mail address: liangliu@nuaa.edu.cn (L. Liu).

https://doi.org/10.1016/j.ins.2022.11.075
0020-0255/

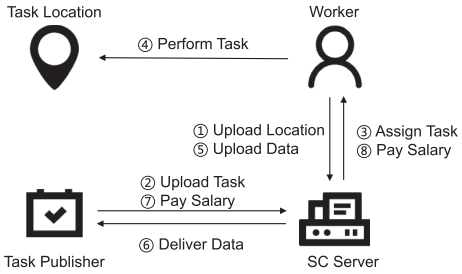

Fig. 1. Spatial crowdsourcing workflow.

Compared with the traditional wireless sensor networks (WSNs), SC has many advantages. The SC platform saves additional costs in installing and maintaining large scale hardware infrastructure and provides broader coverage than WSN. In recent years, SC has been extensively studied and applied in academia and industry, such as WiFiManager [3], an SC platform that collects Wi-Fi information using Android devices across 200 + countries/regions.

How to allocate tasks efficiently is an essential problem in SC. Some studies [4] aim to increase the number of assignments as many as possible during the task allocation process and reduce the total distance traveled by workers, so that SC platforms can improve the quality and effectiveness of their services. To efficiently assign tasks, workers are usually required to expose their accurate location to the SC server. Then the SC server assigns the tasks to the near workers.

However, the SC server can not always be fully trusted. Based on the location information uploaded by the worker, the SC server and other SC participants can deduce the sensitive information of workers, such as identity, address, etc. Once the location information is compromised by the untrusted SC server, the workers' privacy will be seriously threatened [5]. Adversaries can launch attacks from such information as surveillance, tracking, identity theft, etc. Workers may not be willing to upload their true location information or even quit the SC service. Therefore, not only allocating tasks efficiently but also protecting the worker's location privacy is essential to SC.

In recent years, there have been some studies [6,7] on location privacy protection in SC. In [6], Qiu et al. take the dynamic traffic conditions and network topology into consideration. The weighted directed graph is used to describe the movement of workers. Based on the graphical model, they design a geo-obfuscation (GO) function for workers to maximize the location privacy of workers and formulate the problem of obtaining the best GO function into a linear programming problem. Wang et al. propose a location privacy-preserving task allocation framework with geo-obfuscation [7]. Workers obfuscate their locations under the guarantee of differential and distortion privacy. In order to achieve optimal task allocation with the differential-and-distortion geo-obfuscation, they formulate a mixed-integer non-linear programming problem to minimize the expected travel distance of the selected workers.

However, most of the existing studies have two major drawbacks. First, they assume that various workers have same location privacy levels, which makes some workers are under-protected and others are over-protected. Second, they only take into account single-location tasks (for example, collecting data at a location). But in the practical application of SC, many tasks published by submitters are multi-location tasks (for example, taking photos at location A and location B). As far as we know, there is no research on multi-location task allocation with location privacy protection.

In this paper, we propose the Multi-location Task Allocation Problem with personalized location privacy protection in SC (MLTAP). There are two challenges in the MLTAP problem.

1. How to estimate the proximity of worker to task given only the worker's confused location.
2. How to maximize utility on multi-location task allocation.

We propose an efficient Multi-location task Allocation framework with Personalized location Privacy-protecting (MAPP). It can effectively assign multi-location tasks to appropriate workers while protecting the location privacy of workers. The basic idea is to convert the real location of each worker into a circular area. The radius of the circle is determined by the privacy level, which is set by the worker. These circular areas are indexed by the R-tree. The SC server saves the location of the task with minimum bounding rectangle (MBR). The candidate workers for the task are those whose circular areas intersect with the MBR of the task. The SC server evaluates the proximity between the candidate worker and the task, and then pairs the worker and the task accordingly.

The main contributions of this paper are as follows:

- We propose an efficient Multi-location task Allocation framework with Personalized location Privacy-protecting, which not only allocates multi-location tasks efficiently, but also provides personalized protection of location privacy for workers.
- We propose a task pruning strategy to improve allocation performance. We prune the unreachable workers using the R-tree of workers and the MBR of each multi-location task to produce the candidate worker set for each task.

- We conduct extensive experiments on real-world and virtual datasets. The results show that MAPP can provide higher utility and lower average error.

The rest of this paper is organized as follows. In Section 2, we introduce the related works of SC. The problem modeling is presented in Section 3. In Section 4, we introduce the proposed solutions and algorithms in detail. In Section 5, we show the experimental settings and experimental results. We summarize the full paper in Section 6.

2. Related works

In this section, we mainly present the related works from two aspects: privacy protection and task allocation in SC.

2.1. Privacy protection

Privacy protection in SC can be divided into two categories: location privacy protection and data privacy protection.

2.1.1. Location privacy protection

In SC, workers upload their locations to the SC server. The attacker may infer the identity, address, and other privacy information from their locations, which poses a great threat to workers. Existing location privacy protecting methods mainly include cloaking, differential privacy, and encryption.

Cloaking. When the locations of workers are cloaked in an area, the attacker can not distinguish them in this area. Krontiris et al. propose a scheme to transform workers' point coordinates into a block in space using K-anonymity [8]. The locations uploaded by workers contain block ID and time interval ID, instead of absolute location and time. At least k workers are in the same block in a time interval so that it is difficult for adversary to distinguish k workers according to the locations. Dong et al. propose a solution to protect location privacy when providing fine-grained location services [9]. The workers send anonymous coarse-grained location information to the server and encrypt fine-grained location information to the task submitter. Chen et al. introduce Pseudonym Certification Authority(PCA) to generate different pseudonyms for the same worker [10]. The worker submits multiple data reports to the server using different pseudonyms, making it impossible to associate each worker.

Differential Privacy. Differential privacy makes it impossible for the attacker to find the original location information of workers by confusing their location information. To et al. use the trusted Cellular Service Provider (CSP) to receive the real locations of workers [11]. CSP applies differential privacy to hide the real locations of workers and sends them to the SC server. Qiu et al. take into account the features of vehicle workers' mobility in vehicle road networks [12]. They design a location obfuscation strategy to minimize the loss of quality-of-service (QoS) due to task distribution with location obfuscation, while guaranteeing geographic differential privacy to be satisfied.

Encryption. Encryption is often used for location privacy protection. The location information of workers and tasks is encrypted so that only participants with keys can know their real location information. Shen et al. propose an encryption-based task allocation method [13], which introduces the Privacy Service Provider (PSP) to collect encrypted data and encrypted location information from workers. The SC server assigns workers to tasks through communication with PSP.

2.1.2. Data privacy protection

In addition to location privacy information, the data collected by workers may also contain key privacy information. For example, street view photos may expose the travel information of other people, so the information collected by workers must be protected. The prior art is mainly homomorphic encryption, adding noise, cloaking or based on the system structure.

Homomorphic Encryption. Homomorphic encryption means that the data is processed and decrypted after homomorphic encryption, and the result is consistent with that obtained by direct processing. Gunther et al. propose an encryption scheme [14], which introduces a trusted third party to issue keys to workers. Then workers use the private key to encrypt data, avoiding the server or other attackers from obtaining data. Zhuo et al. use BGV homomorphic encryption for identity management and key management through a trust organization [15]. The scheme uses homomorphic encryption and homomorphic hash function to verify the correctness of data.

Adding Noise. Adding noise to the data can make the attacker unable to know the original data and protect the data security. Chen et al. propose a scheme to protect data privacy by adding random noise to each data message [16]. The noise is carefully selected to ensure privacy. Other schemes [17] add random disturbance to sensor data to protect data privacy. [18] uses a deep neural network to decide what kind of noise to inject. When numerous disturbed data are aggregated together, they can offset the deviation caused by disturbance.

Cloaking. Cloaking is a helpful method for data privacy protection, much as location privacy protection. Mahanan et al. propose a data privacy preservation heuristic algorithm using k-anonymity [19]. The algorithm is created based on the observations on the anonymous property of the problem structure, which can eliminate the privacy constraints consideration. Gisdakis et al. introduced a trusted third party for workers' identity and key management [20]. So the adversary can not associate the data and the worker, which well protects the data privacy of workers.

Based on System Structure. Based on different system structures, there can be different ways to protect data privacy. Wang et al. create a state decomposition scheme to safeguard privacy from eavesdroppers in distributed systems [21]. Each agent's original state is divided into two sub-states to avoid disclosing individual state information. One of which takes over the original state's function in interactions with other nodes. The other sub-state only interacts with the first one and is invisible to other nearby agents. Zhang et al. investigate dynamic average consensus, which relies on algebraic graph theory and Lyapunov-based control theory [22]. An et al. propose a new framework for opacity in cyber-physical systems to find distributed estimate algorithms that increase opacity while maintaining estimation accuracy [23].

2.2. Task allocation

Task allocation in SC can be divided into two categories: single task allocation and multiple task allocation.

2.2.1. Single task allocation

Each task is only assigned to one worker, who is selected from the candidate workers. Single task allocation can be divided into online allocation and offline allocation according to the time of the task allocation.

Online Allocation. In online allocation, the SC server assigns workers or tasks as soon as it receives their locations. Tong et al. design an online task allocation mechanism to deal with the micro-task allocation problem [24]. Pu et al. propose an online distribution system called CrowdLet [25], and task requesters can actively recruit appropriate workers to complete tasks.

Offline Allocation. Offline allocation means that the server knows all the information of workers and tasks, so it can design algorithms to achieve optimal allocation [26]. The task offline allocation problem can be solved by being reduced to the problem of maximum weighted bipartite matching [27].

2.2.2. Multiple task allocation

Multiple task allocation is mainly divided into homogeneous task allocation and isomeric task allocation according to different task types.

Homogeneous Task Allocation. Homogeneous task means that the requirements of each task on the platform are basically the same except for the location. Song et al. consider the requirements of different SC tasks on Quality of Information (QoI) [28]. They propose a multiple task allocation strategy by selecting a minimum subset of workers under the total budget constraint to ensure QoI. Yu et al. build a multitask screening decision tree to allocate multitasks. They calculate the candidate tasks' time difference, travel cost ratio, etc. to improve the resource utilization rate of the SC platform. [29].

Isomeric Task Allocation. Isomeric task means that each task has different requirements for worker's ability and deadline. Compared with homogeneous tasks, isomeric tasks also need to consider the differences between each task. Xiao et al. study the worker selection problem for deadline-sensitive tasks [30]. They recruit multiple workers for a task in order to meet the task deadline. Wang et al. transform the multitask allocation problem into a bipartite graph matching problem [31]. Considering the availability of sensors carried by participants, they propose an iterative greedy algorithm to solve it.

3. Attack model and problem formalization

In this section, we first introduce the attack model, then propose the Multi-location Task Allocation Problem with personalized location privacy protection in SC (MLTAP) and finally formalize it.

3.1. Attack model

In this paper, we adopted the semi-honest model proposed in [32] as the attack model. That is, SC servers, workers and task publishers are curious, but not malicious. They collect all the information they can, but do not collude with each other to gather more information. The assumption is realistic, because if workers and task publishers want to finish the task, there is little incentive for them to collude with others [33]. Besides, if the SC servers act maliciously, their reputation will be seriously damaged [34].

After a task is completed, the worker responsible for it sends results to the task publisher via the SC server, and the task publisher pays the salary. We mainly focus on the worker's location privacy. The data privacy of workers and task submitters (e.g., IP address, PayPal account, etc.) is not considered in this paper.

3.2. Problem definition

In SC, workers upload their location information to the SC server. When a task publisher submits a multi-location task to the SC server, the SC server immediately filters the appropriate workers for the task and assigns a near worker to the task. The selected worker goes to the multiple places to perform the task and sends results to the task publisher. Table 1 lists the commonly used notations in this paper.

In order to facilitate the description and formalization of the problem, we first introduce the following definitions.

Table 1
Summary of notations.

Notation	Description
w	worker
l_w	real location of the worker
$d_{will}(w)$	willing distance of the worker
l'_w	confused location of the worker
ϵ	privacy level
t	task
$l_s(t)$	sub-location of the task t
$d(w,t)$	distance between the worker w and the task t
$asm(w,t)$	assignment with the worker w and the task t
$c(l_w, d_{will}(w))$	willing circle of the worker
$c(l'_w, \epsilon)$	confusion circle of the worker

Definition 1 (*Worker*). Worker w has smart devices with sensors (e.g., mobile phones), and can move to the locations of the task to complete it.

