Dynamic Private Task Assignment under Differential Privacy

Leilei Du
East China Normal University
Shanghai, China
leileidu@stu.ecnu.edu.cn

Peng Cheng
East China Normal University
Shanghai, China
pcheng@sei.ecnu.edu.cn

Libin Zheng
Sun Yat-sen University
Guangzhou, China
zhenglb6@mail.sysu.edu.cn

Wei Xi
Xi'an Jiaotong University
Shaanxi, China
xiwei@xjtu.edu.cn

Xuemin Lin
Shanghai Jiao Tong University
Shanghai, China
xuemin.lin@gmail.com

Wenjie Zhang
The University of New South Wales
Sydney, Australia
wenjie.zhang@unsw.edu.au

Jing Fang
East China Normal University
Shanghai, China
jingfang@stu.ecnu.edu.cn

***Abstract*—Data collection is indispensable for spatial crowdsourcing in data computing, such as resource allocation, policy-making, and scientific explorations. However, privacy issues make it challenging for users to share their information unless receiving sufficient compensation. Differential privacy (DP) gives a solution to release helpful information while protect individuals' privacy. However, most DP mechanisms only consider a fixed compensation for each user's privacy loss. In this paper, we design a task assignment scheme that allows workers to dynamically improve their utility with dynamic distance privacy leakage. Specifically, we propose two solutions to improve the matching accuracy of task assignment, Private Utility Conflict-Elimination (PUCE) approach and Private Game Theory (PGT) approach, respectively. We prove that our PUCE achieves higher utility than the current work. We demonstrate the efficiency and effectiveness of our PUCE and PGT approaches on both synthetic and real data sets compared with the recent distance-based approach, Private Distance Conflict-Elimination (PDCE). PUCE is always better than PDCE slightly. PGT costs 50% to 63% less time than PDCE and improve 16% utility on average when worker ratio is large enough.**

***Index Terms*—Spatial Crowdsourcing, Differential Privacy**

I. Introduction

With the growing popularity of cloud computing, spatial crowdsourcing has emerged as a computing paradigm for spatial task solutions involving human participation. Workers are appealed to share their data with servers for computation in exchange for excellent service or satisfactory benefits. However, sometimes workers are reluctant to do so because it may leak their vital privacy (i.e., location). Workers usually compete against other workers within a reachable area to gain as much profit as possible. They also need to weigh their earnings and location privacy in front of the platform.

Differential privacy (DP) [1] is often used to protect individual data in statistics. It trades off the utility and privacy by well designing the privacy budget (ϵ). However, different people have different demands for utility and privacy. For some secret departments, such as confidential agencies, privacy receives great attention. They would rather get high-level privacy protection by sacrificing some utility. While in ride-sharing circumstances, the utility gets more concerns. Taxi drivers seeking higher incomes may even want to serve more passengers regardless of location leakage. So there is a need for different users to adjust their utility by altering the privacy leakage themselves in a platform.

In this paper, we propose a dynamic task assignment scheme that workers can compete for a task to get desired utility at the cost of acceptable privacy leakage. We regard the worker as the agent in game theory and design strategies to compete for tasks by trading off utility and privacy. Consider an example of a matching problem as follows.

As shown in Figure 1, there are three workers: w_1, w_2, w_3, and four tasks: t_1, t_2, t_3, and t_4. Distances between each worker and task are shown in Table I. Each task t_i has a value v_i. t_i publishes v_i and its location l_i to the server. Each worker w_j wants to compete for those tasks with a high utility relevant to task value, distance cost, and privacy cost. w_j first calculates the distance to each task and ranks these distances in ascending order. In order to protect their locations, all workers employ a differential privacy mechanism to disturb their distance values and send these values to the server. After that, the server uses these disturbing distances to get a suboptimal matching. Each worker w_j will see other workers' matching and disturbing distances. Suppose w_j is free (without matching any tasks) and finds that his real distance to task t_i is smaller than the disturbing distance of one of the other workers w_k and his disturbing distance is larger than that of w_k. In that case, he will send a new disturbing distance (with a new larger privacy budget) to the server competing for t_j to get a high utility value.

We list our contributions as follows. (1) We firstly design the model that supports dynamic competition with privacy budget sale in task assignment (PA-TA). We define a new dynamic measurement standard for utility. The utility will decrease with the publishing of task-value distance, which is more realistic than static measurement. (2) We propose a new comparison method, *Partial Probability Comparison Function* (PPCF), between real distance and noise distance. We prove PPCF is

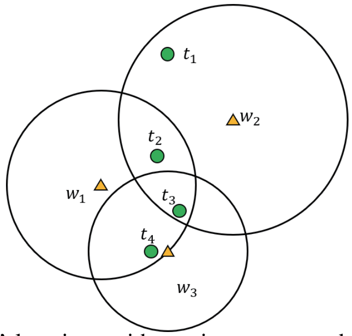


Fig. 1: Workers' locations with serving ranges and tasks' locations.

better than *Probability Compare Function* (PCF) [2] in both theory and practice. (3) We propose two effective solutions Private Utility Conflict-Elimination (PUCE) and Private Game Theory (PGT) for PA-TA. PUCE applies our new partial probability compare function and alter distance-based CEA to utility-based CEA. It can get higher utility than applying only the probability comparison function with distance-based CEA. PGT is based on game iterator and can achieve higher accuracy than PUCE. (4) We realize our approaches in both synthetic and real data sets. The results validate the efficiency and effectiveness of our methods.

We first summarize related works in Section 2. Next, we introduce some preliminaries and declare our problem definition in Section 3. Then we propose our solutions PUCE and PGT in Section 4. After that, we analyze the performance of our two solutions in Section 5. We compare them with other solutions on real and synthetic data sets in Section 6. Finally, we give a conclusion in Section 7.

TABLE I: Task-worker distances.

Worker/Task	t_1	t_2	t_3	t_4
w_1	17.89	8.06	10.44	10.00
w_2	11.31	9.85	12.59	18.87
w_3	24.00	12.04	5.39	2.00

II. Related Work

There has been significant work on task assignments in spatial crowdsourcing. We summarize it with privacy protection under differential privacy. Besides, we summarize the works of compensation allocation, which provides a reward for privacy release.

A. Privacy Protection in Spatial Crowdsourcing

Privacy protection is always an essential concern in spatial crowdsourcing. One of the interests is to protect the location information of tasks and workers. Differential privacy [1] is a golden tool for privacy protection and private data release.

To et al. [3] are the first to study location privacy for spatial crowdsourcing. They adopted *Private Spatial Decomposition* (PSD) [4] to create sanitized data releases of workers and devised a geocast mechanism for task request dissemination. This method can well protect the privacy of workers' locations. However, it needs a trusted entity to help sanitize workers' location data. Wang et al. [5] studied *Bayesian attack* [6], [7] on sparse mobile crowdsourcing and proposed a privacy-preserving framework to reduce the data quality loss caused by differential location obfuscation. They provided the method to get the optimal location obfuscation matrix satisfying ϵ-differential privacy. It can be used to protect workers' location without relying on the trust entity. To et al. [8] proposed a privacy-aware framework that protects the privacy of both tasks and workers in spatial crowdsourcing without assuming any trusted entity. It employs *Geo-indistinguishability* (Geo-I) [6] to transform both tasks' and workers' locations into noise locations. The platform can identify a set of candidate workers for the task requester through these noise locations without knowing the real locations of both workers and the task. These two works get rid of reliance on trusted third parties. However, they are only suitable for individual privacy protection without inspiring tasks or workers to participate in the platform.

B. Private Data Compensation

In order to incentivize users (task requesters and workers) to join while protecting their location privacy, we need a connection between their utility and their privacy cost.

Jin and Zhang [9] provided a framework for spectrum-sensing participants selection, which achieves differential location privacy, approximate social cost minimization, and truthfulness simultaneously. (It's a weak point.) Ghosh et al. [10] modeled the utility of competing agents considering privacy cost. They hold the privacy cost related to both some unknown quantities v and suppose the privacy cost is changing linearly with privacy budget ϵ (ϵv). Nissim et al. [11] argued that ϵv should be the upper bound rather than the total privacy cost. They proposed a privacy-aware mechanism with v below a certain threshold. Xiao [12] proposed two models for quantifying an agent's privacy cost using mutual information and max divergence, respectively. However, it requires the privacy variable $\delta > 0$.

Wang et al. [2] proposed a personalized privacy-preserving task allocation method for mobile crowdsensing. They defined *Probability Compare Function* (PCF), which can be used to compare two noise values with the acknowledgment of their privacy budget. Besides, they proposed *Probabilistic Winner Selection Mechanism* to minimize the total travel distance and *Vickrey Payment Determination Mechanism* to determine the appropriate payment to each winner of workers. The latter mechanism satisfies the truthfulness, profitability, and probabilistic individual rationality. However, all workers can only have a fixed budget for each task, which cannot dynamically complete for hopeful tasks to get better utility values.

III. Problem definition

We first introduce some preliminaries and then give the problem definition of the privacy-aware task assignment problem (PA-TA).

Let $[m, n]$ denote the integers in close interval between m and n. If $m = 1$, we denote it as $[n]$. Let $\boldsymbol{v} = [v^{(1)}, v^{(2)}, ..., v^{(n)}]$ be a vector containing n element. If $v^{(i)} \in \{0, 1\}$ for all $i \in [n]$, we denote $\boldsymbol{v}$ as $\boldsymbol{b} = [b^{(1)}, b^{(2)}, ..., b^{(n)}]$ and call it as *state vector*. Let $CP(\boldsymbol{b})$ be the compression of $\boldsymbol{b}$, which means remove all zero element of $\boldsymbol{b}$. For example, if

$\boldsymbol{b} = [1, 1, 0, 0, 0]$, then $CP(\boldsymbol{b}) = [1, 1]$. Besides, we use $sum(\boldsymbol{v})$ to denote the summation of each element in $\boldsymbol{v}$, and use $min(\boldsymbol{v})$ to denote the minimal element in $\boldsymbol{v}$.

A. Basic Conceptions

Definition 1. (Spatial Tasks). Let t_i denote a task. Its location and value are denoted as l_i and v_i, respectively.

Definition 2. (Spatial Workers). Let w_j denote a worker located at l_j. His service area is denoted as R_j, and his service radius is r_j. We abuse R_j to denote the task set in the service area of w_j.

Definition 3. (One-to-one Match). Let $G = (U, E, V)$ be a bipartite graph, and $M \subseteq E$ be a match in G. M is called a one-to-one match if for any two different edges $e_{u,v}, e_{u',v'} \in M$, we have $e_{u,v} \cap e_{u',v'} = \phi$.

Definition 4. (Differential Privacy [13]). A randomized algorithm $\mathcal{A}$ with domain $\mathbb{N}^{|\mathcal{X}|}$ is (ϵ, δ)-differential private if for all $\mathcal{S} \subseteq \text{Range}(\mathcal{A})$ and for all $x, y \in \mathbb{N}^{|\mathcal{X}|}$ such that $||x-y||_1 \leq 1$:

$$\Pr[\mathcal{A}(x) \in \mathcal{S}] \leq \exp(\epsilon)\Pr[\mathcal{A}(y) \in \mathcal{S}] + \delta.$$

Especially, when $\delta = 0$, $\mathcal{A}$ is ϵ-differential private.

Definition 5. (ℓ_1-sensitivity). The ℓ_1-sensitivity of a function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$ is:

$$\Delta f = \max_{x,y \in \mathbb{N}^{|\mathcal{X}|}, ||x-y||_1 = 1} ||f(x) - f(y)||_1.$$

Definition 6. (The Laplace Mechanism [13]). Given any function $f : \mathbb{N}^{|\mathcal{X}|} \to \mathbb{R}^k$, the Laplace mechanism is defined as:

$$\mathcal{A}_L(x, f(\cdot), \epsilon) = f(x) + (Y_1, ..., Y_k)$$

where Y_i ($i \in [k]$) is an i.i.d. random variable drawn from $Lap(\Delta f/\epsilon)$.

Definition 7. (Probability Compare Function [2]). A function $f : R^4 \to [0, 1]$ is called a probability compare function (PCF) if it takes two sanitized values and the parameters of noise density functions as input and outputs the probability that the first real value is less than the second real value.

For example, if the two real values are d_1 and d_2, the sanitized values are $\hat{d}_1 = d_1 + Lap(0, 1/\epsilon_1)$ and $\hat{d}_2 = d_2 + Lap(0, 1/\epsilon_2)$, where $Lap(a, b)$ is a random variable drawn from Laplace distribution with parameter a, b. A PCF function will take $(\hat{d}_i, \hat{d}_j, \epsilon_1, \epsilon_2)$ as input, and output $\Pr[d_1 < d_2]$.

