Time-Efficient Geo-Obfuscation to Protect Worker Location Privacy over Road Networks in Spatial Crowdsourcing

Chenxi Qiu
Department of Computer Science
Rowan University
qiu@rowan.edu

Anna Squicciarini
College of Information
Science and Technology
Pennsylvania State University
acs20@psu.edu

Zhuozhao Li
Globus Lab
University of Chicago
zhuozhao@uchicago.edu

Ce Pang
Department of Computer Science
Rowan University
pangc7@students.rowan.edu

Li Yan
Senseable City Lab
Massachusetts Institute of Technology
liyan_20@mit.edu

ABSTRACT

To promote cost-effective task assignment in *Spatial Crowdsourcing (SC)*, workers are required to report their location to servers, which raises serious privacy concerns. As a solution, *geo-obfuscation* has been widely used to protect the location privacy of SC workers, where workers are allowed to report perturbed location instead of the true location. Yet, most existing geo-obfuscation methods consider workers' mobility on a 2 dimensional (2D) plane, wherein workers can move in arbitrary directions. Unfortunately, 2D-based geo-obfuscation is likely to generate high traveling cost for task assignment over roads, as it cannot accurately estimate the traveling costs distortion caused by location obfuscation. In this paper, we tackle the SC worker location privacy problem over road networks. Considering the network-constrained mobility features of workers, we describe workers' mobility by a *weighted directed graph*, which considers the dynamic traffic condition and road network topology. Based on the graph model, we design a *geo-obfuscation (GO) function* for workers to maximize the workers' overall location privacy without compromising the task assignment efficiency. We formulate the problem of deriving the optimal GO function as a *linear programming (LP)* problem. By using the *angular block structure* of the LP's constraint matrix, we apply *Dantzig-Wolfe decomposition* to improve the time-efficiency of the GO function generation. Our experimental results in the real-trace driven simulation and the real-world experiment demonstrate the effectiveness of our approach in terms of both privacy and task assignment efficiency.

CCS CONCEPTS

• **Security and privacy → Security services**; • **Theory of computation → Mathematical optimization**; *Parallel algorithms.*

KEYWORDS

location privacy, spatial crowdsourcing, geo-obfuscation

ACM Reference Format:
Chenxi Qiu, Anna Squicciarini, Zhuozhao Li, Ce Pang, and Li Yan. 2020. 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 and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3411863

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3411863

1 INTRODUCTION

With ubiquitous wireless connectivity and continued advances in positioning technologies in mobile devices (e.g., smartphones), *spatial crowdsourcing (SC)* is emerging as a novel paradigm to engage a large number of mobile users (workers) to participate in a variety of *location-based services (LBS)* [1, 2], from real-time navigation (e.g., Waze [3]) to journalism and crisis response (MediaQ [4]) to commercial transportation systems (e.g., Uber-like platforms [5]). In SC, workers are expected to physically move to the tasks' location to perform an assigned task (e.g. provide a ride to a customer, take photos, make measurements). As such, to promote cost-effective crowdsourcing work, tasks need to be assigned to workers with low *traveling cost* (e.g., traveling distance/time), which requires workers to disclose their location information to SC servers in real-time. This practice raises privacy issues that are not only related to whereabouts of workers but also related to some other sensitive information such as religions, home/working address, sexual preference, etc [6].

Location privacy protection in SC has been a very active research area in the past few years [7–17]. Considering mobile devices' limited computation capability, instead of using cryptographic techniques [9], a large body of work has been centered on *geo-obfuscation* [10–12, 14], a location privacy protection paradigm that allows workers to report perturbed location instead of true location to servers. Yet, most existing geo-obfuscation designs still consider workers' mobility on a 2 dimensional (2D) plane [18], under which workers are assumed to be able to move in arbitrary directions at random speed without any restriction. Nevertheless, **when workers' mobility is constrained by road networks, 2D-based geo-obfuscation is more likely to generate high cost for task assignment (low quality of service (QoS))**, i.e., tasks are possibly

assigned to workers whose reported locations are physically near to the tasks, but the actual traveling cost is high over roads. Different from on 2D, the sensitivity of *traveling cost* (*cost*) estimation errors to obfuscation in road networks varies considerably with the underlying network structure. As the example in Fig. 1 shows, both obfuscated locations A and B have a small deviation from the actual location P on 2D and their cost estimation errors are close (i.e., which are 60m and 50m, respectively). However, in the road network, the cost estimation error generated by B (1090m) is much higher than that of A (90m), since a direct path exists from A to Q (500m), while there is an unavoidable detour from B to Q (1680m). Besides the road network topology, other mobility constraints over roads also impact the QoS, e.g. traffic [20]. To date, these conditions are not considered in 2D-based methods.