To protect the location privacy, the worker w does not upload the real location l_w to the SC server. Instead, the worker w sets the privacy level ϵ and generates the confused location l'_w, where $d(l_w, l'_w) \leqslant \epsilon$ and $l_w \neq l'_w$. Then the worker sends l'_w and ϵ to the SC server. The circle $c(l'_w, \epsilon)$ with center l'_w and radius ϵ is called the confusion circle.

The worker w sets the willing distance $d_{will}(w)$ to decide how far he or she will accept the task. The willing circle $c(l_w, d_{will}(w))$ is a circle with the real location l_w as the center and the willing distance $d_{will}(w)$ as the radius. Once the task t is not within the willing circle, i.e., $d(w,t) > d_{will}(w)$, the worker w is not accessible to the task t, and will refuse it.

The set of workers is represented by W, where $W = \{w_1, w_2, \ldots\}$.

Definition 2 (*Task*). Each task t is a multi-location task, which requires a worker w to go to all locations of the task t and perform corresponding actions. Each location that needs to be visited in the above process is called a sub-location $l_s(t)$. Each task contains a set of sub-location $L_s(t) = \{l_{s1}(t), l_{s2}(t), \ldots\}$, and a worker w is assigned to traverse these sub-locations.

The set of tasks is denoted by T, where $T = \{t_1, t_2, \ldots\}$. For each task t, all the workers who are possible to accept it form a candidate worker set $W_c(t)$.

Definition 3 (*Distance between the Worker and the Task*). The distance between the worker w and the task t is defined as the distance between w and the nearest sub-location $l_s(t)$ to w, which is denoted by $d(w,t)$. That is,

$$d(w,t) = \min\left(d(w, l_s(t)) | l_s(t) \in L_s(t)\right) \tag{1}$$

Definition 4 (*Task Assignment*). In this paper, we focus on online allocation, that is, workers upload their real-time locations to the SC server. The task publishers submit tasks to the SC server constantly. After receiving the task information published by the task submitter, the SC server immediately assigns a worker to the task. So the worker-task pair forms an assignment $asm(w,t)$, where $d(w,t) \leqslant d_{will}(w)$. The set of all task assignments is represented by $\mathcal{A}(W,T)$, where $\mathcal{A}(W,T) = \{asm(w_1,t_1), asm(w_2,t_2), \ldots\}$.

We use minimum bounding rectangle (MBR) to surround the sub-locations $L_s(t)$ of t. If the worker w's willing circle $c(l_w, d_{will}(w))$ has an intersection with the MBR of the task t, it means that the worker is possible to reach the task and should be added to the candidate worker set $W_c(t)$ of the task t.

As shown in Fig. 2, the solid rectangle MBR_i represents the MBR of the task t_i, the dotted circle represents $c(l_w, d_{will}(w))$. Since MBR_1 intersects c_1, the worker w_1 can be a candidate of t_1, i.e., $W_c(t_1) = \{w_1\}$. Similarly, MBR_2 has intersection with c_1, c_2 and c_3. MBR_3 intersects c_2 and c_3. So, $W_c(t_2) = \{w_1, w_2, w_3\}, W_c(t_3) = \{w_2, w_3\}$. In the online scenario, tasks arrive in a certain order. Assuming that tasks arrive in the order of t_1, t_2 and t_3, w_1 can be assigned to t_1. w_2 and w_3 can be assigned to t_2 and t_3, respectively. That is, $\mathcal{A}(W,T) = \{asm(w_1,t_1), asm(w_2,t_2), asm(w_3,t_3)\}$.

Definition 5 (*Utility*). Utility $U(W,T)$ refers to the number of assignments after task allocation, i.e., $U(W,T) = |\mathcal{A}(W,T)|$. This is an important metric to measure the performance of the SC framework. Due to the confusion of the worker's location, the SC server may incorrectly select candidate workers for a task. Therefore, the challenge is to maximize the utility with location confusion. High utility indicates that more workers are assigned to tasks and more tasks can be performed.

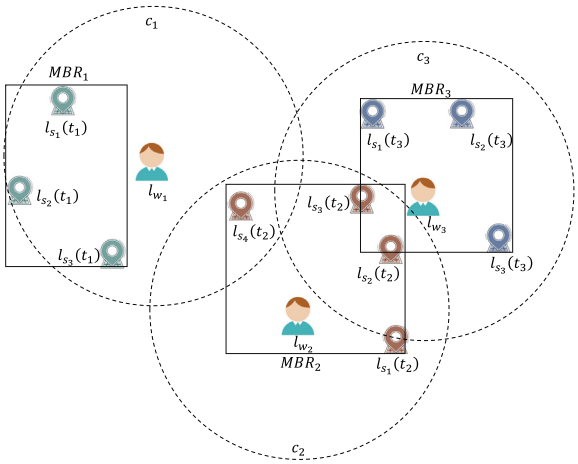

Fig. 2. Task allocation.

Definition 6 (*Average Error*). Owing to the confusion of the worker's location, the SC server may misjudge the accessibility of the worker w to the task t and tries to assign the task t to the unreachable worker w. The worker w will reject the task t, which forms an error assignment $asm_{err}(w,t)$. The average error $E(W,T)$ is the ratio of the number of error assignments to utility, that is,

$$E(W,T) = \frac{|\{asm_{err}(w,t)|w \in W, t \in T\}|}{U(W,T)} \tag{2}$$

where $|\{asm_{err}(w,t)|w \in W, t \in T\}|$ represents the number of error assignments. The average error reflects the trial-and-error cost required for each successful allocation. The challenge is to minimize the average error, which can reduce the additional communication and calculation costs caused by the reassignment.

Definition 7 (*Location Privacy*). To protect location privacy, the worker uploads confusion circle $c(l'_w, \epsilon)$ to the SC server. The real location of the worker l_w is hidden in the confusion circle. If the information of the confusion circle is illegally disclosed, the adversary can guess the real location of the worker. The guess about the location is denoted by l_g. If the distance between l_g and l_w is closer than the threshold r, which is a constant, we assume that the real location is leaked. The probability of leaking the real location of a worker $Pr(leak)$ can be approximated as $Pr(d(l_w, l_g) \leqslant r)$, which equals to $Pr(l_g \in c(l_w, r))$.

We define the location privacy LP as the probability that the real location of the worker is not leaked. Then LP can be calculated as follows.

$$LP = \begin{cases} 1 - \frac{\pi r^2}{\pi \epsilon^2}, & d(l_w, l'_w) + r \leqslant \epsilon, \\ 1 - \frac{\mu r^2 + \nu \epsilon^2 - r \times d(l_w, l'_w) \sin \mu}{\pi \epsilon^2}, & d(l_w, l'_w) + r > \epsilon \end{cases} \tag{3}$$

where $d(l_w, l'_w)$ is the distance between worker's real location l_w and confused location l'_w, $\mu = \arccos \frac{r^2 + d(l_w, l'_w)^2 - \epsilon^2}{2r \times d(l_w, l'_w)}$, $\nu = \arccos \frac{\epsilon^2 + d(l_w, l'_w)^2 - r^2}{2\epsilon \times d(l_w, l'_w)}$.

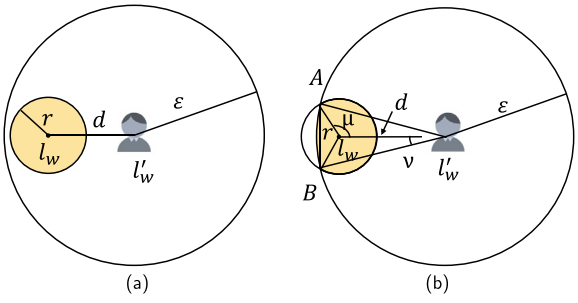

Fig. 3. Two kinds of position relationships between $c(l'_w, \epsilon)$ and $c(l_w, r)$. (a) $c(l'_w, \epsilon)$ contains $c(l_w, r)$. (b) $c(l'_w, \epsilon)$ intersects $c(l_w, r)$.

Proof. As Figs. 3,4, shows, since l_w is inside $c(l'_w, \epsilon)$, there are only two kinds of position relationships between $c(l'_w, \epsilon)$ and $c(l_w, r)$, that is, containing and intersecting. The intersection of $c(l'_w, \epsilon)$ and $c(l_w, r)$ is denoted by $Zone_i$.

When $c(l'_w, \epsilon)$ contains $c(l_w, r)$, $Zone_i$'s area A_{Zone_i} can be calculated by $A_{Zone_i} = \pi r^2$, where $d(l_w, l'_w) + r \leqslant \epsilon$.

When $c(l'_w, \epsilon)$ intersects $c(l_w, r)$, $A_{Zone_i} = A_{\triangledown Al_wB} + A_{\triangledown Al'_wB} - A_{\Delta l'_wAl_w} - A_{\Delta l'_wBl_w}$, where $d(l_w, l'_w) + r > \epsilon$. Let $\mu = \angle Al_wl'_w$, $v = \angle Al'_wl_w$. According to the cosine theorem, $\mu = \arccos\frac{r^2+d(l_w,l'_w)^2-\epsilon^2}{2r\times d(l_w,l'_w)}$, $v = \arccos\frac{\epsilon^2+d(l_w,l'_w)^2-r^2}{2\epsilon\times d(l_w,l'_w)}$. The area of the circular sectors $A_{\triangledown Al_wB}$ and $A_{\triangledown Al_wB}$ can be calculated by $A_{\triangledown Al_wB} = \pi r^2 2\mu$, and $A_{\triangledown Al'_wB} = \pi\epsilon^2 2v$, respectively. The formula for the area of the triangles $A_{\Delta l'_wAl_w}$ and $A_{\Delta l'_wBl_w}$ are $A_{\Delta l'_wAl_w} = A_{\Delta l'_wBl_w} = r \times d \times \sin\mu$. So, $A_{Zone_i} = \mu r^2 + v\epsilon^2 - r \times d(l_w, l'_w) \times \sin\mu$, where $d(l_w, l'_w) + r > \epsilon$.

We can derive that $LP = 1 - Pr(leak)$, based on the definition of LP.

When $d(l_w, l'_w) + r \leqslant \epsilon$, we randomly select a point in $c(l'_w, \epsilon)$, the probability density function pdf of the point in $c(l'_w, \epsilon)$ is $\frac{1}{A_{c(l'_w,\epsilon)}}$. $Pr(leak)$ is calculated as follows.

$$\begin{aligned} Pr(leak) &= \int\int_{Zone_i} pdf(x,y)\,dx\,dy \\ &= \int\int_{Zone_i} \frac{1}{A_{c(l'_w,\epsilon)}}\,dx\,dy \\ &= \frac{A_{Zone_i}}{A_{c(l'_w,\epsilon)}} \end{aligned} \tag{4}$$

Eq. (3) can be derived from the above equations.

Theorem 1. When the privacy level ϵ increases, LP also increases.

Proof. As Eq. (3) shows, when ϵ is deterministic, the minimum value of LP is $1 - \frac{\pi r^2}{\pi\epsilon^2}$. As the privacy level ϵ increases, LP also increases, which can better ensure the protection of location privacy.

3.3. Problem formalization

Given the worker set W and the ordered task set T, the objective is to select an appropriate set of assignments $\mathcal{A}(W, T)$ while maximizing the utility $U(W, T)$.

The allocation process is subject to the following constraints:

- Spatial Constraint: The assignment $asm(w, t)$ needs to ensure that the distance $d(w, t)$ from the worker w to the assigned task t does not exceed the willing distance of worker $d_{will}(w)$.
- Quantity Constraint: Each task t can only be assigned to one worker w, and each worker w can only accept one task t, that is, w and t can only appear once in $\mathcal{A}(W, T)$, respectively.

Then, the MLTAP problem can be formulated as follows:

$$maximize\ U(W,T) \tag{5}$$

Subject to:

$$d(w,t) \leqslant d_{will}(w)\ (w \in W, t \in T) \tag{6}$$

$$\forall w, |\{w|w \in \mathcal{A}(W,T)\}| \leqslant 1 \tag{7}$$

$$\forall t, |\{t|t \in \mathcal{A}(W,T)\}| \leqslant 1 \tag{8}$$

There are two major challenges to solve the MLTAP problem:

1. How to estimate the proximity of the worker to the task when given only the worker's confused location.
2. How to maximize utility on multi-location task allocation.

4. Proposed scheme

In this section, we mainly introduce MAPP. MAPP realizes multi-location task allocation efficiently and provides personalized location privacy protection for workers.