B. Privacy-aware Task Assignment Problem

Definition 8. (Privacy-aware Task Assignment Problem). Given a set of worker and task distance $\{\hat{d}_{i,j} | i \in [m], j \in [n]\}$ added noise $\eta_{i,j}$ subjecting to distribution $D(\epsilon_{i,j})$, a PA-TA problem is to find a one-to-one match $M = \{M_1, ..., M_n\}$ that satisfies

$$\begin{aligned}
\max \quad & \sum_{t_i \in \mathcal{T}} \sum_{w_j \in \mathcal{W}} (s_{i,j} \cdot (v_i - f_1(d_{i,j})) - f_2(\boldsymbol{b_{i,j}} \cdot \boldsymbol{\epsilon_{i,j}})) \\
s.t. \quad & \sum_{t_i \in \mathcal{T}} s_{i,j} \leq 1, \quad \forall i = 1, 2, ..., m \\
& \sum_{w_j \in \mathcal{W}} s_{i,j} \leq 1, \quad \forall j = 1, 2, ..., n \\
& \sum_{z \in Z} b_{i,j}^{(z)} \leq Z, \quad \forall z = 1, 2, ..., Z \\
& s_{i,j}, b_{i,j} \in \{0, 1\}, \ \forall i = 1, 2, ..., m; \forall j = 1, 2, ..., n
\end{aligned}$$

where $s_{i,j}$ is the matching state representing whether task t_i is allocated to worker w_j. $s_{i,j}$ equals 1 if t_i is allocated to w_j, and 0 otherwise. Function $f_1$is the function that transforms distance to value cost. Function $f_2$is the function that changes privacy cost to value cost. $\boldsymbol{\epsilon_{i,j}} = \langle \epsilon_{i,j}^{(1)}, ..., \epsilon_{i,j}^{(Z)} \rangle$ is the privacy budget vector between task $t_i$and worker w_j where $\epsilon_{i,j}^{(u)} (u \in Z)$ stands for the u-th apply of worker w_j for task t_i. $\boldsymbol{b_{i,j}} = \langle b_{i,j}^{(1)}, ..., b_{i,j}^{(Z)} \rangle$ is the the state vector corresponding to $\boldsymbol{\epsilon_{i,j}}$. Take $\boldsymbol{b_{1,2}} = \langle 1, 1, 0, 0, 0 \rangle$ as an example. It means in the total competition, w_2 can apply for t_1 five times and has already applied two times with the privacy leakage $\epsilon_{1,2}^{(1)}$ and $\epsilon_{1,2}^{(2)}$.

IV. Solution

In this section, we first introduce some new conceptions and techniques, then propose *Private Utility Conflict-Elimination* (PUCE) and *Private Game Theory* (PGT).

A naive method to solve our matching problem is collecting all workers' applications for tasks with privacy budgets and noise distances and using the Hungary algorithm to get the optimal matching. However, using the Hungary algorithm, we have to compare the path length calculated by many noise distances. The sum of those noise distances leads to complex comparison and low accuracy.

A. Conflict Elimination Algorithm (CEA)

In order to simplify the comparison and diminish the deviation of noise distance summation, we replace the Hungary algorithm with CEA [2], which is designed to solve the winner conflict problem. Given all distances from each task-worker pair, we construct the distance rank matrix $A_{m \times n} = (a_{i,k})_{m \times n}$ where $a_{i,k}$ stands for the index of the worker who is k nearest from t_i. Thus $a_{i,k} = j$ means w_j is the k nearest worker of t_i.

The main idea of CEA is that, for any conflict worker w_c selected by φ tasks, CEA allocates only one task to w_c and finds another candidate other than w_c for each of the rest $\varphi - 1$ conflict tasks.

For each conflict worker w_c, there will be φ candidate distance choices shown in equation 1. $\mathcal{C}_u (1 \leq u \leq \varphi)$ stands for the u-th solution of solving this conflict with task t_{c_u} allocated to w_c and other tasks allocated to the successive workers.

$$\begin{cases}
\mathcal{C}_1 : D_{c_1} = D(a_{c_1,1}) + D(a_{c_2,2}) + ... + D(a_{c_k,2}) \\
\mathcal{C}_2 : D_{c_2} = D(a_{c_1,2}) + D(a_{c_2,1}) + ... + D(a_{c_k,2}) \\
... \\
\mathcal{C}_{c_\varphi} : D_k = D(a_{c_1,2}) + D(a_{c_2,2}) + ... + D(a_{c_k,1})
\end{cases} \tag{1}$$

Deciding which of the φ solutions to use requires comparing four distance values. Suppose we choose one of $\mathcal{C}_u$ and $\mathcal{C}_v$ $(1 \leq u, v \leq \varphi)$. We need to compare $D(a_{c_u,1}) + D(a_{c_v,2})$ with $D(a_{c_v,1}) + D(a_{c_u,2})$. If the distance is noise distance, we have to get the result from four Laplace random variables. In CEA, it supposes the difference between the travel distances for different tasks is relatively small for the same worker. That is, it sets $D(a_{c_u,1}) = D(a_{c_v,1})$. In this way, the comparison is reduced to compare two Laplace random variables, which can be calculated by *Probability Compare Function* [2].

Algorithm 1: Strategy Table Generation

Input: The distance from each worker w_j to each task t_i: $\{d_{1,1}, d_{1,2}, ..., d_{m,n}\}$, the privacy budget sequence for each task-worker pair (t_i, w_j): $\{\epsilon_{i,j}^{(1)}\ \epsilon_{i,j}^{(2)}\ ..., \epsilon_{i,j}^{(Z)}\}$, the noise distribution D
Output: Strategy table ST

```
Initialize ST as a table with n rows
for each worker w_j in W do
    Initialize ST[j] as a table with m rows and Z columns;
    for each task t_i in T do
        for each column u in [Z] do
            Draw η_{i,j}^{(z)} from D(ε_{i,j}^{(z)});
            Set ST[j][i][u] as (d_{i,j} + η_{i,j}^{(z)}, ε_{i,j}^{(u)});
    Sort ST[j][i] by ε_{i,j}^{(u)} in ascending order;
return ST;
```

B. Strategy Table Generation

In order to facilitate the competition process, each worker w_j extracts a series of noise $\boldsymbol{\eta_{i,j}} = [\eta_{i,j}^{(1)}, \eta_{i,j}^{(2)} ..., \eta_{i,j}^{(Z)}]$ from Laplace distribution for each task t_i with privacy budget array $\boldsymbol{\epsilon_{i,j}} = [\epsilon_{i,j}^{(1)}, \epsilon_{i,j}^{(2)}, ..., \epsilon_{i,j}^{(Z)}]$, where Z is the array size. w_j calculates the distance $d_{i,j}$ to t_i and the relevant noise distance vector $\boldsymbol{\hat{d}_{i,j}} = [\hat{d}_{i,j}^{(1)}, ..., \hat{d}_{i,j}^{(Z)}]$. We combine $\hat{d}_{i,j}^{(z)}$ and $\epsilon_{i,j}^{(z)}$ as a pair and sort each pair by budget $\epsilon_{i,j}^{(z)}$ in ascending order. We call all pairs for each task and worker as Strategy Table. The algorithm for Strategy Table generation is shown in Algorithm 1.

Especially, if there is only one task t, we omit the index i and denote $\boldsymbol{\eta_j} = [\eta_j^{(1)}, \eta_j^{(2)} ..., \eta_j^{(Z)}]$ as w_j's noise series, and $\boldsymbol{\epsilon_j} = [\epsilon_j^{(1)}, \epsilon_j^{(2)}, ..., \epsilon_j^{(Z)}]$ as w_j's privacy budget array.

C. Utility function

For each worker w_j, we define the utility function as

$$U_j(i) = v_i - f_1(d_{i,j}) - \sum_{t_i \in \mathcal{T}} f_2(\boldsymbol{b_{i,j}} \cdot \boldsymbol{\epsilon_{i,j}}).$$

Notice that the competition process may execute many rounds. In each round, each worker will compare their distance to tasks with other workers' and judge whether to release a more accurate distance with a larger privacy budget. The state vector $\boldsymbol{b_{i,j}}$ will change if the w_j release a new noise distance and privacy budget for w_i.

D. Partial Probability Compare Function

Before detailing the competition process, we need to redefine our probability compare function to make the probability more accurate. Suppose there are two values d_i and d_j. The sanitized value of d_j is $\hat{d}_j$, which is calculated by adding noise η_j drawn from a type of distribution $D(\epsilon_j)$. So we have

$$\hat{d}_j = d_j - \eta_j, \quad \eta_j \sim D(\epsilon_j),$$

$$\begin{aligned}\Pr[d_i < d_j] &= \Pr[d_i < \hat{d}_j - \eta_j] \\ &= \Pr[\eta_j < \hat{d}_j - d_i].\end{aligned}$$

Let $f(x)$ be the probability density function of η_j, then we have

$$\Pr[d_i < d_j] = \int_{-\infty}^{\hat{d}_j - d_i} f(\eta_j) d\eta_j.$$

Similar to PCF, we define our PPCF as $\text{PPCF}(d_i, \hat{d}_j, \epsilon_j) = \Pr[d_i < d_j]$. If the distribution of $D(\epsilon_j)$ is symmetric about the y axis (i.e., Laplace distribution), then

$$\text{PPCF}(d_i, \hat{d}_j, \epsilon_j) > \frac{1}{2} \Leftrightarrow d_i < \hat{d}_j. \tag{2}$$

E. Effective Noise Distance and Effective Privacy Budget

For analysts, we suppose that they adopt *maximum likelihood estimation* (MLE) to get the final distance release from a worker w's release set $\boldsymbol{DE} = \{(\hat{d}_1, \epsilon_1), (\hat{d}_2, \epsilon_2), ..., (\hat{d}_u, \epsilon_u)\}$ for a task t.

Let $\boldsymbol{DE.\hat{d}}$ denote the set $\{\hat{d}_1, \hat{d}_2 ..., \hat{d}_u\}$ in $\boldsymbol{DE}$. Let $\boldsymbol{DE.\epsilon}$ denote the set $\{\epsilon_1, \epsilon_2 ..., \epsilon_u\}$ in $\boldsymbol{DE}$. Let $L(X) = L(\hat{d}_1, \hat{d}_2, ..., \hat{d}_u; X) = \prod_{k=1}^{u} \Pr[\hat{d}_k; X]$, where $\Pr[\hat{d}_k; X]$ is the probability function of $Lap(\epsilon_k)$. When analysts get $\boldsymbol{DE}$, they would like to get the estimation of d as follow.

$$\begin{aligned}\check{d} &= \arg\max_d \prod_{k=1}^{u} \frac{\epsilon_k}{2} \exp(-|\hat{d}_k - d| \cdot \epsilon_k) \\ &= \arg\min_d \sum_{k=1}^{u} \epsilon_k \cdot |\hat{d}_k - d|.\end{aligned}$$

In order to simplify, we limit the domain of $\hat{d}$ in set $\boldsymbol{DE.\hat{d}}$ and denote the noise distance estimation as $\tilde{d}$. We call $\tilde{d}$ as *Effective noise distance* and its privacy budget $\tilde{\epsilon}$ in $\boldsymbol{DE.\epsilon}$ as *Effective privacy budget*. We denote the pair $(\tilde{d}, \tilde{\epsilon})$ as EDE and call it as *effective distance-budget pair*.

Next, we define our *Effective value function* (EVF).

Definition 9. (Effective Value Function). Given a release set of noise distance-privacy budget pairs, $\boldsymbol{DE}$ and the state vector $\boldsymbol{b}$, a function $f : R^{3U} \rightarrow R^2$ is called *Effective value function* (EVF) if it takes $\boldsymbol{DE}$ and $\boldsymbol{b}$ as the input, and outputs the effective noise distance and effective privacy budget of subset $S \subseteq \boldsymbol{DE}$ satisfying that $S = \{(\hat{d}_i, \epsilon_i) | (\hat{d}_i, \epsilon_i) \in \boldsymbol{DE}, b_i = 1\}$.

F. Variance Value Transforms

We define our distance value function f_1 and privacy budget value function f_2 as follows.

Definition 10. (Distance Value Function (f_1)). Given a distance $d \in R^*$, a function $f_1 : R^* \rightarrow R^*$ is called *distance value function* if it takes d as the input and outputs a value v, satisfying that $f_1(0) = 0$, $f_1^{'}(\cdot) \geq 0$.

Definition 11. (Privacy Budget Value Function (f_2)). Given a privacy budget $\epsilon \in R^*$, a function $f_2 : R^* \rightarrow R^*$ is called *privacy budget value function* if it takes ϵ as input and outputs a value v, satisfying that $f_2(0) = 0$, $f_2^{'}(\cdot) \geq 0$ and $\forall \epsilon_1, \epsilon_2 \in R, f_2(\epsilon_1) + f_2(\epsilon_2) = f_2(\epsilon_1 + \epsilon_2)$.