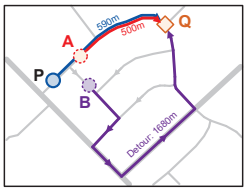
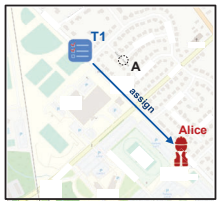


Figure 1: Traveling distances estimated from obfuscated locations A and B over roads, where the distances are calculated by the Dijkstra's algorithm, which can find the shortest path in a graph [19] (P: Actual location. Q: Task location). The Euclidean distances between (A, Q), (B, Q), and (P, Q) on 2D are 480m, 490m, and 540m, respectively.

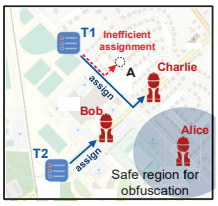
The main reason for the above research gap is that optimal geo-obfuscation over road networks is a very hard problem. First, the impact of geo-obfuscation on both privacy and QoS may vary significantly over different road segments. As such, geo-obfuscation needs to be adaptive to various local road network topology and traffic conditions, which generates high computation load. On the other hand, the geo-obfuscation derivation has to be highly time-efficient, as the obfuscation needs to be updated continuously as workers move from one road segment to another. Moreover, the sensitivity of QoS to geo-obfuscation is non-static over time, i.e., it may change frequently due to traffic conditions [21] (e.g., peak/off-peak hours) and dynamics of the worker pool (e.g., workers can enter/leave the platforms at any time as needed) [12, 13]. With these concerns, a key challenge is how to design a geo-obfuscation approach with *high time-efficiency* to protect worker location privacy over *complex road networks*, particularly in highly dynamic large-scale SC systems.

In this paper, we tackle the aforementioned issues by developing *a time-efficient geo-obfuscation strategy to protect SC worker location privacy over road networks*. Rather than assuming workers' mobility on 2D, we describe workers' mobility in a time-varying *weighted directed graph*, a straightforward and convenient model to take into both road network topology and dynamic traffic conditions. On the basis of this new mobility model, we design a *geo-obfuscation (GO) function* to provide a reference for workers to select their obfuscated location. The objective of the GO function design is twofold: *i) maximize the overall privacy level of all the workers and ii) ensure the cost-effectiveness of task assignment.*

i) Privacy level maximization. The privacy criteria we aim to maximize is the *expected inference error (EIE)* [22], i.e., the expected distortion from the estimated location (by adversary) to the actual location. EIE assumes certain types of prior information that the adversary may obtain, but without considering the posterior information leakage from obfuscated location. As a complementary criterion, we also require the GO function to achieve *geo-indistinguishability (GI)* [10], which limits the posterior information leakage through a *differential privacy* based criterion.



(a) C-I



(b) C-II

Figure 2: Example: Impact of worker distribution on QoS.
C-I: Worker *Alice* and task *T1* are in the region. There is no quality loss if *Alice* selects location "A" to report as *T1* will be always assigned to *Alice*.
C-II: Task *T2*, and workers *Bob* and *Charlie* are added, where the optimal assignment is to assign *T1* to *Charlie* and *T2* to *Bob*. To preserve the assignment optimality, *Alice* has to limit her obfuscated location in a "safe region". If she selects her obfuscated location as "A" that is outside the safe region, *T1* will be assigned to *Alice*, which increases the cost to complete *T1*.

ii) Cost effective task assignment. To promote a cost-effective assignment, existing geo-obfuscation methods (e.g., [13, 22]) primarily focus on reducing the *cost estimation error* for single worker, but without considering the worker location distribution over the region as a whole. In fact, worker location distribution significantly impacts to what extent the cost estimation error is allowed for high QoS. Fig. 2(a)(b) gives an example, which illustrates that the selection of the same obfuscated location ("A") with the same cost estimation error to the task (*T1*) may lead to a significantly different impact on QoS given different worker distribution around. As such, by performing task assignment *sensitivity analysis*, we identify a "safe" region for each worker's obfuscated location, within which the obfuscated location still preserves the assignment optimality. Considering the uneven distribution of workers, the privacy levels of all workers are preserved but can be achieved at different levels across different regions. For example, in the downtown area, the obfuscation is limited to 0.5km maximum for the sake of QoS, but such constraint is unnecessary to be enforced in the rural area, where workers are sparsely distributed with larger safe regions on average.