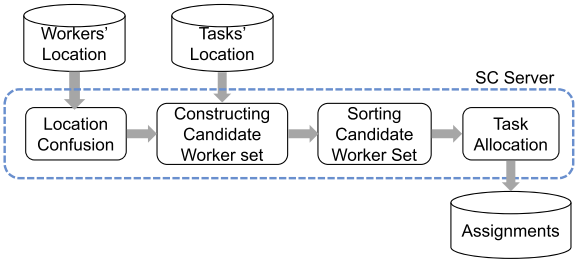

Fig. 4. Architecture of MAPP.

4.1. Architecture

The design principle of MAPP is as follows.

- Privacy: Each worker could choose the appropriate privacy level according to the worker's own needs. Our designed framework MAPP could effectively protect location privacy according to worker's privacy level.
- Performance: MAPP can effectively estimate the proximity of worker to task given only the worker's confused location and maximize the utility on multi-location task allocation.

To achieve the design principle, MAPP is mainly composed of four stages: location confusion, constructing candidate worker set, sorting candidate worker set and task allocation.

1. Location confusion. Each worker does not upload the real location directly, but submits a confusion location generated through the location confusion algorithm.
2. Constructing candidate worker set. We use the R-tree to store the MBR of all confusion circles of workers. Sub-locations in a task are surrounded by an MBR. We can get the candidate worker set by finding the R-tree nodes intersected with the MBR of the task.
3. Sorting candidate worker set. In order to increase utility, the SC server ranks each candidate worker according to the proximity of workers to tasks. However, due to the location confusion, it is hard to estimate the proximity, especially when the privacy level is various. We propose various algorithms using different ranking metrics, that is, distance, probability and expectation. The SC server sorts candidate workers by the rank.
4. Task allocation. The SC server sends the task to the candidate worker, who determines whether the task is reachable. If this is the case, the worker will accept it. Otherwise, the task is sent to the next candidate worker by the SC server.

4.2. Location confusion

In order to provide different levels of location privacy for various workers, MAPP allows workers to choose the appropriate level of location privacy according to their own needs. For example, workers who are more serious about their location privacy can raise their privacy level ϵ. Conversely, workers who pay less attention to their location privacy can downgrade their privacy level, which can better match their expectations for the protection of location privacy.

As shown in Fig. 5, we confuse the location of each worker, convert the location into a circle and then upload it to the SC server. As shown in Algorithm 1, the detailed steps of location confusion are as follows:

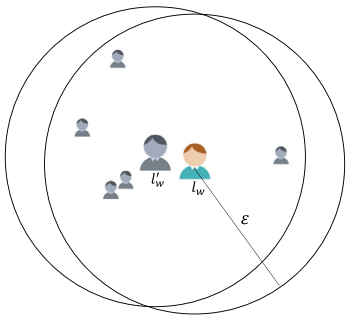

Fig. 5. Location confusion.

1. Each worker chooses a privacy level ϵ according to his or her will. The circle with the real location l_w of the worker as the center and ϵ as the radius is denoted by $c(l_w, \epsilon)$
2. Randomly select k points $(l_1, l_2, \ldots, l_k)$ in the circle $c(l_w, \epsilon)$, and calculate the average value of them as the confusion location l'_w according to Eq. (9). Taking l'_w as the center and ϵ as the radius, we can get the confusion circle $c(l'_w, \epsilon)$. It is obvious that the confusion circle $c(l'_w, \epsilon)$ must contain the real location l_w of the worker.

$$l'_w = \left(\frac{1}{k}\sum_{i=1}^{k} x_i, \frac{1}{k}\sum_{i=1}^{k} y_i\right) \tag{9}$$

3. The worker sends the confusion circle $c(l'_w, \epsilon)$ to the SC server, which ensures that the SC server and other SC participants can only see the confused location uploaded by the worker, so as to avoid the disclosure of real location.

Algorithm 1 Calculate Confused Location

Input: ϵ, k, x_w, y_w
Output: x'_w, y'_w
Initialize empty X, Y, $count = 0$
while $count < k$ **do**
  repeat
  Randomly generate $x \in (x - \epsilon, x + \epsilon)$
  Randomly generate $y \in (y - \epsilon, y + \epsilon)$
  until $\sqrt{(x - x_w)^2 + (y - y_w)^2} \leqslant \epsilon$
  Add x to X, add y to Y, $count++$
end while
$x'_w = \text{sum}(X)/k$, $y'_w = \text{sum}(Y)/k$
return x'_w, y'_w, ϵ

The location confusion algorithm protects the workers' location privacy but makes the SC server difficult to judge the accessibility between the worker w and the task t. As shown in Fig. 6, compared with Fig. 2, the locations of workers are confused, which are denoted by l'_{w_i}. The solid rectangle MBR_i represents the MBR of the task t_i, the dotted circle represents the willing circle $c(l'_w, d_{will}(w))$.

MBR_1 only intersects c_1. MBR_2 intersects c_1, c_2 and c_3. MBR_3 intersects c_3. So, w_1 is the candidate worker of t_1. w_1, w_2 and w_3 are the candidate workers of t_2. w_3 is the candidate worker of t_3. That is, $W_c(t_1) = \{w_1\}, W_c(t_2) = \{w_1, w_2, w_3\}$, and $W_c(t_3) = \{w_3\}$, which is different from the worker accessibility described in Section 3.2 (see Fig. 7).

4.3. Constructing candidate worker set

It is obvious that enumerating all the worker-task pairs and judging the accessibility of them is costly. In order to allocate tasks efficiently, the SC server needs to screen out the workers who are unable to reach the task.

In the constructing candidate worker set stage, we use the R-tree to index workers. R-tree is a tree data structure used for spatial access methods. The key idea of R-tree is using the MBR to group nearby nodes in the higher level of the tree. If a query does not intersect the MBR, it cannot intersect nodes inside the MBR. R-tree is a balance tree, so it has good performance for indexing multidimensional information [35].

The steps of constructing candidate worker set using the R-tree are as follows:

1. Build the R-tree for W.
   After the location confusion algorithm is applied, the SC server needs to eliminate the deviation of reachable workers caused by location confusion as much as possible. As shown in Fig. 8, the SC server build the MBR of $c(l'_w, \epsilon + d_{will}(w))$, where the radius is the sum of the privacy level ϵ and willing distance $d_{will}(w)$. A node of R-tree is denoted by R_i, which contains the MBR. Fig. 9 shows the constructed R-tree structure.
2. Build the MBR for t.
   When the task t arrives, the SC server surrounds the sub-locations $L_s(t)$ with the MBR which is shown in Fig. 8.
3. Put workers in leaf nodes intersecting with the MBR of t into $W_c(t)$.
   The SC server prunes those R-tree nodes which do not intersect with the MBR of the task t. In Fig. 8, since R_{15} and R_{11} and their child nodes do not intersect with the MBR of t, they can be pruned. The workers stored in the rest of the leaf nodes form the candidate worker set $W_c(t)$. In Fig. 8, R_1 and R_2 intersects the MBR of t, so, $W_c(t) = \{w_1, w_2\}$.

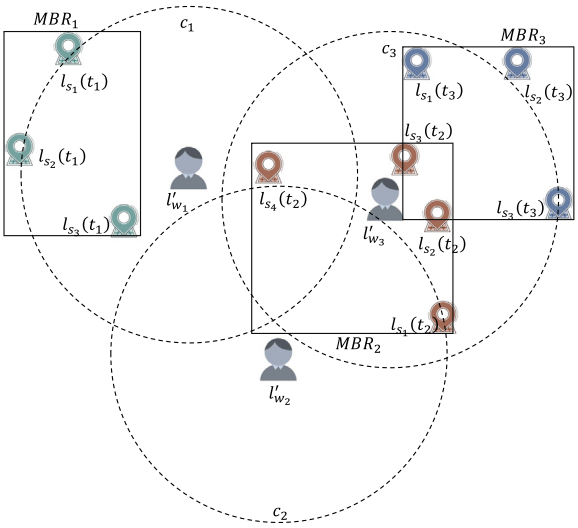

Fig. 6. Wrong accessibility after location confusion.

4.4. Sorting candidate worker set

For task allocation, we need to rank the reachable workers, sort them, and select the worker with the highest rank for the task in order to maximize utility [33].

The ranking metric is essential, since it determines which worker should be chosen and selected for the task. The worker who is closer to the task needs to have a higher rank. In this paper, we use the proximity from different aspects as various ranking metrics, that is, distance, probability and expectation.

Distance is often used to evaluate proximity. We first propose a distance-based algorithm as the baseline algorithm, namely, Distance-based Algorithm (DA). In DA, the SC server calculates the distance between the worker's confused location and the task to estimate the proximity. However, it is inaccurate to use the distance between the worker and the task to estimate the proximity given only the worker's confused location.

In order to solve the adverse effects of location confusion, we calculate the probability that the distance between the worker and the task is less than the willing distance, namely, accessibility probability, which is denoted by $Pr(d(w,t) \leqslant d_{will}(w))$. Probability-based metric considers the possible real location of workers. The workers who are more likely to be assigned to the task are prioritized when ranking based on accessibility probability. We propose two probability-based algorithms, that is, the Area Probability-based Algorithm (APA), and the Monte Carlo Probability-based Algorithm (MPA). APA uses integral to calculate the accessibility probability, but it is time-consuming. MPA uses the Monte Carlo algorithm to estimate the accessibility probability, hence reducing the overhead.

However, probability-based algorithms do not consider the distance between workers and tasks. Finally, we take both the distance and the accessibility probability into consideration and combine them as the ranking metric. To eliminate the error of location confusion, we use distance expectation to evaluate the distance between the worker and the task. It is hard to calculate expectation through integral, so we use the Monte Carlo algorithm instead and propose Monte Carlo Expectation-based Algorithm (MEA).

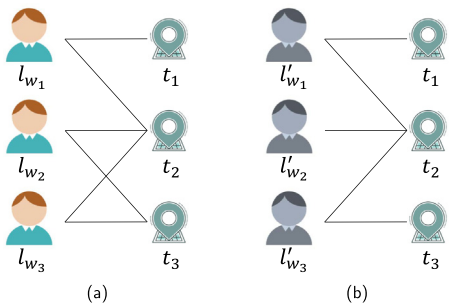

Fig. 7. Accessibility has changed after location confusion. (a) Accessibility before location confusion. (b) Accessibility after location confusion.

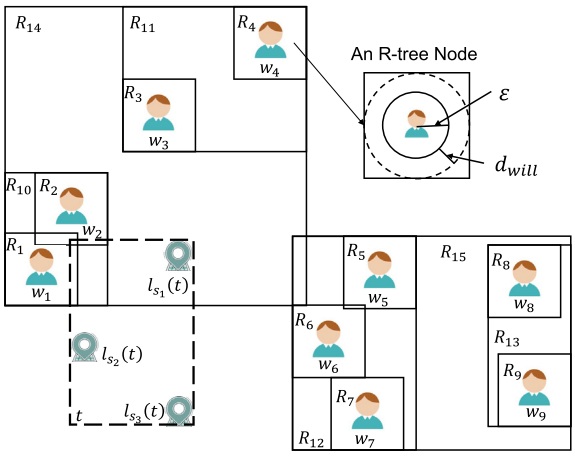

Fig. 8. Constructing candidate worker set using R-tree.

4.4.1. Distance-based Algorithm (DA)

Distance is a common metric for evaluating proximity. The idea of DA is that the SC server regards the confused location submitted by the worker as the real location, and calculates the distance between the workers in $W_c(t)$ and the task to estimate the real distance.

As Algorithm 2 shows, the SC server calculates $d(l'_w, t)$ for each worker w in $W_c(t)$. Then the server sort $W_c(t)$ by $d(l'_w, t)$ in ascending order.

Algorithm 2 Distance-based Algorithm

Input: $W_c(t)$, t
Output: sorted $W_c(t)$
1: **for** $w_i \in W_c(t)$ **do**
2: Calculate $d\left(l'_{w_i}, t\right)$
3: **end for**
4: Sort $W_c(t)$ in ascending order by $d\left(l'_{w_i}, t\right)$
5: **return** $W_c(t)$

4.4.2. Probability-based algorithm

Due to the location confusion, there is a deviation between the real location and the confused location. So DA can not reflect the true proximity.

Considering the process of the location confusion, the worker's real location is evenly distributed in the confused circle. Thus, we can calculate the accessibility probability $Pr(d(w,t) \leqslant d_{will}(w))$ to estimate the proximity.