G. Private Utility Conflict-Elimination Solution

We first consider the most straightforward circumstance $\text{Cond}_{1,n}$ that contains one task and n workers satisfying that the task is in all workers' serving range. After that, we extend it to the condition $\text{Cond}_{m,n}$ that satisfies multi-tasks and multi-workers with different serving range for each worker. And we propose an effective solution PUCE to this problem.

Algorithm 2: OneTaskCompetition

Input: Task t, All workers $\mathcal{W}$
Output: The winner worker w_k
1 Initialize w_{win} as the fake worker w_{-1} with effective distance $\hat{d}^{(e)}_{win} = Inf$ and effective privacy budget $\epsilon^{(e)}_{win} = Inf$;
2 Initialize candidate competition workers CW as all workers;
3 Initialize published noise distance-privacy budget pair set PDB as null;
4 **while** *CW is not empty* **do**
5   Set next candidate worker set NCW as empty;
6   **for** *each worker w_j in CW* **do**
7   **if** *w_j is w_{win}* **then**
8   continue;
9   **if** *w_j's privacy budget has been exhausted* **then**
10   continue;
11   **if** $U_j(t) \leq 0$ **then**
12   continue;
13   Set new privacy budget $\epsilon_j = ST_{j,c_j}.budget$;
14   Set new release distance $\hat{d}_j = ST_{j,c_j}.distance$;
15   Add $(\hat{d}_j, \epsilon_j)$ to PDB_j;
16   Get effective value $[\hat{d}^{(e)}_j, \epsilon^{(e)}_j] = EVF(PDB_j)$;
17   **if** $\mathrm{PCF}(\hat{d}^{(e)}_j, \epsilon^{(e)}_j, \hat{d}^{(e)}_{win}, \epsilon^{(e)}_{win}) \leq 0.5$ **then**
18   continue;
19   Add w_j to NCW;
20     $c_j = c_j + 1$;
21   **for** *w_j in NCW* **do**
22   **if** $PCF(\hat{d}^{(e)}_j, \epsilon^{(e)}_j, \hat{d}^{(e)}_{win}, \epsilon^{(e)}_{win}) > 0.5$ **then**
23   Replace w_k with w_j;
24   Replace $\hat{d}^{(e)}_{win}$ with $\hat{d}^{(e)}_j$;
25   Replace $\epsilon^{(e)}_{win}$ with $\epsilon^{(e)}_j$;
26 **return** w_k;

1) One task and multiple workers: Let t be the unique task with location l and value v, w_j be the worker with location l_j ($j \in [n]$). Suppose t is in the range of all workers. At first, t publishes its location and value. After that w_j calculates his distance d_j to t and generates his strategy table. Then all workers compete for t.

In the competing process, all the workers first participate in the competition set called CW. Each worker in CW will execute a series of judgments to test whether he has advantages and will gain more utility to compete. If so, he will be added to the next new candidate set NCW, which will be used for the server to pick out the final winner. The competing algorithm is shown in Algorithm 2. Each worker has four judgments to determine whether he can compete for t. The first three are from line 7 to line 12, meaning that if the current worker is already the winner or his privacy budget has been exhausted or competing for t does not give him positive utility value, then he will not propose. The fourth one shown from line 17 to line 18 means that the current worker compare his distance with the previous winner's using PCF. He will give up the competition if his distance value to t is no less than the previous winner's.

At the end of each proposing round, the server will choose a winner with a short distance to t as the new winner. The competition process continues until no workers compete in the next round.

2) m tasks and n workers: We suppose that each worker are allowed to complete multiple tasks at a time when there are m tasks. Let t_i denote the i-th task with location l_i ($i \in [m]$).

Different from the one-task circumstance, w_j can choose a set of different tasks in different rounds considering the last round competing of all tasks. Besides, the server needs to eliminate conflict for each proposed task. We propose a method, Private Utility Conflict-Elimination (PUCE), to decide which tasks a worker will apply in the next round and which workers will be matched with each task for the server.

Algorithm 3: WorkerApplication

Input: Not winning worker set NWW
Output: Candidate list CL
Initialize candidate list CL as m empty sets;
for *each worker w_j in NWW* **do**
  Initialize applicable list BL as empty list;
  for *each task $t_i \in R_j$* **do**
    if *w_j's privacy budget has been exhausted* **then**
      continue;
    Calculate new $U_j(i)$;
    if $U_j(i) \leq 0$ **then**
      continue;
    Get $EDB_{i,win(i)} = [\tilde{d}_{i,win(i)}, \tilde{\epsilon}_{i,win(i)}]$;
    Calculate new $EDB_{i,j} = [\tilde{d}_{i,j}, \tilde{\epsilon}_{i,j}]$;
    Calculate $\tilde{d}'_{i,win(i),j}$ by Equation 3;
    if *PPCF*$(d_{i,j}, \tilde{d}'_{i,win(i),j}, \epsilon_{i,win(i)}) \leq 0.5$ **then**
      continue;
    if *PCF*$(\tilde{d}_{i,j}, \tilde{d}'_{i,win(i),j}, \tilde{\epsilon}_{i,win(i)}, \tilde{\epsilon}_{i,j}) \leq 0.5$ **then**
      continue;
    Add $\tilde{d}_{i,j}$ to $CL[i]$
return CL;

As for the process of worker proposing, we notice that w_j may not be willing to apply for only one task in each round because when this task is competed by many other workers, the failure probability of w_j will be much higher.

We suppose that w_j will apply for all tasks T_j in his serving range R_j. In order to further decline unnecessary privacy costs, we add an extra judgment for workers by using the PPCF function.

As for the process of winner choice, the server needs to eliminate conflict among workers for each task. We can easily use CEA directly to choose only one worker for each task. However, in CEA, the comparison is based on noise distance rather than utility function, which may not satisfy our optimized goal. If we use utility value as a comparison, the server must know the utility value in each round, which will leak the real distance between tasks and workers.

In order to handle the problem above, we convert the utility into distance and use CEA to choose the high-utility one under the distance form. For any two utilities $U_a(x)$ and $U_b(y)$, let $V_a(x) = U_a(x) + f_1(d_{x,a})$ and $V_b(y) = U_b(y) + f_1(d_{y,b})$. Then we have

$$\begin{aligned}\Pr(U_a(x) > U_b(y)) &= \Pr(V_a(x) - f_1(d_{x,a}) > V_b(y) - f_1(d_{y,b})) \\ &= \Pr(f_1^{-1}(V_a(x)) - d_{x,a} > f_1^{-1}(V_b(y)) - d_{y,b}) \\ &= \Pr(d_{x,a} < d_{y,b} + f_1^{-1}(V_a(x)) - f_1^{-1}(V_b(y))),\end{aligned}$$

$$\hat{d}'_{y,b,a} = \hat{d}_{y,b} + f_1^{-1}(V_a(x)) - f_1^{-1}(V_b(y)), \tag{3}$$

Thus, $$\begin{aligned}\Pr(U_a(x) > U_b(y)) &= \Pr(\eta_{x,a} - \eta_{y,b} > \hat{d}_{x,a} - \hat{d}'_{y,b,a}) \\ &= \mathrm{PCF}(\hat{d}_{x,a}, \hat{d}'_{y,b,a}, \epsilon_{x,a}, \epsilon_{y,b}).\end{aligned}$$

Therefore, we can calculate $\hat{d}'_{y,b,a}$ for each pair of w_a and w_b with the same task t_y and use PCF function to compare the utility. Similarly, we can compare $U_a(x)$ and $U_a(x)$ through PPCF:

Algorithm 4: WinnerChosen

```
   Input: Candidate list CL, last term allocation list AL'
   Output: Allocation list AL, updating state upd
1  if All set in CL are empty then
2      return (AL', false)
3  Initialize each update task set list TSL as empty list;
4  Initialize AL as m null values;
5  Initialize competing table CT as empty table;
6  for Each candidate set CS_i in CL do
7      if CS_i is empty then
8          Set AL[i] = AL'[i];
9      else
10         Add CT[i] = CS_i ∪ {d̃_{i,win(i)}} to CT;
11         Calculate d̂'_{i,a,b} for each pair in CT[i];
12         Sort CT[i] in descending order by PCF(d̂'_{i,a,b}, d̂_{i,b}, ε_{i,a}, ε_{i,b});
13 Get updated matching M set by using CEA for CT;
14 Add M each AL;
15 return (AL, true);
```

$$\begin{aligned}\Pr(U_a(x) > U_b(y)) &= \Pr(\eta_{y,b} < \hat{d}'_{y,b,a} - d_{x,a}) \\ &= \text{PPCF}(d_{x,a}, \hat{d}'_{y,b,a}, \epsilon_{y,b}).\end{aligned}$$

The worker application process and winner-chosen algorithms are respectively shown in Algorithm 3 and Algorithm 4.

In Algorithm 3, each worker w_j checks all the tasks in his serving range and judges whether it is worth to complete for the tasks (check whether $U_j(i) > 0$ for $t_i \in R_j$). Besides, he also judges whether he has advantages over the before-winner worker for these tasks by utility comparison. The utility comparison is shown from line 10 to line 16. If the two conditions are satisfied, w_j will apply for this task with a new privacy budget and noise distance.

Algorithm 4 takes candidate list CL (constructed by Algorithm 3) and last term allocation list AL' as the input. It outputs the updating allocation list with the updating state upd. The $false$ value of upd means there is no change for AL. The candidate list will be partitioned into two parts. Ones with no workers' application are the same as the last term ones, which is shown from line 7 to line 8. The others containing workers' applications will be added to a new competing table with the winners of the last term. Each set of workers for applied tasks in competing table will be sorted by the utility value (compared by PCF($\hat{d}'_{x,a,b}, \hat{d}_{x,b}, \epsilon_{x,a}, \epsilon_{x,b}$)) in descending order. The process is shown from line 10 to line 12.

By executing Algorithm 3 and Algorithm 4, we can construct our PUCE algorithm as shown in Algorithm 5. The process is shown in Figure 2. In the beginning, the not winning worker set NWW is initialized as the whole worker set W, and the allocation list AL' is initialized as an empty set. We execute Algorithm 3 to get candidate allocation list CL. Then we execute Algorithm 4 to pick a new allocation list AL and a updating state upd. When there are still some workers applying for tasks (CL is not empty), upd will be set as $true$. We will update NWW (by adding new winner workers), remove the losers and update AL' as AL. When no workers apply for any task, upd will be set as $false$. Thus we get the final task-worker matching pairs TWM as AL.

Algorithm 5: PUCE

```
Input: All task 𝒯, All workers 𝒲
Output: The task-worker matching pairs TWM
Initialize not winning worker set NWW as 𝒲;
Initialize halt state HS as false;
Initialize allocation list AL as m empty set list;
while hs is not true do
    Get CL by executing Algorithm 3;
    Get AL and upd by executing Algorithm 4;
    Set hs = upd;
Set TWM as AL;
return TWM;
```

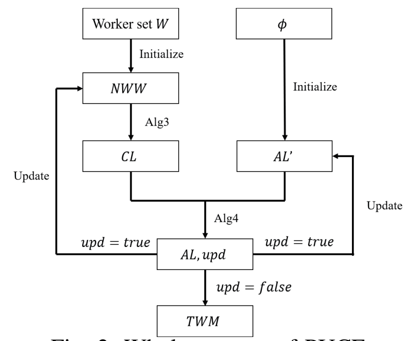


Fig. 2: Whole process of PUCE

H. Private Game Theory Solution

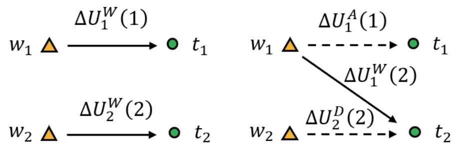
In this section, we declare that each worker can compete for each task within their serving, whether or not they have already won a task. We modeled our problem as an exact potential game with at least one Nash equilibrium in pure strategy. To compare the utility values to make a choice, we approximate our utility function by replacing real distance with effective noise distance.

1) Cases of Utility Change in Competition: There are three cases of utility change in each time of competition for each task-worker pair. They are *Wining Change*, *Abandoned Change* and *Defeated Change*. We denote them as $\Delta U_j^W(i)$, $\Delta U_j^A(i)$ and $\Delta U_j^D(i)$ respectively, which are expressed as follows:

$$\Delta U_j^W(i) = v_i - f_1(\tilde{d}_{i,j}) - f_2(\epsilon_{i,j}^{(z)}),$$
$$\Delta U_j^A(i) = -v_i + f_1(\tilde{d}_{i,j}),$$
$$\Delta U_j^D(i) = -v_i + f_1(\tilde{d}_{i,j}).$$

$\Delta U_j^W(i)$ means the utility change of winning task t_i for worker w_j. $\Delta U_j^A(i)$ means the utility change of abandoning task t_i (because each worker can only match at most one task) for worker w_j. $\Delta U_j^D(i)$ means the utility change of being defeated by some other competitor in competing for task t_i for worker w_j. It is the same with $\Delta U_j^D(i)$. We use $\Delta U_j^{W(k)}(i)$, $\Delta U_j^{A(k)}(i)$ and $\Delta U_j^{D(k)}(i)$ to denote the above three utility change in k-th competition.