To achieve both objectives i) and ii), we formulate the problem of *GO function generation (GFG)* as a *linear programming (LP)* problem. To solve GFG, the standard LP approaches (e.g., the simplex methods [23]) will generate extremely high computation load due to GFG's complexity. As a solution, we first conduct *constraint reduction* by exploring network features of GI (Corollary 3.1). Further, by using the angular block structure of the GFG's constraint matrix, we apply Dantzig-Wolfe decomposition to reformulate GFG into a two-level optimization framework, which is composed of a *master program* and *a set of subproblems*. The problems in both levels can be solved efficiently and a near-optimal solution of the original GFG can be iteratively derived via the communication between the two levels.

With respect to performance, simulation results based on Rome taxi trajectory records [24] (including over one million GPS traces) demonstrate that the privacy (measured by EIE) achieved by our approach outperforms the state-of-the-art algorithms by at least 22.34%. Moreover, the simulation results indicate that, compared

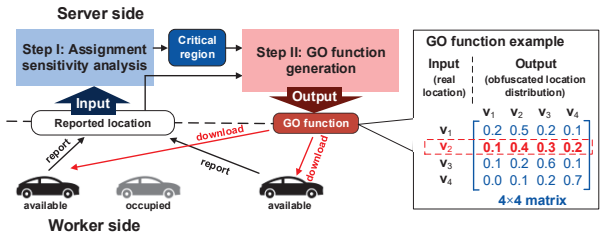


Figure 3: The framework of geo-obfuscation in SC.

with the 2D-based strategy, our approach reduces the total traveling cost of participated workers by 14.22% on average.

In a nutshell, our contributions can be summarized as follows:
1) We develop a mobility model for SC workers operating over roads by taking network-constrained features of workers over roads. Based on the model, we design a GO function for workers to choose their obfuscated location over the road network.
2) We formulate the problem of deriving the GO function as an LP problem, called GFG, which aims to maximize the worker overall privacy without compromising the QoS. GFG is novel not only because it is a new class of location privacy protection problem, but also due to the network-constrained mobility features taken into account in the framework that can be applied to other LBS applications.
3) We design a time-efficient algorithm to solve GFG via constraint reduction and Dantzig-Wolfe decomposition. We conduct a simulation based on real-world dataset to test the performance of our strategy. The experimental results demonstrate the superiority of our method over the state of the arts. We also developed an SC prototype and carried out a pilot study based on the prototype.

2 OVERALL APPROACH

Fig. 3 shows our geo-obfuscation framework in the SC system. Instead of frequently requiring location report from workers, our framework only requires workers to upload their obfuscated location before a snapshot of task assignment [12]. Before uploading location, workers first need to download a GO function generated by the server, and use the function to select the obfuscated location. Consistent with state-of-the-art methods [13, 22], we assume that the server may suffer from *passive (eavesdropping) attack*, not *active (modification) attacks*, i.e., the adversary may obtain workers' locations and the GO function, but cannot modify the GO function.

With the GO function, each worker takes his/her current location as the input and obtains a probability distribution of the obfuscated location as the output. Fig. 3 gives an example, where workers' possible location is assumed to be discrete: $\{v_1, v_2, v_3, v_4\}$. In this case, the GO function can be represented as a (4×4)-matrix. Suppose that a worker's actual location is v_2. As indicated by the matrix in Fig. 3, the probabilities that this worker selects v_1, v_2, v_3, and v_4 as the obfuscated location are 0.1, 0.4, 0.3, and 0.2, respectively.

Note that although the server takes charge of generating the GO function, the workers' location privacy is still guaranteed [12]. Specifically, the GO function is designed to satisfy the privacy criteria (EIE and GI) even if the adversary knows workers' reported location and the GO function (more details will be given in Section 3).