When ranking according to the probability of accessibility, workers who are more likely to be allocated to the task will be given higher priority. Therefore, we use accessibility probability as ranking metric and propose probability-based algorithms to evaluate it. We first estimate the probability using integral calculation, but it is hard to calculate when integral region is the intersection of circles. Thus, we use Monte Carlo algorithm to calculate probability, which reduces the calculation time.

Area Probability-based Algorithm. In this paper, we propose the Area Probability-based Algorithm (APA) to calculate the accessibility probability between the worker and the task. The detailed process of calculating the accessibility probability is shown in Algorithm 3.

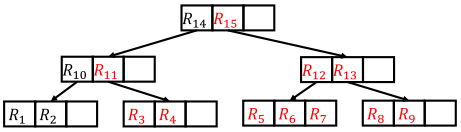

Fig. 9. The R-tree structure.

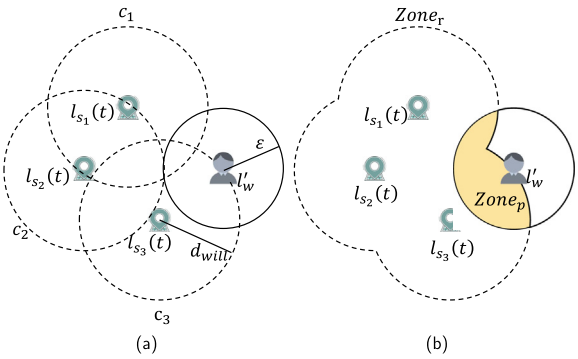

Fig. 10. Calculation of reachable zone. (a) Find the reachable circles of each task. (b) Merge all reachable circles.

Different from single-location tasks, there are multiple sub-locations in multi-location tasks. It is hard to calculate the accessibility probability, because not only the distance between the sub-locations and the worker but also the distribution of sub-locations impacts the accessibility probability.

To calculate it accurately, first, we introduce the reachable zone. In Fig. 10a, the reachable circle is the dotted circle $c(l_s(t), d_{will}(w))$, where the center is the sub-location $l_s(t)$ and the radius is the willing distance of worker $d_{will}(w)$.

As long as the real location of worker l_w is in the reachable circle, it indicates that the worker w can go to perform the task. These reachable circles are combined to get the reachable zone $Zone_r$, where $Zone_r = \bigcup_{i=1}^{n} c_i$. If the worker w wants to perform the task, the real location l_w should be in $Zone_r$. We can intersect $Zone_r$ and the worker's confusion location circle $c(l'_w, \epsilon)$ to get the shaded part $Zone_p$ in the Fig. 10b. That is, $Zone_p = Zone_r \cap c(l'_w, \epsilon)$.

Since the real location of the worker l_w appears with equal probability within the circle $c(l'_w, \epsilon)$, assume point $q = (x, y)$ is the possible real location of the worker, the *pdf* of q is as follows.

$$pdf(q) = \begin{cases} \frac{1}{A_{c(l'_w,\epsilon)}}, & q \in c(l'_w, \epsilon), \\ 0, & otherwise \end{cases} \tag{10}$$

Therefore, we use the following equation to calculate the accessibility probability of workers.

$$\begin{aligned} Pr(d(w,t) \leqslant d_{will}(w)) &= \int\int_{Zone_p} pdf(x,y)\, dx\, dy \\ &= \int\int_{Zone_p} \frac{1}{A_{c(l'_w,\epsilon)}}\, dx\, dy \\ &= \frac{A_{Zone_p}}{A_{c(l'_w,\epsilon)}} \end{aligned} \tag{11}$$

where $Pr(d(w,t) \leqslant d_{will}(w))$ is the probability of worker who is reachable, $pdf(x,y)$ is the *pdf* of $q = (x,y)$, A_{Zone_p} is the area of $Zone_p$, and $A_{c(l'_w,\epsilon)}$ is the area of $c(l'_w, \epsilon)$.

It is mathematically difficult to use conventional tools to calculate A_{Zone_p}, so we approximate the circle as a regular 64-gon. When merging and intersecting polygons, we calculate the vertexes of the new polygon. According to the shoelace theorem, we can calculate the area of the polygon using the following equation.

$$S = \frac{1}{2}\left|\sum_{i=1}^{n} x_i y_{i+1} - x_{i+1} y_i\right| \tag{12}$$

Algorithm 3 Area Probability-based Algorithm

Input: $W_c(t)$, $L_s(t)$, α
Output: $W_c(t)$
Initialize empty list C
for $w_i \in W_c(t)$ **do**
  for $l_s(t) \in L_s(t)$ **do**
    Add $c(l_s(t), d_{will}(w))$ to C
  end for
  $Zone_r$ =Merge(C)
  $Zone_p = Zone_r \cap c(l'_w, \epsilon)$
  $p_i = A_{Zone_p}/A_{c(l'_w,\epsilon)}$
  if $p_i < \alpha$ **then**
    Remove w_i from $W_c(t)$
  end if
end for
Sort $W_c(t)$ in descending order by p
return $W_c(t)$

For each candidate worker of the task, the SC server uses Eq. (11) and Eq. (12) to calculate the accessibility probability, and sorts $W_c(t)$ in descending order by probability. Workers with low probability are filtered out by setting the probability threshold α.

Monte Carlo Probability-based Algorithm. Because there is no efficient mathematical algorithm for merging and intersecting polygons, it is time-consuming to calculate A_{Zone_p} in Eq. (11). In order to solve this problem, we introduce the Monte Carlo Probability-based Algorithm (MPA) to evaluate the probability by using the Monte Carlo algorithm instead of calculating the area of the reachable zone. The Monte Carlo algorithm is a method to estimate the probability through a large number of simulation experiments. As shown in Algorithm 4, K sampling points are randomly selected in the worker's confusion circle $c(l'_w, \epsilon)$. K is the sampling number, and each sampling point is recorded as l. By calculating how many points meet $d(l,t) \leqslant d_{will}(w)$, we can calculate the accessibility probability of workers with the following equation:

$$Pr(d(w,t) \leqslant d_{will}(w)) = \frac{|\{l|d(l,t) \leqslant d_{will}(w)\}|}{K} \tag{13}$$

where $Pr(d(w,t) \leqslant d_{will}(w))$ is the probability that the task is reachable for the worker, and $|\{l|d(l,t) \leqslant d_{will}(w)\}|$ is the number of points where the distance from the sampling point to the task is less than the willing distance of worker.

Algorithm 4 Monte Carlo Probability-based Algorithm

Input: $W_c(t)$, t, K, α
Output: $W_c(t)$
Initialize empty list C, $count = 0$, $count_a = 0$
for $w_i \in W_c(t)$ **do**
  while $count < K$ **do**
    Randomly generate $l_i \in c(l'_w, \epsilon)$
    Calculate $d(l_i, t)$, $count++$
    if $d(l_i, t) < d_{will}(w)$ **then**
      $count_a++$
    end if
  end while
  $p_i = count_a/K$
  if $p_i < \alpha$ **then**
    Remove w_i from $W_c(t)$
  end if
end for
Sort $W_c(t)$ in descending order by p
return $W_c(t)$

4.4.3. Expectation-based algorithm

The above algorithms only consider the accessibility probability of the worker to the task. They ignore the distance between the worker and the task. In this section, we calculate the distance expectation between the worker and the task $Exp(d(w,t))$ and combine both distance expectation and accessibility probability as the ranking metric. We propose an expectation-based algorithm to take both probability and distance into consideration.

Similar to Eq. (11), we can use the following equation to calculate the distance expectation between the worker and the task.

$$Exp(d(w,t)) = \int \int_{c(l'_w,\epsilon)} d(x,y,t)\,dx\,dy \tag{14}$$

where the point q is the possible real location of the worker, $d(x,y,t)$ is the distance between the point $q=(x,y)$ and the task t.

However, owing to the multi-location task, it is hard to find the distance function between the point q and the task t. Different from MPA, we propose the Monte Carlo Expectation-based Algorithm (MEA) to estimate the distance expectation between workers and tasks.

In MEA, K sampling points l are randomly selected, and the sampling range is in $c(l'_w,\epsilon)$. As shown in Algorithm 5, the SC server calculates the distance $d(l,t)$ from each point to the task and uses Eq. (13) to calculate the accessibility probability. The workers with low accessibility probability can be screened out by setting the threshold. The distance expectation $Exp(d(w,t))$ from the worker to the task can be calculated as follows.

$$Exp(d(w,t)) = \frac{\sum_{i=1}^{k} d(l_i,t)}{K} \tag{15}$$

where $\sum_{i=1}^{k} d(l_i,t)$ represents the sum of the distances from each sampling point to the task. We combine expectation and probability as ranking metric:

$$rank = Pr(d(w,t)) + q/Exp(d(w,t)) \tag{16}$$

where q controls influence of the expectation on *rank*.

rank is positively correlated with $Pr(d(w,t))$ and negatively correlated with $Exp(d(w,t))$.

We use ranking threshold β to filter unreachable workers effectively. Finally, the SC server sorts the candidate worker set according to *rank*.

Algorithm 5 Monte Carlo Expectation-based Algorithm

Input: $W_c(t)$, t, K, β
Output: $W_c(t)$
1: Initialize empty list C, $D=0$, $count=0$, $count_a=0$
2: **for** $w_i \in W_c(t)$ **do**
3: **while** $count < K$ **do**
4: Randomly generate $l_i \in c(l'_w,\epsilon)$
5: Calculate $d(l_i,t)$, $D+=d(l_i,t)$, $count++$
6: **if** $d(l_i,t) < d_{will}(w)$ **then**
7: $count_a++$
8: **end if**
9: **end while**
10: $p_i = count_a/K$, $Exp_i = D/K$, $rank_i = p_i + 50/Exp_i$
11: **if** $rank_i < \beta$ **then**
12: Remove w_i from $W_c(t)$
13: **end if**
14: **end for**
15: Sort $W_c(t)$ in descending order by *rank*
16: **return** $W_c(t)$

4.5. Task allocation

After sorting candidate worker set by the above rank metrics, the candidate worker set $W_c(t)$ is in order. In the task allocation stage, the SC server pops out the first worker w in $W_c(t)$ and sends the task information of t to the worker w. After receiving the information, the worker w judges whether the distance to the task is within the willing distance, that is,

$d(w,t) \leqslant d_{will}(w)$. If so, the worker w accepts the task t. Otherwise, the worker rejects the task, and the SC server continues to send the task information to the next candidate worker and repeats the above process until there is no candidate worker in W_c.

Algorithm 6 Task Allocation Algorithm

Input: $W_c(t)$, t
Output: $asm(w,t)$
1: **while** $|W_c(t)| > 0$ **do**
2: Pop w
3: **if** w accepts t **then**
4: Generate $asm(w,t)$
5: **return** $asm(w,t)$
6: **end if**
7: **end while**
8: **return** NULL

Theorem 2. The time complexity of MAPP is $O(m\log(n)) + O(mN) + O(mNlog(N)) + O(mN)$, where m and n are the number of tasks and workers, respectively, and N is the average number of candidate workers for each task.

Proof. In Constructing Candidate Worker Set stage, the SC server uses R-tree to index workers. The time complexity for searching in the R-tree for all the tasks is $O(m\log(n))$. In Sorting Candidate Worker Set stage, the SC server needs to calculate ranking metrics for all the candidate workers of each task. So, the time complexity is $O(mN)$. Besides, the SC server needs to sort the candidate workers, the time complexity of quicksort is $O(mNlog(N))$. In Task Allocation stage, the worst case is that each candidate worker-task pair needs to be confirmed. The time complexity is $O(mN)$. Hence, the time complexity of MAPP is $O(m\log(n)) + O(mN) + O(mNlog(N)) + O(mN)$.

5. Experiment

In this section, we evaluate the performance of MAPP in terms of utility, average error and overhead. We carry out an extensive evaluation of our method in Section 5.1, and report experimental results with our system prototype in Section 5.2.

5.1. Experimental setup

5.1.1. Datasets

We choose a section of Beijing as the experimental area. The latitude range of the area is about 39.8265 to 39.9824 and the longitude range is about 116.2732 to 116.4923. We choose three different types of datasets to evaluate the performance of MAPP.

Real World Dataset (RWD). We use the GeoLife dataset [36] as our first dataset. The dataset collects GPS track information of 182 users for more than 5 years, a total of 17621 track information. Each track information includes latitude, longitude, altitude, and so on. In addition to daily life of users such as going home, these tracks also include some entertainment and sports activities, for example, sightseeing, shopping, cycling, etc. Fig. 11 shows the distribution of the dataset.