We give examples of these three utility changes. Suppose there are two workers w_1, w_2 and two tasks t_1, t_2. At the first stage, w_1 competes for t_1 and w_2 competes for t_2. Then the corresponding $\Delta U_1^W(1)$ and $\Delta U_2^W(2)$ are shown in Figure 3(a). At the second stage, w_1 competes for t_2 and gets it successfully. As is shown in Figure 3(b). The utility change between w_1 and t_2 is $\Delta U_1^W(2)$. The utility change



(a) State 1 (b) State 2

Fig. 3: Utility change.

between w_1 and t_1 is $\Delta U_1^A(1)$. The utility change between w_2 and t_2 is $\Delta U_2^D(2)$.

2) Game Modeling and Nash Equilibrium: We approximate our PA-TA as *Privacy-aware Approximate Task Assignment* (PAA-TA) problem by replacing the real distance as effective distance. We formulate PAA-TA as an n-player strategic game, $\mathcal{G} =< \mathcal{W}, \boldsymbol{S}, \boldsymbol{UT} >$. $\mathcal{G}$ consists of players $\mathcal{W}$, strategy spaces $\boldsymbol{S}$, and utility functions $\boldsymbol{UT}$. We specify these three components as follows:

(1) $\mathcal{W} = \{w_1, ..., w_n\}$ denotes the finite set of n workers with $n \geq 2$. We will use worker and player interchangeably in the rest of the paper.

(2) $\boldsymbol{S} = \{S_j\}_{j=1}^n$ is the strategy spaces (i.e., the overall strategy set of all players). S_j is the finite set of strategies available to worker w_j. We define $S_j = \{\{ST[j][i][u]\}_{z=1}^Z\}_{t_i \in w_j.R} \cup \{null\}$, where $ST[j][i][z]$ is an element of the strategy table defined before and $w_j.R$ stands for the task set in the serving range of w_j.

(3) $\boldsymbol{UT} = \{UT_j^{(k)}\}_{j=1}^n$ is the utility functions of all players w_j where k is the total competition number[1]. For each chosen strategy $\boldsymbol{st} \in S$, $UT_j^{(k)}(\boldsymbol{st}) \in \mathbb{R}$ is the utility of player w_j. We calculate $UT_j^{(k)}(\boldsymbol{st})$ as follows:

$$\begin{aligned} UT_j^{(k)}(\boldsymbol{st}) &= \Delta U_j^{W(k)}(i_2) + \Delta U_{win(i_2)}^{D(k-1)}(i_2) + \Delta U_j^{A(k-1)}(i_1) \\ &= v_{i_2} - f_1(\tilde{d}_{i_2,j}^{(k)}) - f_2(\epsilon_{i_2,j}^{(z_k)}) - v_{i_2}. \end{aligned} \tag{4}$$

In equation 4, w_j wins t_{i_1} and $w_{win(i_2)}$ wins t_{i_2} in $(k-1)$-th competition. w_j will compete for t_{i_2} in k-th competition.

In the following part, we define *exact potential game* (EPG) and prove that PAA-TA is an EPG.

Definition 12. (Exact Potential Game). A strategic game, $\mathcal{G} =< \mathcal{W}, \boldsymbol{S}, \boldsymbol{UT} >$, is an Exact Potential Game (EPG) if there exists a function, $\Phi : \boldsymbol{S} \rightarrow \mathbb{R}$, such that for all $\boldsymbol{st}_j \in \boldsymbol{S}$, it holds that, $\forall w_j \in \mathcal{W}$, $\forall k \in N^+$,

$$\begin{aligned} &UT_j^{(k)}(st'_j, \boldsymbol{st}_{-j}) - UT_j^{(k)}(st_j, \boldsymbol{st}_{-j}) \\ =& - \Phi^{(k)}(st'_j, \boldsymbol{st}_{-j}) - \Phi^{(k)}(st_j, \boldsymbol{st}_{-j}). \end{aligned}$$

Theorem IV.1. *PAA-TA is an Exact Potential Game (EPG).*

Proof: We define a potential function as

$$\Phi^{(k)}(\boldsymbol{st}) = \sum_{t_i \in \mathcal{T}} \sum_{w_j \in \mathcal{W}} (s_{i,j}^{(k)} \cdot (v_i - f_1(\tilde{d}_{i,j})) - f_2(\boldsymbol{b_{i,j}^{(k)}} \cdot \boldsymbol{\epsilon_{i,j}}))$$

which represents the total utility value of the matching result in k-th competition that all worker gain. Let $\tilde{U}_j^{(k)}(i) = v_i - f_1(\tilde{d}_{i,j}^{(k)}) - \sum_{t_i \in \mathcal{W}} f_2(\boldsymbol{b_{i,j}^{(k)}} \cdot \boldsymbol{\epsilon_{i,j}})$ be the approximate value of $U_j(i)$ by replacing the real distance $d_{i,j}$ with the effective noise distance $\tilde{d}_{i,j}$. Then we get the recurrence relation of $\tilde{U}_j^{(k)}(i)$ for k as

$$\tilde{U}_j^{(k)}(i) = \begin{cases} \tilde{U}_j^{(k-1)}(i) + v_i - f_1(\tilde{d}_{i,j}^{(k)}) - f_2(\epsilon_{i,j}^{(z_k)}) & \sharp 1 \\ \tilde{U}_j^{(k-1)}(i) - v_i + f_1(\tilde{d}_{i,j}^{(k-1)}) & \sharp 2 \\ \tilde{U}_j^{(k-1)}(i) & \sharp 3 \end{cases}$$

where condition $\sharp 1$ means w_j wins t_i in k-th competition, condition $\sharp 2$ means w_j gives up his original task or is defeated in k-th competition and condition $\sharp 3$ means others. Suppose that w_j, w_{j_x}, w_{j_y} wins $t_{i_1}, t_{i_2}, t_{i_3}$ in $(k-1)$-th competition respectively and w_j will compete for t_{i_2} (st_j) or t_{i_3} (st'_j) in k-th competition, then we obtain

$$\begin{aligned} &\Phi^{(k)}(st'_j, \boldsymbol{st}_{-j}) - \Phi^{(k)}(st_j, \boldsymbol{st}_{-j}) \\ =&\tilde{U}_j^{(k)}(i_1) + \tilde{U}_j^{(k-1)}(i_2) + \tilde{U}_j^{(k)}(i_3) + \tilde{U}_{j_y}^{(k)}(i_3) + \tilde{U}_{j_x}^{(k-1)}(i_2) \\ &- (\tilde{U}_j^{(k)}(i_1) + \tilde{U}_j^{(k)}(i_2) + \tilde{U}_j^{(k-1)}(i_3) + \tilde{U}_{j_y}^{(k-1)}(i_3) + \tilde{U}_{j_x}^{(k)}(i_2)) \\ =&\tilde{U}_j^{(k)}(i_3) - \tilde{U}_j^{(k-1)}(i_3) - (\tilde{U}_j^{(k)}(i_2) - \tilde{U}_j^{(k-1)}(i_2)) \\ &+ \tilde{U}_{j_y}^{(k)}(i_3) - \tilde{U}_{j_y}^{(k-1)}(i_3) - (\tilde{U}_{j_x}^{(k)}(i_2) - \tilde{U}_{j_x}^{(k-1)}(i_2)) \\ =&v_{i_3} - f_1(\tilde{d}_{i_3,j}^{(k)}) - f_2(\epsilon_{i_3,j}^{(z_k)}) - (v_{i_2} - f_1(\tilde{d}_{i_2,j}^{(k)}) - f_2(\epsilon_{i_2,j}^{(z_k)})) \\ &- v_{i_3} + f_1(\tilde{d}_{i_3,j_y}^{(k-1)}) - (-v_{i_2} + f_1(\tilde{d}_{i_2,j_x}^{(k-1)})) \\ =& - f_1(\tilde{d}_{i_3,j}^{(k)}) - f_2(\epsilon_{i_3,j}^{(z_k)}) + f_1(\tilde{d}_{i_3,j_y}^{(k-1)}) \\ &+ f_1(\tilde{d}_{i_2,j}^{(k)}) + f_2(\epsilon_{i_2,j}^{(z_k)}) - f_1(\tilde{d}_{i_2,j_x}^{(k-1)}) \\ =&v_{i_3} - f_1(\tilde{d}_{i_3,j}^{(k)}) - f_2(\epsilon_{i_3,j}^{(z_k)}) - v_{i_3} + f_1(\tilde{d}_{i_3,j_y}^{(k-1)}) - v_{i_1} + f_1(\tilde{d}_{i_1,j}^{(k-1)}) \\ &- (v_{i_2} - f_1(\tilde{d}_{i_2,j}^{(k)}) - f_2(\epsilon_{i_2,j}^{(z_k)}) - v_{i_2} + f_1(\tilde{d}_{i_2,j_x}^{(k-1)}) - v_{i_1} + f_1(\tilde{d}_{i_1,j}^{(k-1)})) \\ =&\Delta U_j^{W(k)}(i_3) + \Delta U_{j_y}^{D(k-1)}(i_3) + \Delta U_j^{A(k-1)}(i_1) \\ &- (\Delta U_j^{W(k)}(i_2) + \Delta U_{j_x}^{D(k-1)}(i_2) + \Delta U_j^{A(k-1)}(i_1)) \\ =&UT_j^{(k)}(st'_j, \boldsymbol{st}_{-j}) - UT_j^{(k)}(st_j, \boldsymbol{st}_{-j}) \end{aligned}$$

According to Definition 12, the strategic game of the PAA-TA is an exact potential game. So PAA-TA has pure Nash equilibrium.

3) Key Process: The server executes the competition process with the aid of workers. Each worker w_j needs to repeat choosing the best task t_b for the maximal utility value. If the maximal value is positive, w_j will update his effective distance-budget pair for t_b and ask the server to update the allocation list.

We give the process in Algorithm 6. The critical step is getting the best response information (maximal UT_j) shown in line 6. The state variable hs is a boolean variable which indicates whether there still exists a task that can improve a utility function UT_j for any $w_j \in \mathcal{W}$. If there is no such task, the process will halt.

We give an example of the whole process as follows. As shown in Figure 4, suppose there are three workers w_1, w_2 and w_3 with serving ranges 15, 15 and 10 respectively. And there are three tasks t_1, t_2 and t_3 with task values 8, 9 and 7 respectively. The distance between each task and each worker is shown in Table II.

Suppose there are three privacy budgets for each task-worker pair. The corresponding effective distance and the privacy budget are shown in Table III.

As shown in Table IV, suppose in the k-th competition, the winners of t_1, t_2 and t_3 are w_1, w_2 and w_3 respectively. And suppose f_1 and f_2 are both identity functions (e.t. $f_1(x) = x$, $f_2(x) = x$).

[1]The total competition number is initialized as 0 and increases by 1 when there is a player compete

Algorithm 6: PGT

Input: All task $\mathcal{T}$, All workers $\mathcal{W}$
Output: The allocation list AL

```
Initialize AL as a list with m null value
Initialize halt state hs as false;
while hs is false do
    Set hs as true;
    for each worker w_j ∈ W do
        Get the maximal UT_j for each task t_i ∈ R_j \ {AL[b]};
        if UT_j(st) is null or UT_j(st) ≤ 0 then
            continue;
        Set hs as false;
        Set t_c as w_j's already wined task;
        Set t_b as the task with maximal UT_j;
        Set w_f as the worker wined t_b before;
        Update effective distance-budget pair between t_b and w_j;
        Set AL[c] = null;
        Set AL[b] = w_j;
return AL;
```

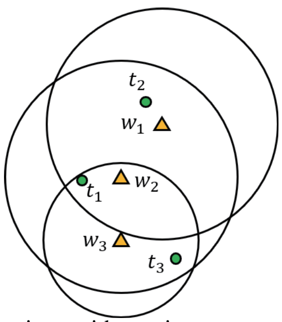


Fig. 4: Workers' locations with serving ranges and tasks' locations.