In each round of task assignment, workers can label their status by either *available* or *occupied*. Only *available* workers are considered as candidates for the task assignment and are responsible for reporting their locations to the server. Once receiving a task, each available worker will head towards the assigned task location instantly. The worker's status will be switched to *occupied* and the status won't be switched back to *available* until the worker completes a task and is ready for new ones. For simplicity, in what follows, when we mention "workers", we refer to "available workers".

As illustrated in Fig. 2(a)(b), we cannot ignore the impact of the worker location distribution on the sensitivity of QoS to obfuscation. Accordingly, we consider the worker location distribution (derived from workers' reported location) as a key parameter to generate the GO function. As Fig. 3 shows, the whole process of our geo-obfuscation strategy is composed of two steps:

Step I: Assignment sensitivity analysis. Given workers' reported location as the input, the SC server needs to distribute each task to at least one worker, to minimize the total traveling cost of all the participated workers. Although geo-obfuscation inevitably introduces errors to the input, we note that such errors do not necessarily degrade the QoS if the errors are controlled. As such, we derive a "safe region" for each possible obfuscated location by resorting to *sensitivity analysis* of the task assignment, such that the obfuscation within such region preserves the assignment optimality.

Step II: GO function generation. After being initialized, the GO function needs to be updated by the server based on the change of workers' reported location in each round of task assignment. We assume the overall workers' location distribution to be spatially correlated in adjacent rounds [21], which allows workers to obfuscate their current location with the GO function derived in the previous round [12]. The GO function specifically focuses on the following two goals:

G1) Cost-effective task assignment, i.e., the task assignment based on the geo-obfuscated location achieves a near-optimal assignment. To achieve this goal, for each real location, its obfuscated location is limited to its *safe region* (derived in Step I) with a high probability by the GO function.

G2) Location privacy maximization, i.e., EIE is maximized and GI is satisfied. Considering that workers are unevenly distributed over the road network, we allow the privacy levels (in term of EIE) to be achieved in different levels in different regions.

3 MODEL

In this section, we introduce the system model, including the math notations and the assumptions used throughout the paper.

GO Function. We let $\mathcal{V}$ denote the possible location set of workers over the road network. The GO function $\mathcal{X}$ can be then represented as a map: $\mathcal{V} \rightarrow \mathcal{F}$, where $\mathcal{F}$ denotes the *set of probability distributions over* $\mathcal{V}$. That is, given a worker's true location $v \in \mathcal{V}$ as the input, $\mathcal{X}$ returns the corresponding probability distribution $f_v \in \mathcal{F}$ as the reference for the worker to select his/her obfuscated location to report. Considering the computational tractability of the GO function generation, like [10, 11], we consider the workers' possible location as a *discrete and finite set* $\mathcal{V} = \{v_1, ..., v_K\}$. As such, a more efficient representation of the GO function is by means of a stochastic matrix $\mathbf{X} = \{x_{k,l}\}_{K \times K}$, namely the *GO matrix*, where each $x_{k,l}$ denotes the probability of taking v_l as the obfuscated location given the actual location v_k. In this case, given a real location v_k as the input, the GO function returns a vector $[x_{k,1}, ..., x_{k,K}]$, where each $x_{k,l}$ $(l = 1, ..., K)$ specifies the probability of selecting v_l as the obfuscated location.

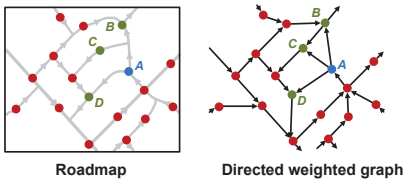


Figure 4: Example of the graph model (The points represent the locations in $\mathcal{V}$, and $\{B, C, D\}$ are the out-neighbors of A).

Graph-based Mobility Model. Considering the network-constrained mobility features of workers over a road network, we model workers' mobility in a *weighted directed graph*. The model assumes that all the locations in $\mathcal{V}$ are in the road network. For each pair of locations $v_k, v_l \in \mathcal{V}$, we use $c_{k,l}$ to denote the *lowest traveling cost* (or *traveling cost* for simplicity) from v_k to v_l over the roads, e.g., which can be interpreted as the shortest traveling distance [13], the lowest traveling time [21], or their combination in the road network. As $c_{k,l}$ is impacted by traffic condition and may change over time, $c_{k,l}$ needs to be updated by the SC server in each round.