We keep trajectories that are completely in the experimental area. Then we randomly select start points from part of them to generate locations of workers, and randomly select some points as the locations of tasks. Due to population aggregation, user activities in the upper left corner of Fig. 11 are more concentrated than other areas.

Artificial Uniform Dataset (AUD). In this dataset, we uniformly generate data points in the experimental area as locations of workers or sub-locations of tasks.

Artificial Non-uniform Dataset (AND). There are no workers or tasks in many areas of the city, such as mountains, lakes, military control zones, etc.

In AND, we randomly select three circles as obstacles, with (39.9332°N, 116.3483°E), (39.8926°N, 116.3490°E), (39.8831°N, 116.4606°E) as the center respectively and 2000 meters as the radius. The location points in the other parts are still generated uniformly.

Fig. 12 shows that the distribution of three datasets are different. The real-world dataset is the most uneven. Workers and tasks are concentrated in the upper left area in Fig. 12a, which is also close to real-life (task points and workers are concentrated in business districts and residential areas). The artificial datasets are shown in Fig. 12b and Fig. 12c. Because there are three circular areas without workers and tasks in Fig. 12c, AND is more uneven compared with AUD in Fig. 12b.

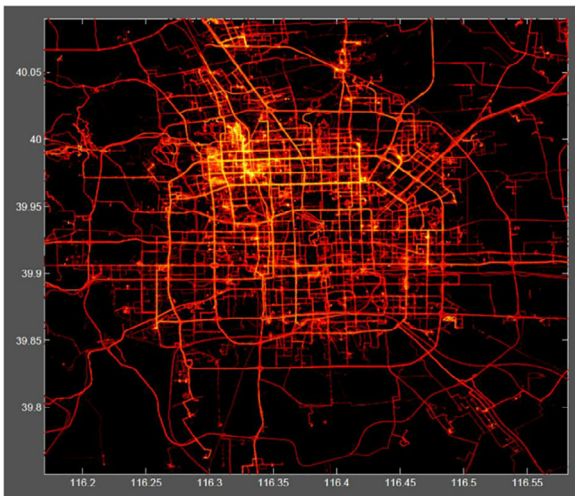

Fig. 11. GeoLife dataset thermodynamic diagram.

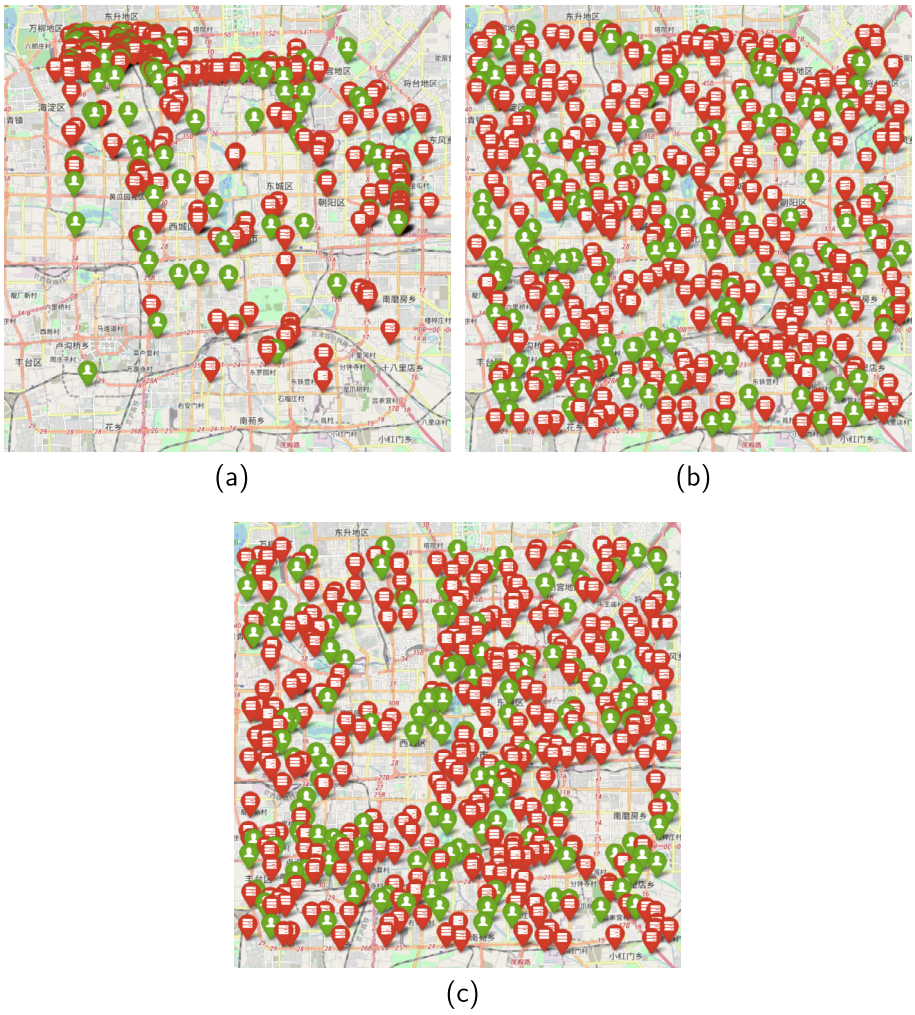

Fig. 12. Comparison of three kinds of datasets. The number of both workers and tasks are set to 150, and each task contains two sub-locations. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

Table 2
Parameter Setting.

Parameter	Range	Default Value
$\|W\|/\|T\|$	$\{0.5, 0.75, \mathbf{1.0}, 1.25, 1.5\}$	1.0
$\|T\|$		800
$\|L_s(t)\|$	$\{2, 3, \mathbf{4}, 5, 6\}$	4
ϵ	$\{2000, 2400, \mathbf{2800}, 3200, 3600\}$	2800
$d_{will}(w)$	$\{800, 900, \mathbf{1000}, 1100, 1200\}$	1000
α	$\{0.00, \mathbf{0.05}, 0.10, 0.15, 0.20\}$	0.05
K	$\{5, 10, \mathbf{15}, 20, 25\}$	15

5.1.2. Parameters

As Table 2 shows, we vary the ratio of workers and tasks $|W|/|T| \in \{0.5, 0.75, \mathbf{1.0}, 1.25, 1.5\}$, the number of sub-locations $|L_s(t)| \in \{2, 3, \mathbf{4}, 5, 6\}$ for each task. We set the number of tasks $|T|$ to 800, the privacy level $\epsilon \in \{2000, 2400, \mathbf{2800}, 3200, 3600\}$, the willing distance $d_{will}(w) \in \{800, 900, \mathbf{1000}, 1100, 1200\}$, the threshold of probability $\alpha \in \{0.00, \mathbf{0.05}, 0.10, 0.15, 0.2\}$, and the sampling number $K \in \{5, 10, \mathbf{15}, 20, 25\}$, where the bold values are the experimental default values. After our pre-test, we set q as 50 and the rank threshold $\beta = \alpha + 50/(1.15 \times d_{will}(w))$ which can effectively filter unreachable workers.

5.1.3. Performance metrics

Utility. The utility is measured by the number of task assignments. Due to the confusion of location, it may lead to the decline of assignments. The goal is to maximize the utility.

Average Error. The average error is measured by the workers' rejections per assignment. It indicates the calculation accuracy of accessibility between workers and tasks.

Overhead. The overhead is measured by the task allocation time, which shows the system cost of the SC server.

5.2. Experimental results

We compare the performance of various algorithms in our experiments. Due to location confusion, there is a deviation between the real location and the confused location of workers. We compare the Distance-based Algorithms that uses confused location (Distance-based Algorithm, DA) and real location (Distance-based Algorithmusing real location, DA-R). Besides, the Area Probability-based Algorithm (APA), the Monte Carlo Probability-based Algorithm (MPA) and the Monte Carlo Expectation Algorithm (MEA) are also compared in our experiment.

As far as we know, there is no research on the MLTAP problem. We choose DA as the baseline algorithm. Since DA-R uses the real location of workers, it is easy to calculate the accurate distance between the worker w and the task t. DA-R is the best algorithm in theory, so we choose DA-R as the optimal algorithm to evaluate performance of our algorithms.

5.2.1. Impact of the ratio of workers and tasks

First, we explore the impact of the ratio of workers and tasks $|W|/|T|$ on various performance metrics. It can be seen from Fig. 13 that the utility increases with $|W|/|T|$. However, after $|W|/|T|$ reaches 1, the utility does not increase significantly. We find that the utility of APA, MPA and MEA are close to the utility of DA-R and much higher than the utility of DA.

The Fig. 14 shows that the average error of DA-R is 0. Because the SC server knows the true location of all workers, the accessibility between workers and tasks can be calculated accurately. With the increase of $|W|/|T|$, the average error fluctuates in a certain range, which shows that $|W|/|T|$ has little effect on the average error. The average error of DA-R, APA, MPA and MEA are close but far better than DA.

As shown in Fig. 15, as $|W|/|T|$ increases, so does the overhead. Because there is no efficient mathematical algorithm for APA, it is time-consuming to calculate the proximity between workers and tasks. The overhead of APA is much higher than other algorithms.

5.2.2. Impact of the number of sub-locations

As shown in Fig. 16, with the increase of $|L_s(t)|$, the utility of DA-R, APA, MPA and MEA are almost unchanged. The utility of MEA is slightly lower than other algorithms, but much higher than the utility of DA. It is noted that in DA, the utility increases with the increase of $|L_s(t)|$. The reason is that when $|L_s(t)|$ increase, the reachable zone also increases. More workers will enter the candidate worker set, which improves the possibility of successful allocation.

According to Fig. 17, the average error decreases with the increase of $|L_s(t)|$. This is because when $|L_s(t)|$ increases, the reachable zone expands, which increases the fault tolerance of each algorithm and reduces the average error. When $|L_s(t)|$ increases, the gap between DA and other algorithms decreases. When the participants in the dataset are more concentrated, the average error of DA decreases faster.

Fig. 18 shows that with the increase of $|L_s(t)|$, the overhead of all algorithms are increasing.

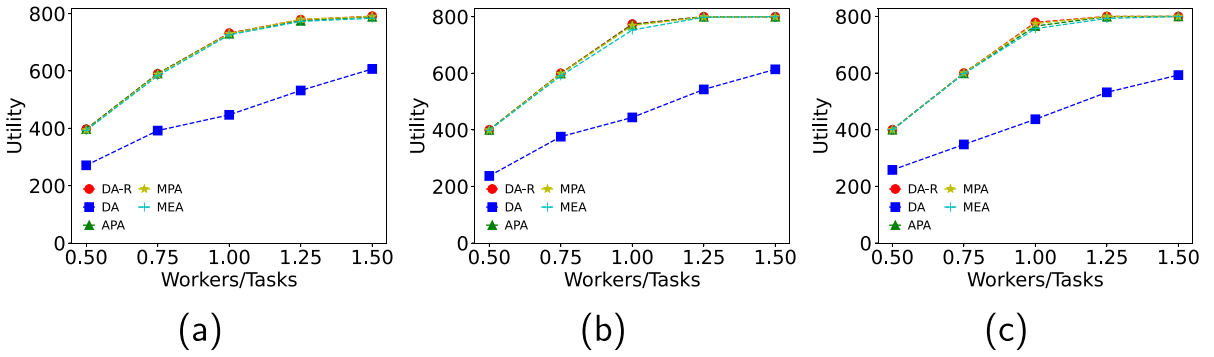

Fig. 13. Impact of the Ratio of Workers and Tasks on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