In the $(k+1)$-th competition, it is w_1's turn to compete. w_1 can only compete for t_2. He first uses his new privacy budget $\epsilon_{2,1}^{(z_k+1)} = \epsilon_{2,1}^{(2)} = 0.3$ and calculates the new effective noise distance $\tilde{d}_{2,1}^{(k+1)} = 4.01$. After that, he calculates $UT_1^{(k+1)} = -f_1(\tilde{d}_{2,1}^{(k+1)}) - f_2(\epsilon_{2,1}^{(z_k+1)}) + f_1(\tilde{d}_{2,2}^{(k)}) - v_1 + f_1(\tilde{d}_{1,1}^{(k)}) = 11.33 > 0$. Then, he publishes his privacy budget $\epsilon_{2,1}^{(2)} = 0.3$ with the corresponding noise distance $\hat{d}_{2,1}^{(2)}$ to the server. The server can also calculate the new effective noise distance $\tilde{d}_{2,1}^{(k+1)}$ and $UT_1^{(k+1)}$. It finds that $UT_i^{(k+1)}$ is positive, which means w_1 wins t_2. The server then alters the allocation table AL by setting the winner of t_2 as w_1 and the winner of t_1 as NULL.

In the $(k+2)$-th competition, it is w_2's turn to compete. w_2 can compete for both t_1 and t_3. He calculates $UT_2^{(k+2)}[t_1] = v_1 - f_1(\tilde{d}_{1,2}^{(k+2)}) - f_2(\epsilon_{1,2}^{(z_k+2)}) = 2.4 > 0$ and $UT_2^{(k+2)}[t_3] = -f_1(\tilde{d}_{3,2}^{(k+2)}) - f_2(\epsilon_{3,2}^{(z_k+2)}) + f_1(\tilde{d}_{3,3}^{(k+1)}) = -5.03 < 0$. After that, w_2 sets $UT_2^{(k+2)}$ as $UT_2^{(k+2)}[t_1]$, which is the maximal positive value in set $\{UT_2^{(k+2)}[t_1], UT_2^{(k+2)}[t_3]\}$. Then, w_2 applies to the server for t_1 by proposing $(\hat{d}_{2,1}^{(2)}, \epsilon_{2,1}^{(2)})$. After similar calculations, the server alters AL by setting the winner of t_1 as w_2.

In the $(k+3)$-th competition, it is w_3's turn to compete. w_3 can only apply for t_1. However, the value $UT_3^{(k+3)} = -3.95 < 0$. So w_3 does not compete for any tasks.

These three steps are repeated until all workers do not apply for any tasks (e.t. until the 6-th competition). Table V records the changing of effective distances and privacy budgets. The red one (with $UT > 0$) means there is a new winner who publishes a new privacy budget and updates the corresponding effective noise distance. The green one (with $UT \leq 0$) means the competitor fails to compete for the task and will publish neither his new noise distance nor his new privacy budget.

TABLE II: Task-worker distances.

Worker/Task	$t_1(8)$	$t_2(9)$	$t_3(7)$
$w_1(15)$	12.2	3.61	17.12
$w_2(15)$	5	10.44	12.21
$w_3(10)$	9.43	18.25	7.28

TABLE III: Effective distance and privacy budget.

Matchable pair	$(\tilde{d}, \epsilon^{(1)})$	$(\tilde{d}, \epsilon^{(2)})$	$(\tilde{d}, \epsilon^{(3)})$
(t_1, w_1)	(12.7,0.1)	(12.4,0.3)	(12.3,0.4)
(t_1, w_2)	(5.5,0.1)	(5.3,0.2)	(5.1,0.5)
(t_1, w_3)	(9.93,0.1)	(9.63,0.4)	(9.53,0.4)
(t_2, w_1)	(4.11,0.1)	(4.01,0.3)	(3.81,0.4)
(t_2, w_2)	(10.94,0.1)	(10.64,0.2)	(10.54,0.5)
(t_3, w_2)	(12.71,0.1)	(12.51,0.3)	(12.31,0.4)
(t_3, w_3)	(7.78,0.1)	(7.58,0.2)	(7.38,0.3)

TABLE IV: Allocation list from the k-th competition.

Task	k-th	$(k+1)$-th	$(k+2)$-th – $(k+6)$-th
t_1	w_1	NULL	w_2
t_2	w_2	w_1	w_1
t_3	w_3	w_3	w_3

TABLE V: The timeline of effective distances and privacy budgets.

<table>
<tr><th>Pair/Times</th><th>k</th><th>$k+1$</th><th>$k+2$</th><th>$k+3$</th><th>$k+4$</th><th>$k+5$</th><th>$k+6$</th></tr>
<tr><td>(t_1, w_1)</td><td colspan="4">(12.7,0.1)</td><td>(12.7,0.1)
(12.4,0.3)</td><td colspan="2">(12.7,0.1)</td></tr>
<tr><td>(t_1, w_2)</td><td colspan="2">(5.5,0.1)</td><td>(5.5,0.1)
(5.3,0.2)</td><td colspan="4">(5.3,0.2)</td></tr>
<tr><td>(t_1, w_3)</td><td colspan="3">(9.93,0.1)</td><td>(9.93,0.1)
(9.63,0.4)</td><td colspan="2">(9.93,0.1)</td><td>(9.93,0.1)
(9.63,0.4)</td></tr>
<tr><td>(t_2, w_1)</td><td>(4.11,0.1)</td><td>(4.11,0.1)
(4.01,0.3)</td><td colspan="5">(4.01,0.3)</td></tr>
<tr><td>(t_2, w_2)</td><td colspan="5">(10.94,0.1)</td><td>(10.94,0.1)
(10.64,0.2)</td><td>(10.94,0.1)</td></tr>
<tr><td>(t_3, w_2)</td><td colspan="2">(12.71,0.1)</td><td>(12.71,0.1)
(12.51,0.3)</td><td colspan="2">(12.71,0.1)</td><td>(12.71,0.1)
(12.51,0.3)</td><td>(12.71,0.1)</td></tr>
<tr><td>(t_3, w_3)</td><td colspan="7">(7.78,0.1)</td></tr>
</table>

V. Analysis

We analyze the performance of our proposed solutions in this section.

A. Running Time Cost

1) Time Cost of PUCE: As for the $\text{Cond}_{1,n}$, there is only one task and n workers. All workers hold Z privacy budget for this task. In the worst case, all workers will compete for this task with Z times. The server only needs to traverse the proposed set to get the winner with time cost $O(n \cdot Z)$. Therefore the worst time cost for $\text{Cond}_{1,n}$ is $O(n \cdot Z)$. In $\text{Cond}_{m,n}$, the number of task is m. Each of the n workers has Z privacy budget for each task. So the worst time cost for PUCE is $O(m \cdot n \cdot Z)$.

2) Convergence of PGT: In order to answer the convergence speed of PGT, we need to know how many rounds it takes to find a pure Nash equilibrium. For the corresponding potential game of a PAA-TA instance, $\mathcal{G} =< \mathcal{W}, \boldsymbol{S}, \boldsymbol{UT} >$, we assume there is an equivalent game with potential function $\Phi_{\mathbb{Z}}(\boldsymbol{st}) = d \cdot \Phi(\boldsymbol{st})$, where d is a positive multiplicative factor satisfying that $\Phi_{\mathbb{Z}}(\boldsymbol{st}) \in \mathbb{Z}$ for $\forall \boldsymbol{st} \in \boldsymbol{S}$. Let $\boldsymbol{st}^*$ be the best strategy the workers can choose in this PAA-TA game instance. Based on the above assumption, we prove that PGT executes at most $\Phi_{\mathbb{Z}}(\boldsymbol{st}^*)$ rounds.

Theorem V.1. *PGT executes at most $\Phi_{\mathbb{Z}}(\boldsymbol{st}^*)$ rounds to achieve a pure Nash equilibrium, where $\Phi_{\mathbb{Z}}(\boldsymbol{st}^*) = d \cdot \Phi(\boldsymbol{st}^*)$ is a scaled potential function with integer value d and $\boldsymbol{st}^*$ is the optimal strategy the workers can choose in the potential PAA-TA game instance.*

Proof. We say PGT converges when no workers deviate from their current strategies. If PGT has not converged, then at least one worker w_j deviates from his current strategy in each round. Besides the new change strategy st'_j of w_j is better than his current strategy st_j. And the change will improve at least 1 (i.e. $\Phi_{\mathbb{Z}}(st'_i, \boldsymbol{s}_{-i}) - \Phi_{\mathbb{Z}}(st_i, \boldsymbol{s}_{-i}) \geq 1$) for potential games. Because the maximum value of scaled potential function is $\Phi_{\mathbb{Z}}(\boldsymbol{st}^*)$, and the total utility is always positive, PGT needs at most $\Phi_{\mathbb{Z}}(\boldsymbol{st}^*)$ rounds to converge to a pure Nash equilibrium. □

B. Utility and Distance Analysis

Observe from the utility function, we know that the total utility U is linear to task value v. When the workers' serving range increases, he can compute more tasks. However, a large distance leads to a small utility, limiting the impact of serving ranges on the utility.

1) PPCF and PCF analysis: We say that PPCF is better than PCF in comparison. We give the relevant theorem in Theorem V.2 which is proved in Appendix VIII-A.

Theorem V.2. *For any given distance $d_x, d_y, \epsilon_x, \epsilon_y$ satisfying $d_x < d_y$. Let $\eta_x \sim Lap(0, 1/\epsilon_x), \eta_y \sim Lap(0, 1/\epsilon_y)$. Let $\hat{d}_x = d_x + \eta_x, \hat{d}_y = d_y + \eta_y$. Then $Pr[PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) > \frac{1}{2}] \leq Pr[PPCF(d_x, \hat{d}_y, \epsilon_y) > \frac{1}{2}]$.*

2) Quality of PGT: Since the distance in our game is rather real distance than effective noise distance, here we give the upper bound of expectation of *price of stability* (EPoS) and the lower bound of expectation of *price of anarchy* (EPoA).

Let $U^L_j(i) = v_i - f_1(d_{i,j}) - f_2(\sum_{t_k \in R_j} sum(\boldsymbol{\epsilon_{k,j}}))$ and $U^H_j(i) = v_i - f_1(d_{i,j}) - f_2(min(\boldsymbol{\epsilon_{i,j}}))$. Let

$$U^+_{min}(i) = \begin{cases} \min\limits_{R_j \ni t_i, U^L_j(i) > 0} U^L_j(i), & \text{if there exists } U^L_j(i) > 0 \\ 0, & \text{otherwise} \end{cases}$$

and

$$U^+_{max}(i) = \begin{cases} \max\limits_{R_j \ni t_i} U^H_j(i), & \text{if there exists } U^H_j(i) > 0 \\ 0, & \text{otherwise} \end{cases}$$

then we have Theorem V.3 as follows.

Theorem V.3. *In the strategic game of PGT, the lower bound of EPoA is $\frac{\sum_{t_i \in \mathcal{T}} U^+_{min}(i)}{\sum_{t_i \in \mathcal{T}} U^+_{max}(i)}$ $(\sum_{t_i \in \mathcal{T}} U^+_{max}(i) \neq 0)$ and the upper bound of EPoS is 1.*

Please refer to details of the proof of Theorem V.3 in Appendix VIII-B in our technical report [**?**].

C. Privacy Analysis

We define the query data set of worker w_j as X_j, which consists of all tasks in the serving range of w_j (i.e., R_j). The neighboring data set of X_j is noted as X'_j. It satisfies that $\|X_j - X'_j\| = 1$, which means there is only one different task item between X_j and X'_j. We focus on the query f as '*Get each distance from w_j to his service tasks R_j*'. That means $f(X_j) = [d_{i_1,j}, ..., d_{i_{|R_j|},j}]$.

Theorem V.4. *PUCE and PGT satisfy $(\sum_{t_i \in R_j} \boldsymbol{b_{i,j} \epsilon_{i,j}} r_j)$-differential privacy for each worker w_j.*

Proof. Let $\mathcal{A}_j$ be the mechanism PUCE or PGT applying to w_j with query f defined above. For query $f(X_j) = [d_{i_1,j}, ..., d_{i_{|R_j|},j}]$, we extend it to an equivalent query $f'(X_j) = f(X_j) \cdot \mathcal{J}$, where

$$\mathcal{J} = \begin{bmatrix} CP(\boldsymbol{b_{i_1,j}}) & & & \\ & CP(\boldsymbol{b_{i_2,j}}) & & \\ & & \ddots & \\ & & & CP(\boldsymbol{b_{i_{|R_j|},j}}) \end{bmatrix}$$

is a block diagonal matrix. Actually, $f'(X_j)$ means query $d_{i_u,j}$ for $sum(\boldsymbol{b_{i_u,j}})$ times for $u \in [|R_j|]$. We denote the size of $f'(X_j)$ as $|f'|$ and the a-th element of $f'(X_j)$ as $f'(X_j)_a$.