By connecting each $v_k \in \mathcal{V}$ to each of its *out-neighbors* v_n (*Definition 3.1*) with a directed edge $e_{k,n}$ from v_k to v_n, we build a *weighted directed graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote the *node set* and the *edge set*, respectively. The weight of each edge $e_{k,n}$ is set by $c_{k,n}$. Fig. 4 gives an example on building the graph given the discrete locations in the road network, where traveling distance is considered as the "cost". Proposition 3.1 implies that it is sufficient to use $\mathcal{G}$ to derive $c_{k,l}$ for any pair of locations $v_k, v_l \in \mathcal{V}$, as the derivation of $c_{k,l}$ is essentially to find the *shortest path* (*Definition 3.2*) from v_k to v_l in $\mathcal{G}$.

Definition 3.1. *(Out-neighbor and in-neighbor) $\forall v_k, v_l \in \mathcal{V}$, v_l is defined as an* out-neighbor *of v_k (and also v_k is defined as an* in-neighbor *of v_l), if workers can travel from v_k to v_l via the* shortest route *without visiting any other location in $\mathcal{V}$.*

Definition 3.2. *(Shortest path) $\forall v_k, v_l \in \mathcal{V}$, the shortest path from v_k to v_l in $\mathcal{G}$ is defined as the path from v_k to v_l such that the total sum of the edges weights is minimum.*

Proposition 3.1. *$\forall v_k, v_l \in \mathcal{V}$, $c_{l,k}$ is equal to the sum weight of the shortest path from v_k to v_l in $\mathcal{G}$ (detailed proof can be found in our technical report in [25]).*

Threat Model and Privacy Criteria. Like [11, 26], we assume that the SC server may suffer from an eavesdropping attack, i.e., information such as workers' reported location, the GO matrix $\mathbf{X}$, and workers' prior location distribution $f_P(v_k)$ ($v_k \in \mathcal{V}$) can be possibly disclosed or leaked to an adversary. The adversary can then estimate the probability distribution of workers' real location via *Bayesian inference models* [12, 14].

We let the random variables P and $\tilde{P}$ denote a worker's real and obfuscated locations, respectively. Given a worker's reported location v_l, the adversary first estimates the *posterior* probability of the worker's real location by resorting to the *Bayes' Equation*:

$$f_{P|\tilde{P}=v_l}(v_k) = \frac{f_P(v_k)x_{k,l}}{\sum_j f_P(v_j)x_{j,l}}, \ \forall k = 1, 2, ..., K. \tag{1}$$

Based on the posterior, the adversary then estimates the worker's actual location by finding the location $\hat{v} \in \mathcal{V}$ that minimizes the expected inference error, i.e.,

$$\hat{v} = \arg\min_{v_r \in \mathcal{V}} \sum_{v_k \in \mathcal{V}} f_{P|\tilde{P}=v_l}(v_k)\, d(v_r, v_k), \tag{2}$$

where d can be either Hamming distance or Euclidean distance [14, 22]. Like [14, 22], in this paper, we consider d as Euclidean distance. The model can be also extended to Hamming distance.

A) Expected inference error (EIE). We define the adversary's EIE, also known as the *unconditional expected privacy* [22, 27], by

$$\sum_l \Pr\left(\tilde{P} = v_l\right) \sum_k f_{P|\tilde{P}=v_l}(v_k)\, d(\hat{v}, v_k) = \sum_l x_{K+1,l}, \tag{3}$$

$$\text{where } x_{K+1,l} = \min_{v_r} \sum_k f_P(v_k) x_{k,l} d(v_r, v_k) \ (l = 1, ..., K) \tag{4}$$

is an intermediate variable to facilitate the computations (details are given in Section 4.2). EIE essentially describes the expected distortion from the estimated location (by adversary) to the actual location, and higher EIE implies higher privacy level achieved. For simplicity, we let $\mathbf{x}_l = [x_{1,l}, x_{2,l}, ..., x_{K,l}, x_{K+1,l}]^\top$ $(l = 1, ..., K)$.