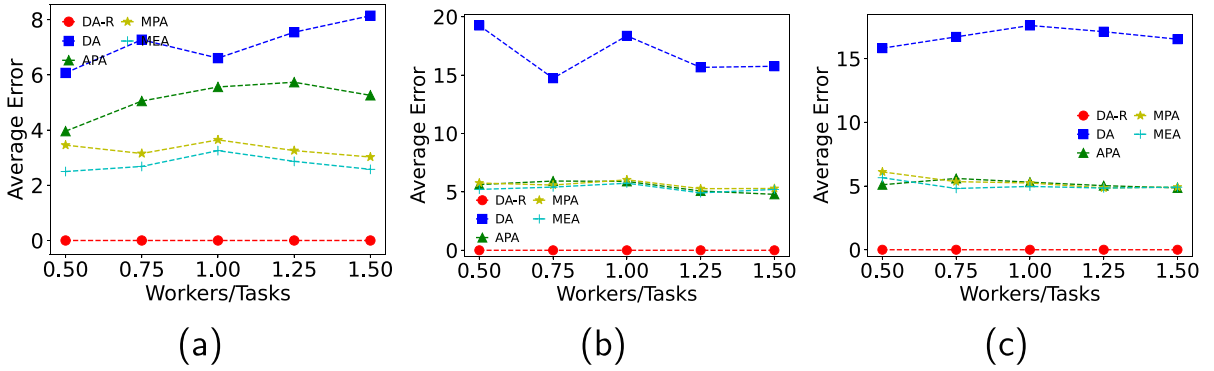

Fig. 14. Impact of the Ratio of Workers and Tasks on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

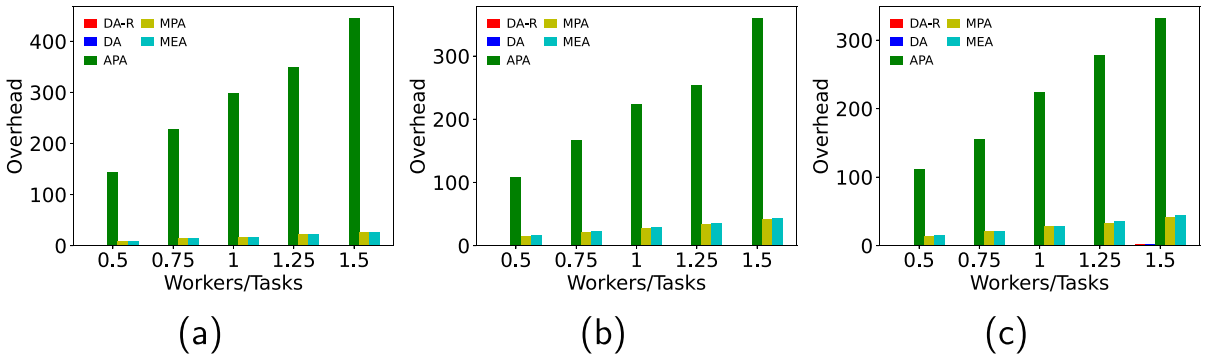

Fig. 15. Impact of the Number of Workers on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

5.2.3. Impact of the privacy level

Since the worker submits the confused location instead of the real location to SC server, the distance between workers and tasks is hard to calculate.

According to Fig. 19, with the increase of ϵ, the utility of DA-R, APA, MPA and MEA remain stable. When the location of SC participants is concentrated, the effect of MEA is slightly worse than that of other algorithms. However, the utility of DA decreases with the increase of privacy level. This is because when ϵ is greater, it is more difficult to judge accessibility, resulting in the decrease of the utility.

In Fig. 20, with the increase of ϵ, the average error of all algorithms except DA increases slightly. The average error of DA is the highest and increases fast. It shows that our algorithms can provide stable services when ϵ increases.

As can be seen from Fig. 21, ϵ has little effect on the overhead of algorithms except APA.

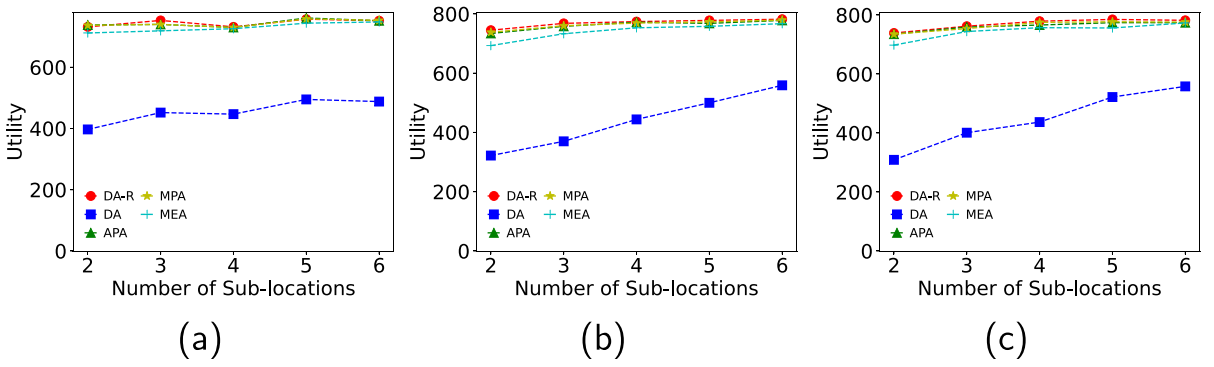

Fig. 16. Impact of the Number of Sub-locations on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

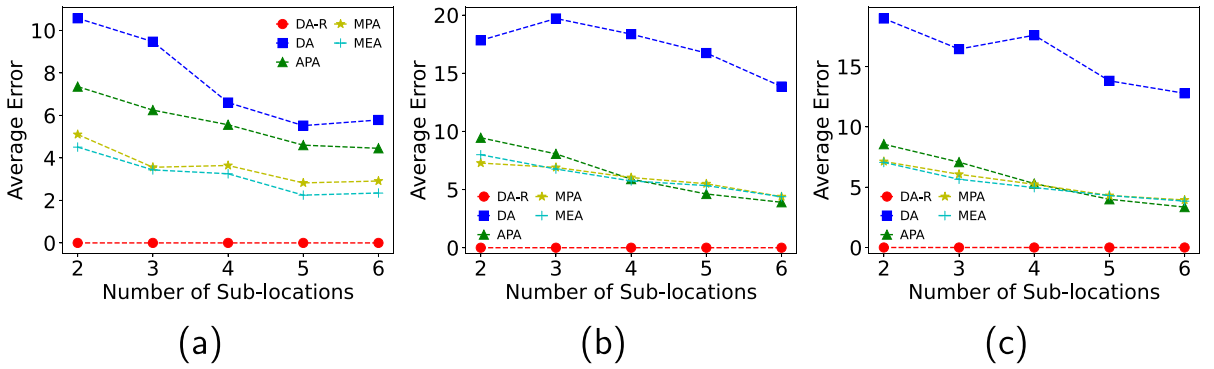

Fig. 17. Impact of the Number of Sub-locations on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

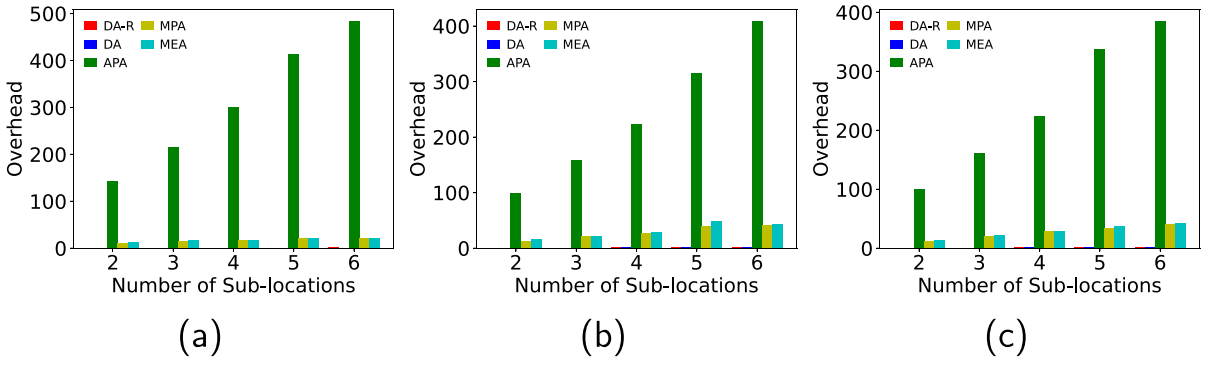

Fig. 18. Impact of the Number of Sub-locations on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

The experiments above show that there is a trade-off between the privacy level and average error. The higher the privacy level, the higher the worker's privacy can be protected. However, it will affect the calculation accuracy of accessibility between workers and tasks, which leads to more task allocation errors.

5.2.4. Impact of the willing distance

Fig. 22 shows that the utility of DA-R, APA, MPA and MEA increases slightly when $d_{will}(w)$ is increased. This is mainly because the candidate worker set for each task is expanded. As $d_{will}(w)$ increases, the reachable zone of each task also expands, which increases the fault tolerance and utility of DA, but there is still a large gap with other algorithms.

Fig. 23 shows that with the increase of $d_{will}(w)$, the average error of each algorithm decreases, and DA decreases the fastest. When $d_{will}(w)$ reaches 1400 meters, the average error are close with each other.

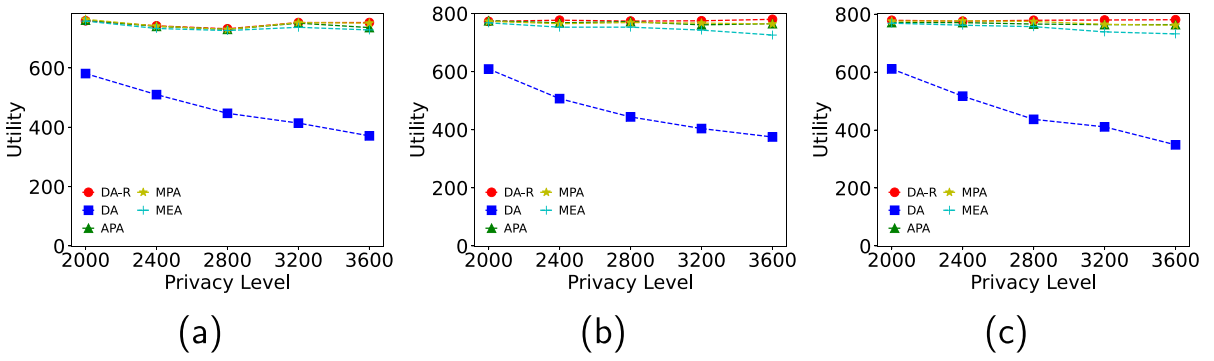

Fig. 19. Impact of the Privacy Level on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

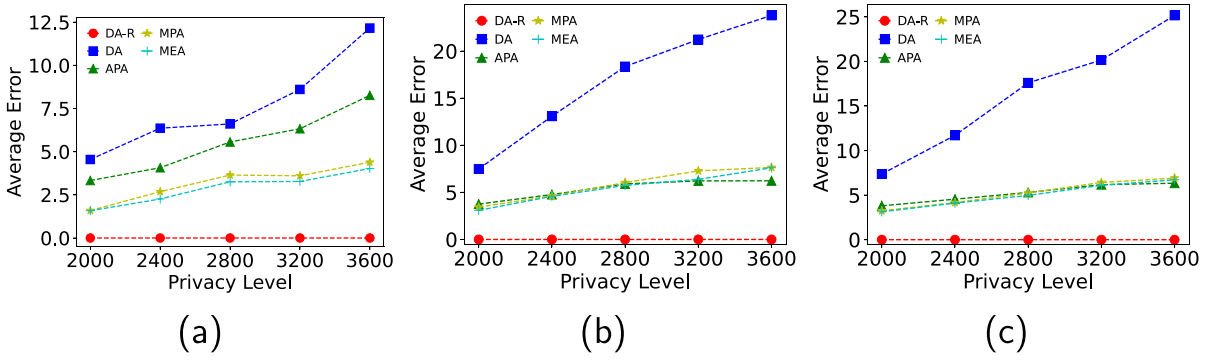

Fig. 20. Impact of the Privacy Level on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

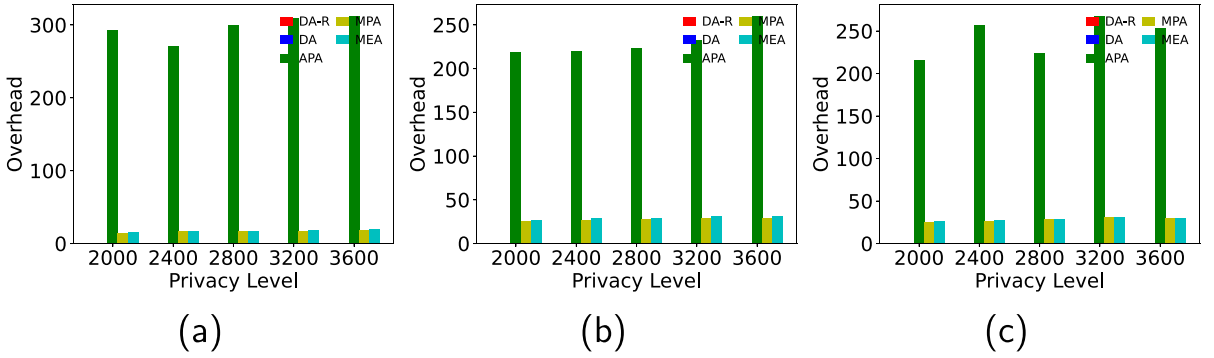

Fig. 21. Impact of the Privacy Level on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

Fig. 24 shows that when increasing $d_{will}(w)$, the overhead of APA is still in high level.