Let Y_j denote the set of all published noise distances of the worker w_j to tasks in R_j. Then we have $Y_j = f'(X_j) + [\eta_1, \eta_2, ..., \eta_{|f'|}]$, where $\eta_a (1 \leq a \leq |f'|)$ is an i.i.d random variable drawn from $Lap(1/\epsilon_a)$. Hence we have

$$\begin{aligned} \frac{\Pr[\mathcal{A}_j(X_j) = Y_j]}{\Pr[\mathcal{A}_j(X'_j) = Y_j]} &= \prod_{a \in [|f'|]} \left(\frac{\exp(-\epsilon_a |Y_{j,a} - f'(X_j)_a|)}{\exp(-\epsilon_a |Y_{j,a} - f'(X'_j)_a|)}\right) \\ &= \prod_{t_i \in R_j} \prod_{u \in [sum(\boldsymbol{b_{i,j}})]} \left(\frac{\exp(-\epsilon^{(u)}_{i,j} |\tilde{d}^{(u)}_{i,j} - d_{i,j}|)}{\exp(-\epsilon^{(u)}_{i,j} |\tilde{d}^{(u)}_{i,j} - d'_{i,j}|)}\right) \\ &\leq \prod_{t_i \in R_j} \prod_{u \in [sum(\boldsymbol{b_{i,j}})]} (\exp(\epsilon^{(u)}_{i,j} (|d_{i,j} - d'_{i,j}|))) \\ &= \prod_{t_i \in R_j} \exp(\boldsymbol{b_{i,j} \epsilon_{i,j}} (|d_{i,j} - d'_{i,j}|)) \\ &\leq \exp(\sum_{t_i \in R_j} \boldsymbol{b_{i,j} \epsilon_{i,j}} r_j). \end{aligned}$$

Then PUCE and PGT satisfy $(\sum_{t_i \in R_j} \boldsymbol{b_{i,j} \epsilon_{i,j}} r_j)$-differential privacy for each worker w_j. □

VI. Experiment

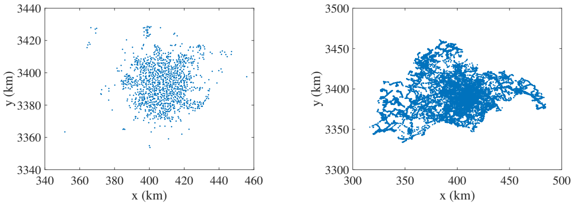
We design experiments to compare our PUCE method and PGT method with non privacy greedy solution. Besides, we also compare our utility CEA with Wang et al. 's distance CEA [2]. In order to verify that PPCF is better than PCF, we compare our solution result between with and without PPCF.

A. Data Sets

We test our mechanisms in synthetic and real data sets.

1) Synthetic Data Set: We generate two data sets with 2-dimensional uniform distribution and normal distribution, respectively. For the uniform distribution data set, we randomly generate 300000 points for tasks and 900000 for workers in a plane with a range of 100×100. Each point follows a 2-dimensional uniform distribution with an average of 0.

For the normal distribution data set, we generate 300k and 900k points for tasks and workers, respectively. The expectation and variance for all points are 0 and 150, respectively.



(a) Order location distribution (b) Taxi location distribution

Fig. 5: orders and taxies of Chengdu Didi data set.

TABLE VI: Experimental settings.

Parameters	Values
worker-task ratio	1, 1.5, **2**, 2.5, 3
task values	1.5, 3, **4.5**, 6, 7.5
worker range	0.8, 1.1, **1.4**, 1.7, 2.0
privacy budget	[0.5, 0.75], [0.75, 1.00], [1.00, 1.25], [1.25, 1.50], [1.50, 1.75]; **[0.5,1.75]**
privacy budget group size	**7**

2) Real Data Set: We use Didi Chuxing[14] in Chengdu, China, as our real data set. We choose the day with the most requests for evaluation (November 18, 2016 in Chengdu) and perform the same preprocessing as in Ref [15] denoted as *chengdu*.

Chengdu contains 259347 orders and 30000 taxis. Each order tuple is a taxi request consisting of a release time, a pickup location, a drop-off location, and some passengers. Each taxi tuple is a basic message consisting of the original location of the taxi and its capacity. The location distribution of the orders and taxis are shown in Figure 5.

B. Experimental Setup

We split the orders into batches by timestamp. Each batch contains at most 1000 orders. We also split the taxis into ten groups for the real data set, each containing 3000 taxis. We use each worker group circularly for each batch. We set the start locations of orders as task locations and the start locations of taxis as worker locations.

Let S_T and S_W be two sets for tasks and workers. We define the value $p_{wt} = \frac{|S_W|}{|S_T|}$ as *worker-task ratio*, which stands for the ratio between worker number, and task number.

We test three dependent variables: time cost, utility and distance. We set the independent variables as the worker-task ratio, task value, worker range, and privacy budget. We list our parameter settings in table VI, where the default values are marked in bold.

As for distance value function f_1 and privacy budget value function f_2, we model them as linear functions and use $f_1(x) = \alpha x$ and $f_2(x) = \beta x$ in our experiment. We set $\alpha = 1$ and $\beta = 1$.

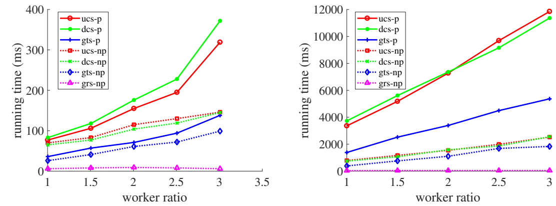
We run our experiment on an Intel(R) Xeon(R) Silver 4210R CPU @ 2.4GHz with 128 GB RAM in Java.

C. Empirical Measures

We design a utility-based empirical measure of the efficiency of our proposed mechanisms.

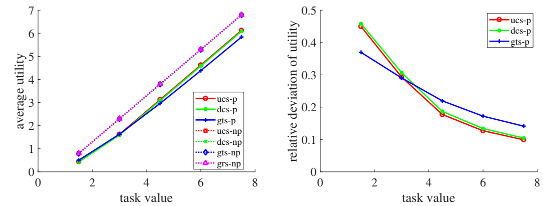
Average Utility. We define the average utility U_{AVG} as $\frac{\sum_{(i,j)\in M} U_j(i)}{|M|}$. It means the average utility value of a successful task-worker pair.

Relative Deviation of Utility. Let the utility of non-privacy solutions be U_{NP} and privacy ones be U_{P}. We define the relative deviation of utility U_{RD} as $\frac{U_{\text{NP}}-U_{\text{P}}}{U_{\text{NP}}}$, which means the utility deviation of privacy solutions from non-privacy ones.



(a) *chengdu* (b) *normal*

Fig. 6: The impact of the worker ratio on the time cost.



(a) Average Utility (b) Relative Deviation of Utility

Fig. 7: The impact of the task value on the utility (*chengdu*).

Average Travel Distance. We define the average travel distance D_{AVG} as $\frac{\sum_{(i,j)\in M} d_{i,j}}{|M|}$. It means the average travel distance of a successful task-worker pair.

Relative Deviation of Distance. Let the distance of non-privacy solutions be D_{NP} and privacy ones be D_{NP}. We define the relative deviation of distance D_{RD} as $\frac{D_{\text{P}}-D_{\text{NP}}}{D_{\text{NP}}}$, which means the distance deviation of privacy solutions from non-privacy ones.

D. Experiment Result

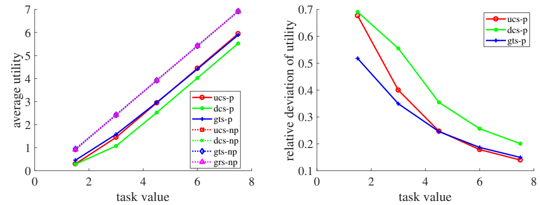
We show the experimental results on time cost, utility, and real travel distance. We construct the non-privacy solution of each privacy solution by eliminating the privacy budget cost in the utility function and replacing noise distance with real distance. We compare our PUCE (ucs-p) and PGT (gts-p) with CEA (dcs-p) and the relevant non-privacy solutions (ucs-np, gts-np, dcs-np) as well as the greedy solution (grs-np).

1) Time Cost: Figure 6 shows the time cost on different worker ratio from 1 to 3 while the other parameters are in the default values in Table VI. We can see that the time cost increases linearly with the worker ratio. That is because when we fix the task quantity, as the worker ratio becomes larger, the competition between workers will become more fierce, and it will cost more time to finish the whole competition.

Besides, we can find that ucs-p costs nearly the same time over the change of worker ratio. gts-p costs much less time than ucs-p and dcs-p. Compared with dcs-p, gts-p costs about 52%–63% less time in *chengdu* and 50%–63% in *normal*.

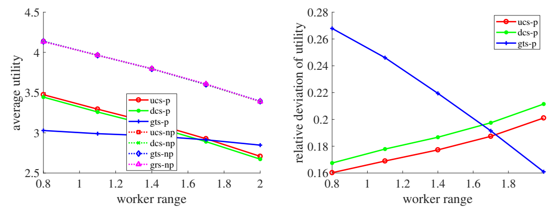
2) Average Utility: Figure 7 and 8 show the relation between the utility and the task value on *chengdu* and *normal* respectively. We change the task value from 1.5 to 7.5 and set other parameters as the default values.

In Figure 7(a) and 8(a), the utility increases approximately linear with the task value. We can see that gts-p performs worse than dcs-p slightly on *chengdu*, but better on *normal*.



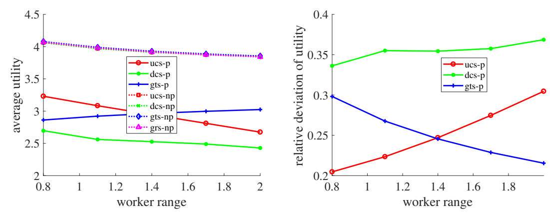
(a) Average Utility (b) Relative Deviation of Utility

Fig. 8: The impact of the task value on the utility (*normal*).



(a) Average Utility (b) Relative Deviation of Utility

Fig. 9: The impact of the worker range on the utility (*chengdu*).



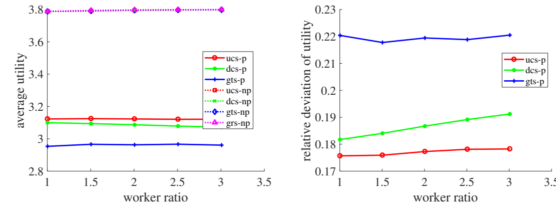
(a) Average Utility (b) Relative Deviation of Utility

Fig. 10: The impact of the worker range on the utility (*normal*).

ucs-p performs better than dcs-p in both of the two data sets. The relative deviation of utility impacted by the task value is shown in Figure 7(b) and 8(b). We can see that the relative deviation of utility decreases with the task value increase from 1.5 to 7.5, which means the absolute deviation between the private and non-private solutions keeps nearly stable. And when the task value becomes larger and larger, the utility of private solutions equals that of non-private solutions asymptotically.

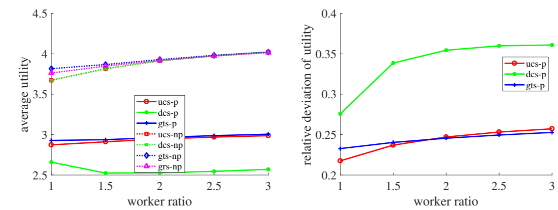
Figure 9 and 10 show the relation between the utility and the worker range on *chengdu* and *normal* respectively. The worker serving range (denoted as worker range) increases from 0.8 to 2, and the other parameters are set as default values. The average utility depends on the total utility and the matching quantity. Specifically, in Figure 9(a), the average utility of all solutions decreases when the worker range increases from 0.8 to 2. It is because when worker serving ranges become larger, more workers (who have no task to apply for in some small range conditions, denoting them as $\mathcal{W}_L$) will be able to apply for some tasks. With the ratio of $\mathcal{W}_L$ becoming larger, the average distance to all matching tasks becomes larger, making the average utility smaller.

Besides, we can see that the utility of gts-p decreases slower than both ucs-p and dcs-p. The utility of gts-p is no less 88% when the worker range is no more than 1.6. And as the worker range increases, the utility of gts-p will exceed the other two. The reason why gts-p keeps lower decrease is



(a) Average Utility (b) Relative Deviation of Utility

Fig. 11: The impact of the worker ratio on the utility (*chengdu*).



(a) Average Utility (b) Relative Deviation of Utility

Fig. 12: The impact of the worker ratio on the utility (*normal*).

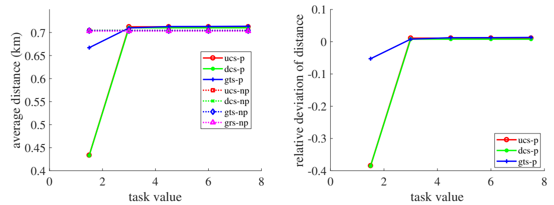
that gts-p can avoid ineffective competition. When the serving range becomes larger, the competition becomes more intense, and the advantage of gts-p becomes more apparent.