B) Geo-indistinguishability (GI). EIE assumes certain types of prior information that the adversary may obtain, but does not consider the posterior information leaked from obfuscated location. As such, we require the GO function to achieve *GI* [10], which limits the posterior information leakage through a *differential privacy* based criteria. GI over roads is formally defined in Definition 3.3 [13]:

Definition 3.3. *(GI) A GO function $\mathbf{X}$ satisfies ϵ-GI if Equ. (5) is satisfied $\forall v_j, v_k \in \mathcal{V}$,*

$$\frac{f_{P|\tilde{P}=v_l}(v_j)}{f_{P|\tilde{P}=v_l}(v_k)} \le e^{\epsilon \min\{c_{j,k}, c_{k,j}\}} \times \frac{f_P(v_j)}{f_P(v_k)}, \ \forall v_l \in \mathcal{V} \tag{5}$$

where ϵ is the parameter to quantify how much the worker's actual location is disclosed according to the reported location, i.e., higher ϵ implies more information disclosed and a lower privacy level achieved.

Intuitively, Equ. (5) indicates that the reported location v_l won't provide enough information to adversary to distinguish the true location among nearby ones. According to *Definition 3.3*, given each possible obfuscated location v_l, we need to check the posteriors of each pair of locations $v_j, v_k \in \mathcal{V}$, which generates $O(K^3)$ constraints in total. Fortunately, the *transitivity property* of GI over roads (Theorem 3.2) allows us to reduce the number of constraints from $O(K^3)$ to $O(KH)$ without losing the optimality (*Corollary 3.1*), where $H = |\mathcal{E}|$ denotes the number of edges in $\mathcal{G}$.

Theorem 3.2. (Transitivity *[13]) Given any pair of locations $v_1, v_n \in \mathcal{V}$ connected by the shortest path: $(v_1, v_2) \to ... \to (v_{n-1}, v_n)$, Each pair (v_k, v_{k+1}) satisfies ϵ-GI $(k = 1, ..., n-1) \Rightarrow (v_1, v_n)$ satisfies ϵ-GI.*

Corollary 3.1. *(Constraint reduction) The end locations of each edge in $\mathcal{G}$ satisfies ϵ-GI $\Rightarrow$ Each pair of locations in $\mathcal{V}$ satisfies ϵ-GI.*

Corollary 3.1 indicates that, to satisfy ϵ-GI, it is sufficient to formulate the GI constraints only for the end points of each edge in $\mathcal{G}$, where the total number of GI constraints is $O(KH)$. In practice, as $\mathcal{G}$ is approximately a *planar graph*, the number of edges and nodes in $\mathcal{G}$ are actually close, i.e., $H \approx K$, which will be also demonstrated by a real-dataset in Table 1 in Section 6. Hence, after the constraint reduction, the number of GI constraints in GFG is approximately $O(K^2)$.

Finally, by plugging the real location posterior (Equ. (1)) into Equ. (5), the GI constraints for each obfuscated location v_l can be rewritten as a set of linear constraints for $\mathbf{x}_l$:

$$x_{k,l} f_P(v_j) - e^{\epsilon c_{j,k}} f_P(v_k)\, x_{j,l} \le 0, \forall (v_k, v_j) \in \mathcal{E}. \tag{6}$$

For simplicity, we use $\mathbf{\Phi}_l^{\text{GI}}$ to represent the *GI constraint matrix* for $\mathbf{x}_l$, i.e., $\mathbf{\Phi}_l^{\text{GI}} \mathbf{x}_l \le \mathbf{0}$, where $\mathbf{\Phi}_l^{\text{GI}}$ has $2H$ rows and $K+1$ columns:

$$\Phi_l^{\mathrm{GI}} = \begin{bmatrix} \ddots & \cdots & \cdots & \cdots & \iddots & \vdots \\ \cdots & f_P(v_j) & \cdots & -e^{\epsilon c_{j,k}} f_P(v_k) & \cdots & 0 \\ \cdots & -e^{\epsilon c_{j,k}} f_P(v_j) & \cdots & f_P(v_k) & \cdots & 0 \\ \iddots & \cdots & \cdots & \cdots & \ddots & \vdots \end{bmatrix} \begin{matrix} \\ \} \\ \} \\ \\ \end{matrix} \begin{matrix} \\ \forall e_{j,k} \\ \in \mathcal{E} \\ \\ \end{matrix}$$

where each 2 rows correspond to a pair of adjacent locations in $\mathcal{G}$.

4 SYSTEM DESIGN

In this section, we introduce the design of our geo-obfuscation strategy, including the assignment sensitivity analysis (in Section 4.1) and the GO function generation (in Section 4.2).