5.2.5. Impact of the probability threshold

As shown in Fig. 25, when α increases, the utility of APA, MPA and MEA decrease significantly. It indicates that when α is less than 0.1, there are enough reachable workers in the candidate worker set. But when α is greater than 0.1, reachable workers are not enough for task allocation.

Fig. 26 shows that when α increases, the average error decreases slightly, which shows that α affects the average error. This is mainly because some low probability candidate workers are screened out and reduce the possibility of errors. But at the same time, some reachable workers are also be screened out. Therefore, it is necessary to find the compromise α, which can not excessively lose utility, and can not make the average error at a high level.

As can be seen from Fig. 27, when α increases, the overhead fluctuates.

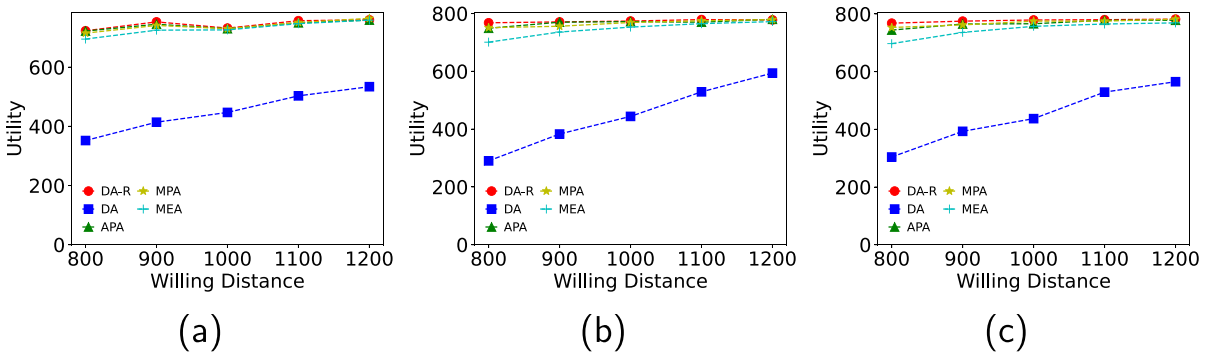

Fig. 22. Impact of the Willing Distance on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

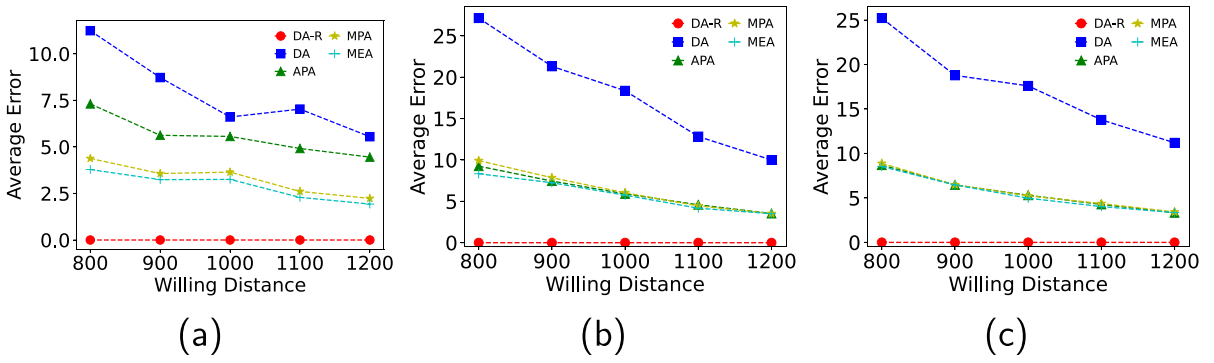

Fig. 23. Impact of the Willing Distance on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

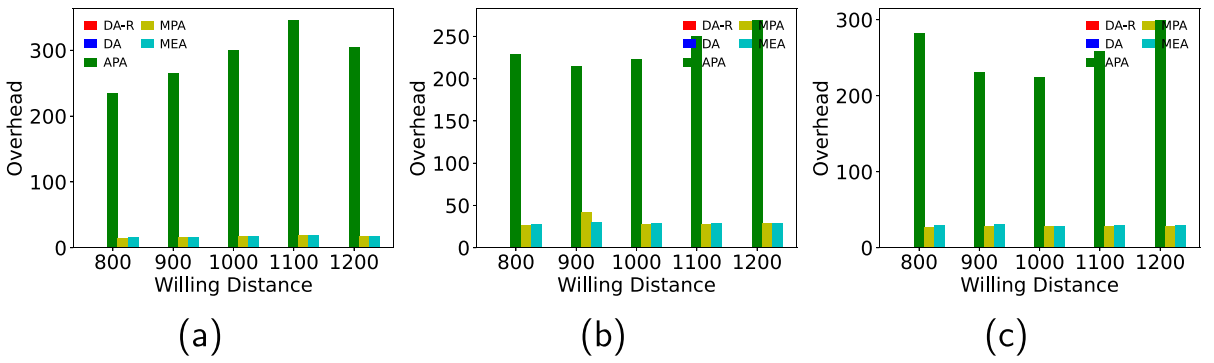

Fig. 24. Impact of the Willing Distance on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

5.2.6. Impact of the sampling number

As can be seen from Fig. 28, when K is less than 10, the utility increases with K, and after K reaches 10, the utility remains stable.

Fig. 29 shows that when K increases, the average error of MPA and MEA fluctuate, but remain basically stable.

It can be seen from Fig. 30 that when K increases, the overhead increases.

5.2.7. Summary

According to the above experiments, we found that when workers and tasks are located evenly, APA, MPA and MEA perform relatively similar in terms of utility and close to optimal. When workers and tasks are concentrated, MEA is inferior to the other two algorithms in utility, but far higher than baseline algorithm. MEA usually outperforms in terms of average error, and MPA and APA perform much better than baseline algorithm. The overhead of APA is higher than other algorithms due to the calculation of area.

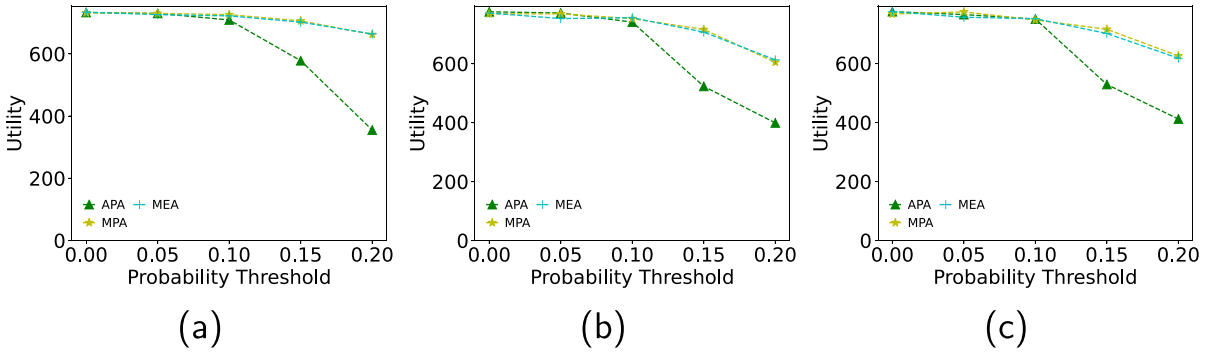

Fig. 25. Impact of the Probability Threshold on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

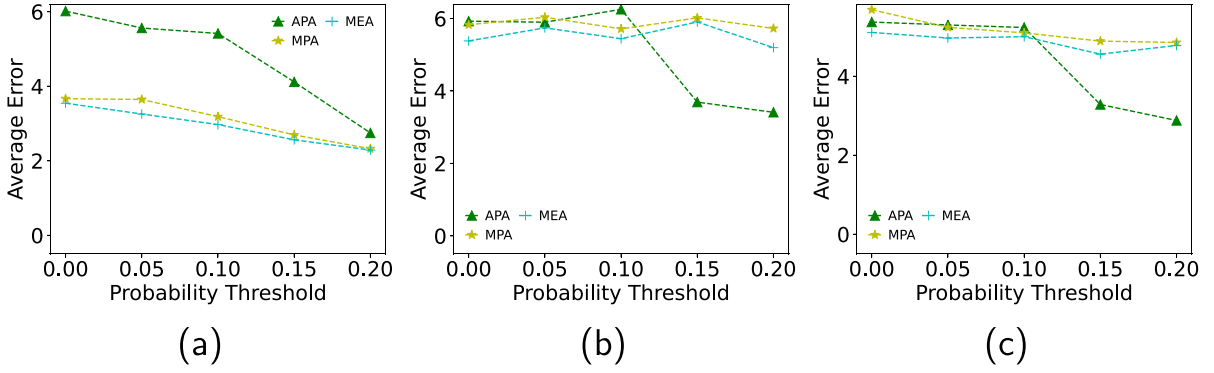

Fig. 26. Impact of the Probability Threshold on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

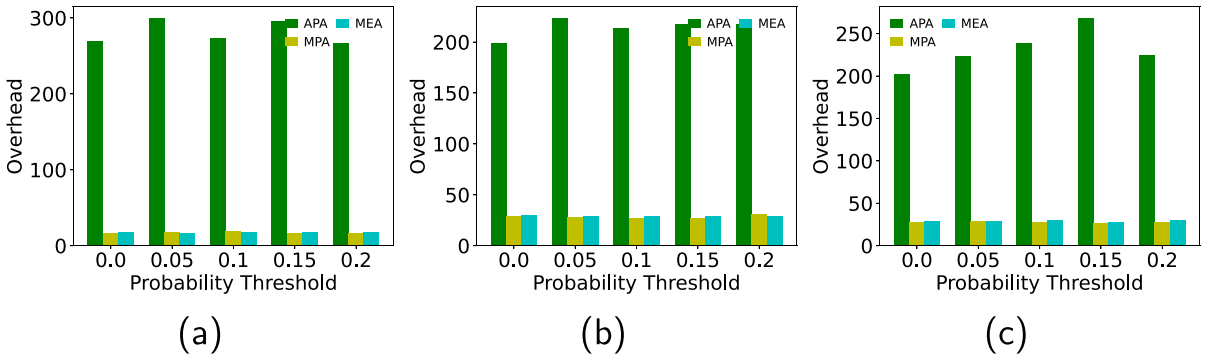

Fig. 27. Impact of the Probability Threshold on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

By comparing with the effect of the optimal and baseline algorithms, we found that our algorithms perform well in various performance metric and are effective and efficient.

6. Conclusion

Aiming at the MLTAP problem, we propose the Multi-location task Allocation framework with Personalized location Privacy-protecting (MAPP) in this paper. MAPP can assign appropriate workers to multi-location tasks and protect location privacy of workers from being leaked by untrusted servers.