Figure 9(b) shows the relative deviation of utility infected by the worker range. We can see that the utility of gts-p will tend to that of its non-privacy solution when the worker range becomes larger and larger. However ucs-p and dcs-p deviate more as the worker range becomes larger. That is because when the worker's serving range becomes larger, it has a greater possibility of disturbing a real large distance to a small noise distance, or a small real distance to a large distance. Without the guarantee of total utility function ST, the total applied workers' utilities in ucs-p and dcs-p both decrease dramatically when the worker range increases.

From Figure 10(a), we can get the similar conclusion to that in Figure 9(a). Besides, we can find when the worker range becomes large enough, the decline rate of average utility for ucs-p and dcs-p tend to be small. That is because being too far away will make the utility value non-positive, and the server will not choose. The average utility of gts-p increases slightly, which is 16% larger than dcs-p on average. That is because gts-p can increase the total utility more rapidly than the matching quantity.

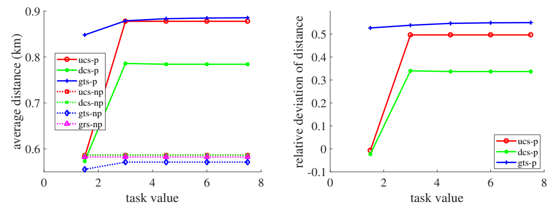
Figure 11 and 12 show the relation between the utility and the worker ratio. We set the worker range and task value as default values in Table VI. From figure 11(a) and 12(a), we can see that the worker ratio does not affect the average utility very much. That is because the increase of workers does not significantly increase applied workers. Besides, we can see that ucs-p always keeps a higher average utility than dcs-p. And gts-p performs worse than dcs-p in *chengdu* but better in *normal*.

3) Average Travel Distance: Figure 13 to Figure 18 show the influence of the task value, worker range and worker ratio on the distance. dcs-p is better than ucs-p and gts-p in most cases. Besides, we can see that different data sets lead to different comparison results for ucs-p, gts-p and dcs-p. The



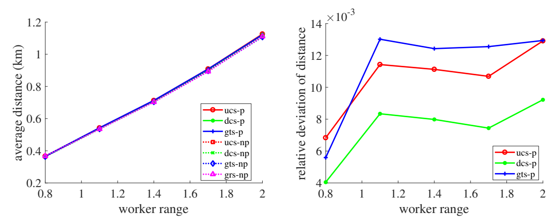
(a) Average Distance (b) Relative Deviation of Distance

Fig. 13: The impact of the task value on the distance (*chengdu*).



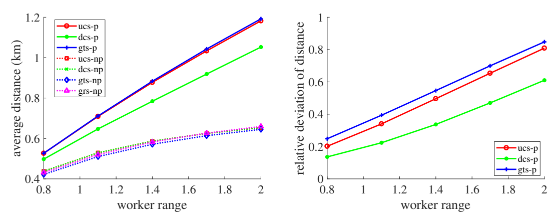
(a) Average Distance (b) Relative Deviation of Distance

Fig. 14: The impact of the task value on the distance (*normal*).



(a) Average Distance (b) Relative Deviation of Distance

Fig. 15: The impact of the worker ratio on the distance (*chengdu*).



(a) Average Distance (b) Relative Deviation of Distance

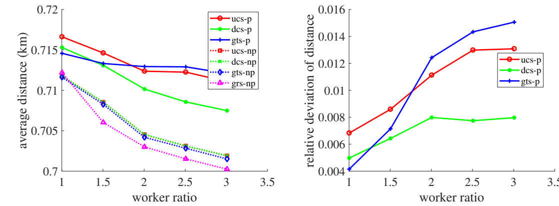
Fig. 16: The impact of the worker ratio on the distance (*normal*).

average travel distance of dcs-p on *normal* outperforms the other two on *chengdu*.

Figure 13 and 14 show the relation between the average distance and the task values. We can see that task values do not affect the average distance when the task value is larger than 3. That is because when the task value is large enough, it will not affect the difference between the two utility values. Workers will not choose many tasks in their range when the task value is minimal, leading to a small average distance.

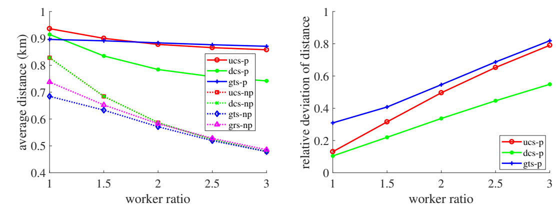
Figure 15 and 16 show the relation between the average distance and the worker serving range. We can see that the average distance increases when worker range increases. That is because a larger range will lead to more applied workers with far distance, making the average distance larger.

Figure 17 and 18 show the relation between the average distance and the worker ratio. Especially in figure 17(a), the average distance in non-privacy solutions decreases when the worker ratio increases. That is because, with the increase in



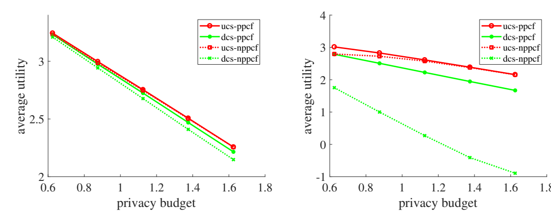
(a) Average Distance (b) Relative Deviation of Distance

Fig. 17: The impact of the worker ratio on the distance (*chengdu*).



(a) Average Distance (b) Relative Deviation of Distance

Fig. 18: The impact of the worker ratio on the distance (*normal*).



(a) Average Distance (b) Average Distance

Fig. 19: The impact of privacy on the utility.

workers, the competition has become rigorous. The number of tasks limits the increase of workers' application, and a task will be allocated to the worker at a small distance. Therefore, the average becomes smaller with the worker ratio becoming larger. As for privacy solutions, competition will also cost more privacy budget on utility value, which will relieve the reduction in privacy solutions.

4) PPCF and Non-PPCF: We compare our PUCE (ucs-ppcf) and distance elimination conflict solution (dcs-ppcf) with non-PPCF ones. We fix the task value as 4.5, the worker range distance as 1.4, and the worker ratio as 2. We divide the privacy budget range into 5 groups shown in Table VI.

Figure 19 shows the relation between the average utility and the privacy budget. We mark the median of each interval as the value of the x-axis.

The solutions with PPCF are better than that without PPCF when the privacy budget is small. It means PPCF is suitable for high privacy situations and continuously more effective than without PPCF. Especially in figure 19(a), as the privacy budget increases, the average utility decreases, and the difference between PPCF and non-PPCF is eliminated. That is because the larger the privacy budget, the more accurate the noise distance, and the smaller difference between PPCF and PCF.

VII. Conclusion

In this paper, we formalize Privacy-aware Task Assignment (PA-TA) Problem, which assigns a task to a worker to get a high utility value. In order to make use of noise distance published by workers, we propose new notations called *effective noise distance* and *effective privacy budget*. To get higher utility value, we offer a new comparison function called PPCF and prove that it can achieve better effectiveness than PCF in both theory and practice. Besides, we propose another game theoretic approach to solve the problem. Extensive experiments have been conducted to show the efficiency and effectiveness of our methods on both real and synthetic data sets.

Our PUCE and PGT only consider the distance privacy of one worker in his serving range. If the serving range of a worker is small enough and the quantity of tasks in this range is large enough, attackers can locate the worker's position through trilateration by viewing the entire range as a position. That is because too much effective noise distance from a worker to many tasks will outline the worker's serving range. Our subsequent work will focus on this problem and consider how to hide correlation privacy caused by the relation between different worker serving ranges.

REFERENCES

[1] C. Dwork, "Differential privacy," in *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II* (M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, eds.), vol. 4052 of *Lecture Notes in Computer Science*, pp. 1–12, Springer, 2006.

[2] Z. Wang, J. Hu, R. Lv, J. Wei, Q. Wang, D. Yang, and H. Qi, "Personalized privacy-preserving task allocation for mobile crowdsensing," *IEEE Trans. Mob. Comput.*, vol. 18, no. 6, pp. 1330–1341, 2019.

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

[4] J. S. Kim, Y. D. Chung, and J. W. Kim, "Differentially private and skew-aware spatial decompositions for mobile crowdsensing," *Sensors*, vol. 18, no. 11, p. 3696, 2018.

[5] L. Wang, D. Zhang, D. Yang, B. Y. Lim, and X. Ma, "Differential location privacy for sparse mobile crowdsensing," in *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain* (F. Bonchi, J. Domingo-Ferrer, R. Baeza-Yates, Z. Zhou, and X. Wu, eds.), pp. 1257–1262, IEEE Computer Society, 2016.

[6] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013* (A. Sadeghi, V. D. Gligor, and M. Yung, eds.), pp. 901–914, ACM, 2013.

[7] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Optimal geo-indistinguishable mechanisms for location privacy," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014* (G. Ahn, M. Yung, and N. Li, eds.), pp. 251–262, ACM, 2014.

[8] H. To, C. Shahabi, and L. Xiong, "Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server," in *34th IEEE International Conference on Data Engineering, ICDE 2018, Paris, France, April 16-19, 2018*, pp. 833–844, IEEE Computer Society, 2018.

[9] X. Jin and Y. Zhang, "Privacy-preserving crowdsourced spectrum sensing," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1236–1249, 2018.

[10] A. Ghosh and A. Roth, "Selling privacy at auction," in *Proceedings 12th ACM Conference on Electronic Commerce (EC-2011), San Jose, CA, USA, June 5-9, 2011* (Y. Shoham, Y. Chen, and T. Roughgarden, eds.), pp. 199–208, ACM, 2011.

[11] K. Nissim, C. Orlandi, and R. Smorodinsky, "Privacy-aware mechanism design," in *Proceedings of the 13th ACM Conference on Electronic Commerce, EC 2012, Valencia, Spain, June 4-8, 2012* (B. Faltings, K. Leyton-Brown, and P. Ipeirotis, eds.), pp. 774–789, ACM, 2012.

[12] D. Xiao, "Is privacy compatible with truthfulness?," in *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013* (R. D. Kleinberg, ed.), pp. 67–86, ACM, 2013.

[13] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.

[14] Didi Chuxing. http://www.didichuxing.com/.

[15] Y. Tong, Y. Zeng, Z. Zhou, L. Chen, J. Ye, and K. Xu, "A unified approach to route planning for shared mobility," *Proc. VLDB Endow.*, vol. 11, no. 11, pp. 1633–1646, 2018.

VIII. APPENDIX

A. Proof for Theorem V.2

Before the proof of Theorem V.2, we declare and prove Lemma VIII.1 and Lemma VIII.2 as follows.

Lemma VIII.1. *For any* $d_x, d_y, \epsilon_x, \epsilon_y$, $\hat{d}_x = d_x + Lap(0, 1/\epsilon_x)$, $\hat{d}_y = d_y + Lap(0, 1/\epsilon_y)$, *we have* $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) > \frac{1}{2} \Leftrightarrow \hat{d}_x < \hat{d}_y$.

Proof. Let $\eta_x \sim \epsilon_x, \eta_y \sim \epsilon_y$. Then we have

$$PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) = \iint_D f(\eta_x, \eta_y)$$

where $f(\eta_x, \eta_y) = \frac{\epsilon_x \epsilon_y}{4} e^{-\epsilon_x|\eta_x| - \epsilon_y|\eta_y|}$ and D is the plane set satisfying $D = \{(\eta_x, \eta_y) : \eta_y - \eta_x < \hat{d}_y - \hat{d}_x\}$. Notice that $f(\eta_x, \eta_y)$ is symmetry about both x-axis and y-axis and D is part of plane split by line $l_\eta : \eta_y = \eta_x + \hat{d}_y - \hat{d}_x$. So we know that only when l_η crosses the origin ($\hat{d}_y = \hat{d}_x$), $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y)$ equals $\frac{1}{2}$. When $\hat{d}_y - \hat{d}_x > 0$, $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) < \frac{1}{2}$, and $\hat{d}_y - \hat{d}_x > 0$, $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) > \frac{1}{2}$. Therefore, $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) > \frac{1}{2} \Leftrightarrow \hat{d}_x < \hat{d}_y$. □

Lemma VIII.2. *For any two continue and differentiable non-negative functions f, g defined in $\mathbb{R}$, if there exists an interval $[a, +\infty)$ satisfying that $\int_a^{+\infty} f(x)dx = \int_a^{+\infty} g(x)dx$ and there exists a point $x_0 \in (a, +\infty)$ satisfying $f(x) \geq g(x)$ for $x \in (a, x_0]$ and $f(x) \leq g(x)$ for $x \in (x_0, +\infty)$, then $\int_a^x f(x)dx \geq \int_a^x g(x)dx$ for all $x \in [a, +\infty)$.*

Proof. For any $x \in [a, +\infty)$, we can divide it into two cases: (1) $x \in [a, x_0]$; (2) $x \in (x_0, +\infty]$. If (1) holds, according to $f(x) \geq g(x)$ *for* $x \in (a, x_0]$, we have directly get

$$\int_a^x f(x)dx \geq \int_a^x g(x)dx \quad \text{for } x \in [a, x_0]. \tag{5}$$

If (2) holds, then we have $\int_{x_0}^{+\infty} f(x)dx \leq \int_{x_0}^{+\infty} g(x)dx$. And we can get

$$\begin{aligned}
&\int_a^x f(x)dx - \int_a^x g(x)dx \\
&= \int_a^{+\infty} f(x)dx - \int_{x_0}^{+\infty} f(x)dx - (\int_a^{+\infty} g(x)dx - \int_{x_0}^{+\infty} g(x)dx) \\
&= \int_{x_0}^{+\infty} g(x)dx - \int_{x_0}^{+\infty} f(x)dx \geq 0.
\end{aligned}$$

So we have

$$\int_a^x f(x)dx \geq \int_a^x g(x)dx \quad \text{for } x \in (x_0, +\infty). \tag{6}$$

From Equation 5 and 6, we can have $\int_a^x f(x)dx \geq \int_a^x g(x)dx$ for $x \in [a, +\infty)$. □

Based on Lemma VIII.1 and Lemma VIII.2, we give the proof of Theorem V.2 as follows.