In order to protect workers' location privacy, MAPP randomly confuses the real location of workers and converts it into a circle. To allocate task efficiently, MAPP uses the MBR and the R-tree to determine the candidate worker set of the task. To eliminate the location deviation caused by location confusion, we combine accessibility probability and distance expectation

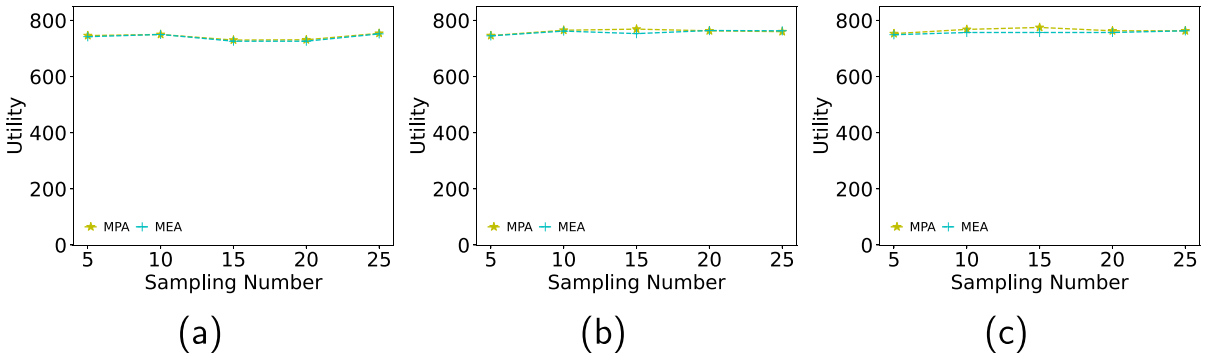

Fig. 28. Impact of the Sampling Number on the Utility. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

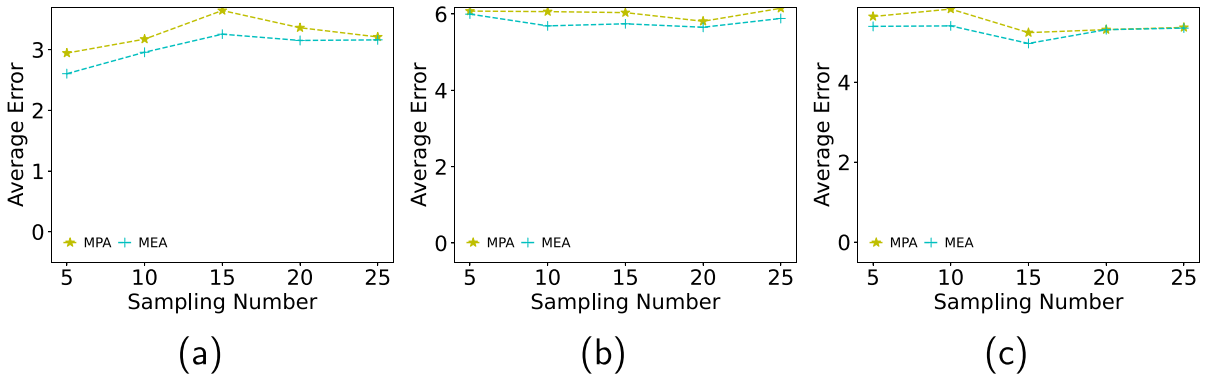

Fig. 29. Impact of the Sampling Number on the Average Error. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

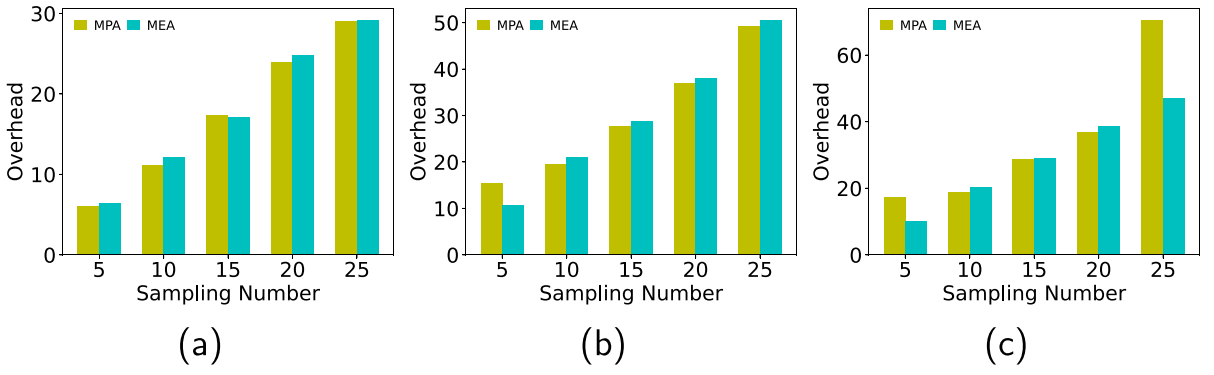

Fig. 30. Impact of the Sampling Number on the Overhead. (a) Real World Dataset. (b) Artificial Uniform Dataset. (c) Artificial Non-uniform Dataset.

as ranking metric to evaluate the proximity of each candidate worker to the task and sort workers according to the rank. The experiments prove that the proposed algorithms are effective and efficient.

CRediT authorship contribution statement

Yu Fan: Methodology, Software, Writing - original draft. **Liang Liu:** Project administration, Methodology, Writing - review & editing. **Xingxing Zhang:** Data curation, Formal analysis. **Huibin Shi:** Investigation, Writing - review & editing. **Wenbin Zhai:** Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the National Key R&D Program of China under No. 2021YFB2700500 and 2021YFB2700502, the Open Fund of Key Laboratory of Civil Aviation Smart Airport Theory and System, Civil Aviation University of China under No. SATS202206, the National Natural Science Foundation of China under No. U20B2050, Public Service Platform for Basic Software and Hardware Supply Chain Guarantee under No. TC210804A.

References

[1] L. Kazemi, C. Shahabi, Geocrowd: enabling query answering with spatial crowdsourcing, in: Proceedings of the 20th international conference on advances in geographic information systems, 2012, pp. 189–198.
[2] Y. Wang, Z. Yan, W. Feng, S. Liu, Privacy protection in mobile crowd sensing: a survey, World Wide Web 23 (1) (2020) 421–452.
[3] D. Gao, H. Lin, Z. Li, F. Qian, Q.A. Chen, Z. Qian, W. Liu, L. Gong, Y. Liu, A nationwide census on wifi security threats: prevalence, riskiness, and the economics, MobiCom (2021) 242–255.
[4] Y. Tong, J. She, B. Ding, L. Wang, L. Chen, Online mobile micro-task allocation in spatial crowdsourcing, 2016 IEEE 32Nd international conference on data engineering (ICDE), IEEE 2016 (2016) 49–60.
[5] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, Y.T. Hou, A survey on security, privacy, and trust in mobile crowdsourcing, IEEE Internet Things J. 5 (4) (2017) 2971–2992.
[6] C. Qiu, A. Squicciarini, Z. Li, C. Pang, L. Yan, Time-efficient geo-obfuscation to protect worker location privacy over road networks in spatial crowdsourcing, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1275–1284.
[7] L. Wang, D. Yang, X. Han, D. Zhang, X. Ma, Mobile crowdsourcing task allocation with differential-and-distortion geo-obfuscation, IEEE Trans. Dependable Secure Comput. 18 (2) (2019) 967–981.
[8] I. Krontiris, F.C. Freiling, T. Dimitriou, Location privacy in urban sensing networks: research challenges and directions [security and privacy in emerging wireless networks], IEEE Wirel. Commun. 17 (5) (2010) 30–35.
[9] K. Dong, T. Gu, X. Tao, J. Lu, Privacy protection in participatory sensing applications requiring fine-grained locations, in: 2010 IEEE 16th International Conference on Parallel and Distributed Systems, IEEE, 2010, pp. 9–16.
[10] J. Chen, H. Ma, D.S. Wei, D. Zhao, Participant-density-aware privacy-preserving aggregate statistics for mobile crowd-sensing, in: 2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS), IEEE, 2015, pp. 140–147.
[11] H. To, G. Ghinita, C. Shahabi, A framework for protecting worker location privacy in spatial crowdsourcing, Proceedings of the VLDB Endowment 7 (10) (2014) 919–930.
[12] C. Qiu, A.C. Squicciarini, C. Pang, N. Wang, B. Wu, Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability, IEEE Trans. Mob. Comput. (2020).
[13] Y. Shen, L. Huang, L. Li, X. Lu, S. Wang, W. Yang, Towards preserving worker location privacy in spatial crowdsourcing, 2015 IEEE global communications conference (GLOBECOM), IEEE 2015 (2015) 1–6.
[14] F. Günther, M. Manulis, A. Peter, Privacy-enhanced participatory sensing with collusion resistance and data aggregation, in: International Conference on Cryptology and Network Security, Springer, 2014, pp. 321–336.
[15] G. Zhuo, Q. Jia, L. Guo, M. Li, P. Li, Privacy-preserving verifiable data aggregation and analysis for cloud-assisted mobile crowdsourcing, in: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.
[16] J. Chen, H. Ma, D. Zhao, Private data aggregation with integrity assurance and fault tolerance for mobile crowd-sensing, Wireless Netw. 23 (1) (2017) 131–144.
[17] H. Jin, L. Su, H. Xiao, K. Nahrstedt, Inception: Incentivizing privacy-preserving data aggregation for mobile crowd sensing systems, in: Proceedings of the 17th ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2016, pp. 341–350.
[18] Y. Wang, M. Gu, J. Ma, Q. Jin, Dnn-dp: Differential privacy enabled deep neural network learning framework for sensitive crowdsourcing data, IEEE Trans. Comput. Soc. Syst. 7 (1) (2019) 215–224.
[19] W. Mahanan, W. Chaovalitwongse, J. Natwichai, Data privacy preservation algorithm with k-anonymity, World Wide Web 24 (5) (2021) 1551–1561.
[20] S. Gisdakis, T. Giannetsos, P. Papadimitratos, Security, privacy, and incentive provision for mobile crowd sensing systems, IEEE Internet Things J. 3 (5) (2016) 839–853.
[21] Y. Wang, Privacy-preserving average consensus via state decomposition, IEEE Trans. Autom. Control 64 (11) (2019) 4711–4716.
[22] K. Zhang, Z. Li, Y. Wang, A. Louati, J. Chen, Privacy-preserving dynamic average consensus via state decomposition: Case study on multi-robot formation control, Automatica 139 (2022) 110182.
[23] L. An, G.-H. Yang, Enhancement of opacity for distributed state estimation in cyber–physical systems, Automatica 136 (2022) 110087.
[24] Y. Tong, Y. Zeng, B. Ding, L. Wang, L. Chen, Two-sided online micro-task assignment in spatial crowdsourcing, IEEE Trans. Knowl. Data Eng. 33 (5) (2019) 2295–2309.
[25] L. Pu, X. Chen, J. Xu, X. Fu, Crowdlet: Optimal worker recruitment for self-organized mobile crowdsourcing, in: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE, 2016, pp. 1–9.
[26] B. Guo, Y. Liu, L. Wang, V.O. Li, J.C. Lam, Z. Yu, Task allocation in spatial crowdsourcing: Current state and future directions, IEEE Internet Things J. 5 (3) (2018) 1749–1764.
[27] R. Burkard, M. Dell'Amico, S. Martello, Assignment problems: revised reprint, SIAM, 2012.
[28] Z. Song, C.H. Liu, J. Wu, J. Ma, W. Wang, Qoi-aware multitask-oriented dynamic participant selection with budget constraints, IEEE Trans. Veh. Technol. 63 (9) (2014) 4618–4632.
[29] D. Yu, X. Zhang, X. Zhang, L. Zhang, Multitask assignment algorithm based on decision tree in spatial crowdsourcing environment, in: International Conference on Algorithms and Architectures for Parallel Processing, Springer, 2019, pp. 300–314.
[30] M. Xiao, J. Wu, H. Huang, L. Huang, C. Hu, Deadline-sensitive user recruitment for mobile crowdsensing with probabilistic collaboration, in: 2016 IEEE 24th International Conference on Network Protocols (ICNP), IEEE, 2016, pp. 1–10.
[31] J. Wang, Y. Wang, D. Zhang, F. Wang, Y. He, L. Ma, Psallocator: Multi-task allocation for participatory sensing with sensing capability constraints, in: Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, 2017, pp. 1139–1151.
[32] D. Yuan, Q. Li, G. Li, Q. Wang, K. Ren, Priradar: A privacy-preserving framework for spatial crowdsourcing, IEEE Trans. Inform. Forens. Secur. 15 (2019) 299–314.
[33] H. To, C. Shahabi, L. Xiong, Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server, 2018 IEEE 34th international conference on data engineering (ICDE), IEEE 2018 (2018) 833–844.

[34] J. Shu, K. Yang, X. Jia, X. Liu, C. Wang, R.H. Deng, Proxy-free privacy-preserving task matching with efficient revocation in crowdsourcing, IEEE Trans. Dependable Secure Comput. 18 (1) (2018) 117–130.
[35] A. Guttman, R-trees: A dynamic index structure for spatial searching, in: Proceedings of the 1984 ACM SIGMOD international conference on Management of data, 1984, pp. 47–57.
[36] Y. Zheng, X. Xie, W.-Y. Ma, et al, Geolife: A collaborative social networking service among user, location and trajectory, IEEE Data Eng. Bull. 33 (2) (2010) 32–39.