Proof. From Lemma VIII.1, we have $PCF(\hat{d}_x, \hat{d}_y, \epsilon_x, \epsilon_y) > \frac{1}{2} \Leftrightarrow \hat{d}_x < \hat{d}_y$. From Equation 2, we have $\Pr[d_x < d_y] > \frac{1}{2} \Leftrightarrow d_x < \hat{d}_y$. So we only need to prove $Pr[\hat{d}_x < \hat{d}_y] \leq Pr[d_x < \hat{d}_y]$ for any d_x, d_y satisfying $d_x < d_y$.

According to the definition, we have

$$\begin{aligned}\Pr[\hat{d}_x < \hat{d}_y] &= \Pr[d_x + \eta_x < d_y + \eta_y] = \Pr[\eta_y > \eta_x + d_x - d_y] \\ &= \int_{-\infty}^{+\infty} \left(\int_{-\infty}^{\eta_y - d_x + d_y} \frac{\epsilon_x \epsilon_y}{4} e^{-(\epsilon_x |\eta_x| + \epsilon_y |\eta_y|)} d\eta_x \right) d\eta_y\end{aligned}$$

and

$$\begin{aligned}\Pr[d_x < \hat{d}_y] &= \Pr[d_x < d_y + \eta_y] = \Pr[\eta_y > d_x - d_y] \\ &= \int_{d_x - d_y}^{+\infty} \frac{\epsilon_y}{2} e^{-\epsilon_y |\eta_y|} d\eta_y.\end{aligned}$$

Let $s = d_y - d_x$. Let $F : s \to \Pr[\hat{d}_x < \hat{d}_y]$ and $G : s \to \Pr[d_x < \hat{d}_y]$. From the definition, we know $s > 0$, $\lim\limits_{s \to 0} F(s) = \lim\limits_{s \to 0} G(s) = \frac{1}{2}$ and $\lim\limits_{s \to +\infty} F(s) = \lim\limits_{s \to +\infty} G(s) = 1$. And we have

$$\begin{aligned}\frac{\partial F(s)}{\partial s} &= \frac{\epsilon_x \epsilon_y}{4} \left(\frac{e^{-s\epsilon_x} + e^{-s\epsilon_y}}{\epsilon_x + \epsilon_y} - \frac{e^{-s\epsilon_x} - e^{-s\epsilon_y}}{\epsilon_x - \epsilon_y} \right) \\ &= \frac{\epsilon_x \epsilon_y}{2} \cdot \frac{e^{-s\epsilon_y} \epsilon_x - e^{-s\epsilon_x} \epsilon_y}{(\epsilon_x + \epsilon_y)(\epsilon_x - \epsilon_y)} > 0, \\ \frac{\partial G(s)}{\partial s} &= \frac{\epsilon_y}{2} e^{-s\epsilon_y} > 0, \\ \frac{\partial F(s)}{\partial s} / \frac{\partial G(s)}{\partial s} &= \frac{\epsilon_x (\epsilon_x - e^{s(\epsilon_y - \epsilon_x)} \epsilon_y)}{(\epsilon_x + \epsilon_y)(\epsilon_x - \epsilon_y)}.\end{aligned}$$

Let $\frac{\partial F(s)}{\partial s} / \frac{\partial G(s)}{\partial s} \leq 1$. Then we have $s \leq \frac{1}{\epsilon_x - \epsilon_y} \ln \frac{\epsilon_x}{\epsilon_y}$. Let $\frac{\partial F(s)}{\partial s} / \frac{\partial G(s)}{\partial s} \geq 1$. Then we have $s \geq \frac{1}{\epsilon_x - \epsilon_y} \ln \frac{\epsilon_x}{\epsilon_y}$. That is to say $\frac{\partial G(s)}{\partial s} \geq \frac{\partial F(s)}{\partial s}$ for $s \in (0, \frac{1}{\epsilon_x - \epsilon_y} \ln \frac{\epsilon_x}{\epsilon_y})$ and $\frac{\partial G(s)}{\partial s} \leq \frac{\partial F(s)}{\partial s}$ for $s \in (\frac{1}{\epsilon_x - \epsilon_y} \ln \frac{\epsilon_x}{\epsilon_y}, +\infty)$. According to Lemma VIII.2, we have $F(s) \leq G(s)$ for $s \in (0, +\infty)$. □

B. Proof for Theorem V.3

Proof. Let $\hat{U}(\boldsymbol{st})$ be the overall utility of the strategy $\boldsymbol{st}$ with (i.e. $\hat{U}(\boldsymbol{st}) = \Phi(\boldsymbol{st})$). Besides, we note the global optimal strategy as $\hat{\boldsymbol{st}}$, the strategy of achieving best competing utility value as $\boldsymbol{st}^*$ and the worst competing utility value as $\boldsymbol{st}^\sharp$. Then we have $\hat{U}(\hat{\boldsymbol{st}}) = \Phi(\hat{\boldsymbol{st}})$, $\hat{U}(\boldsymbol{st}^*) = \Phi(\boldsymbol{st}^*)$ and $\hat{U}(\boldsymbol{st}^\sharp) = \Phi(\boldsymbol{st}^\sharp)$. Thus,

$$EPoS = \frac{E(\hat{U}(\boldsymbol{st}^*))}{E(OPT)} = \frac{E(\hat{U}(\boldsymbol{st}^*))}{E(\hat{U}(\hat{\boldsymbol{st}}))} \leq 1.$$

Notice that if we get the lower bound of $E(\hat{U}(\boldsymbol{st}^\sharp))$ and upper bound of $E(\hat{U}(\hat{\boldsymbol{st}}))$, then we can get the value of EPoA. As for $E(\hat{U}(\boldsymbol{st}^\sharp))$, we have

$$\begin{aligned}E(\hat{U}(\boldsymbol{st}^\sharp)) &\geq \min_k \sum_{t_i \in \mathcal{T}} \sum_{w_j \in \mathcal{W}} (s_{i,j}^{(k)} \cdot (v_i - f_1(d_{i,j})) - f_2(\boldsymbol{b_{i,j}^{(k)}} \cdot \boldsymbol{\epsilon_{i,j}})) \\ &\geq \sum_{t_i \in \mathcal{T}} \min_{R_j \ni t_i, U_j^L(i) > 0} U_j^L(i) = \sum_{t_i \in \mathcal{T}} U_{min}^+(i)\end{aligned}$$

As for $E(\hat{U}(\hat{\boldsymbol{st}}))$, we have

$$\begin{aligned}E(\hat{U}(\hat{\boldsymbol{st}})) &\leq OPT(\sum_{t_i \in \mathcal{T}} \sum_{w_j \in \mathcal{W}} (s_{i,j} \cdot (v_i - f_1(\tilde{d}_{i,j})) - f_2(\boldsymbol{b_{i,j}} \cdot \boldsymbol{\epsilon_{i,j}}))) \\ &\leq \sum_{t_i \in \mathcal{T}} \max_{R_j \ni t_i} U_j^H(i) = \sum_{t_i \in \mathcal{T}} U_{max}^+(i)\end{aligned}$$

So we have

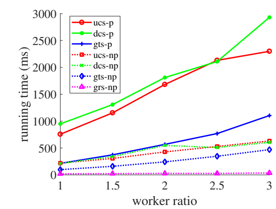
$$EPoA = \frac{E(\hat{U}(\boldsymbol{st}^*))}{E(OPT)} \geq \frac{\sum_{t_i \in \mathcal{T}} U_{min}^+(i)}{\sum_{t_i \in \mathcal{T}} U_{max}^+(i)}$$

□

C. Experiment Result for the Uniform Data set

The experiment result of the uniform data set is shown in this section.

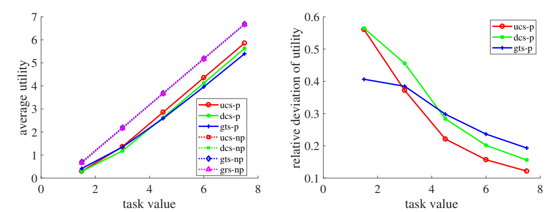
The time cost is shown in Figure 20.



(a) *uniform*

Fig. 20: The impact of the worker ratio on the time cost.

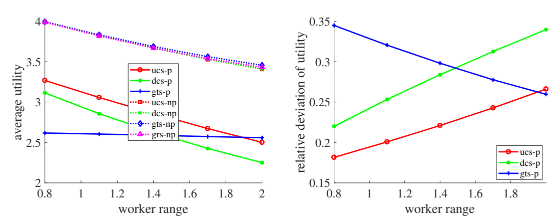
The impact of task value on utility is shown in Figure 21.



(a) Average Utility (b) Relative Deviation of Utility

Fig. 21: The impact of the task value on the utility for *uniform*.

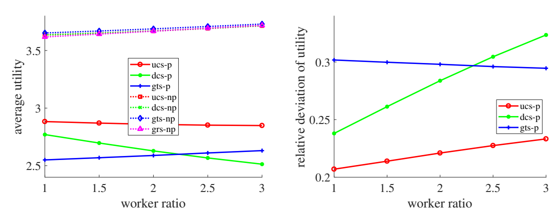
The impact of worker range on utility is shown in Figure 22.



(a) Average Utility (b) Relative Deviation of Utility

Fig. 22: The impact of the worker range on the utility for *uniform*.

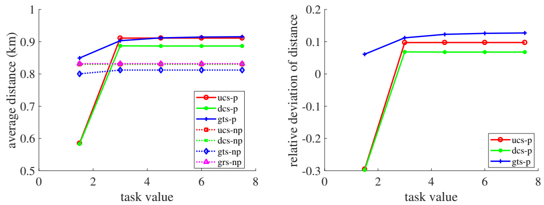
The impact of worker ratio on utility is shown in Figure 23.



(a) Average Utility (b) Relative Deviation of Utility

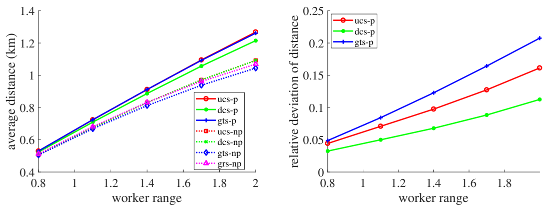
Fig. 23: The impact of the worker ratio on the utility for *uniform*.

The impact of task value on travel distance is shown in Figure 24.



(a) Average Distance (b) Relative Deviation of Distance

Fig. 24: The impact of the task value on the distance for *uniform*.

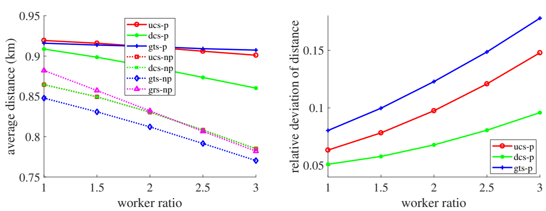


(a) Average Distance (b) Relative Deviation of Distance

Fig. 25: The impact of the worker ratio on the distance for *uniform*.

The impact of worker range on travel distance is shown in Figure 25.

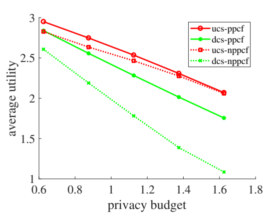
The impact of worker ratio on travel distance is shown in Figure 26.



(a) Average Distance (b) Relative Deviation of Distance

Fig. 26: The impact of the worker ratio on the distance for *uniform*.

The impact of worker ratio on PPCF and non-PPCF is shown in Figure 27.



(a) Average Distance

Fig. 27: The impact of privacy on the utility.