4.1 Task Assignment and Sensitivity Analysis

Task assignment. We consider a scenario where M tasks need to be assigned to N workers ($N > M$, i.e., the platform has more workers than tasks [5]). The objective of task assignment is to ensure each task to be assigned to one worker and the total traveling cost of all the participated workers is minimized. The assignment can be represented by an indicator matrix $\mathbf{Z} = \{z_{i,j}\}_{N\times M}$, where each $z_{i,j}$ indicates whether task j is assigned to worker i, i.e., $z_{i,j} = 1$ if task j is assigned to worker i; otherwise, $z_{i,j} = 0$. $\mathbf{Z}$ needs to satisfy the constraints $\sum_i z_{i,j} = 1$ for each j ($j = 1, ..., M$), i.e., each task j is assigned to one worker, and the constraints $\sum_j z_{i,j} \leq 1$ for each i ($i = 1, ..., N$), i.e., each worker i can complete up to 1 task. We let $\Omega = \left\{\mathbf{Z} \middle| \sum_i z_{i,j} = 1, \forall j,\ \sum_j z_{i,j} \leq 1,\ \forall i\right\}$ denote the constrained space for $\mathbf{Z}$. Given each worker i's reported location v_{l_i} ($i = 1, ..., N$) and each task j's location v_{q_j} ($j = 1, ..., M$), the task assignment problem can be formulated as:

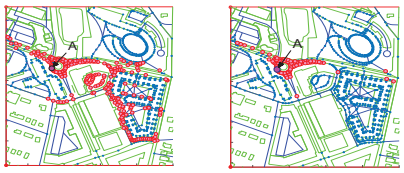
$$\min\ \textstyle\sum_i \sum_j c_{l_i,q_j} z_{i,j}\ \text{ s.t. }\ \mathbf{Z} \in \Omega,\ z_{i,j} \in \{0, 1\}, \tag{7}$$

which can be solved by well-developed algorithms like the *Hungarian algorithm* or *linear programming (LP)* based methods [23].

Sensitivity analysis. We choose to use LP based approaches to solve the assignment problem, from which we can make use of well-developed LP *sensitivity analysis (SA)* tools to yield a "safe region" for geo-obfuscation [23]. Specifically, we first relax the assignment problem to LP by removing the integrality constraints $z_{i,j} \in \{0, 1\}$ in Equ. (7). After that, we derive the optimal solution of the relaxed problem with standard LP approaches (e.g., the simplex methods [23]). As the constraint matrix (defined by Ω) is *totally unimodular*, the LP's solution has to be integral, indicating that it is also the optimal solution of the original assignment problem [23]. In what follows, we let $\tilde{\mathbf{Z}}$ represent the optimal assignment derived based on the workers' obfuscated location.

As we have assumed, the workers' location distribution is spatially correlated in adjacent rounds, which allows us to derive the GO matrix in the next round based on the workers' current reported location. Given the existing reported locations $v_{l_1}, ..., v_{l_{N'}}$ from *unsigned* workers (without loss of generality, assuming workers $i = 1, ..., N'$ receive no assignment), we aim to identify a "safe region" of obfuscated location for any new report. Particularly, for each candidate obfuscated location $v_l \in \mathcal{V}$ from worker i', we derive v_l's *critical region* Θ_l^{cri} ($\Theta_l^{\mathrm{cri}} \subseteq \mathcal{V}$) via SA, such that the corresponding real location within Θ_l^{cri} generates the same optimal solution with $\tilde{\mathbf{Z}}$:

$$\Theta_l^{\mathrm{cri}} = \left\{v_k \middle| \tilde{\mathbf{Z}} = \arg\min_{\mathbf{Z}\in\Omega} \left(\textstyle\sum_j c_{k,q_j} z_{i',j}(t) + \sum_{i=1}^{N'} \sum_j c_{l_i,q_j} z_{i,j}\right)\right\}.$$



(a) # of workers = 5 (b) # of workers = 15

Figure 5: Example: the safe region of the location A.

Here, $\sum_j c_{k,q_j} z_{i',j}(t)$ and $\sum_{i=1}^{N'} \sum_j c_{l_i,q_j} z_{i,j}$ respectively represent the real cost of worker i' and the estimated total cost based on the existing reports in $\mathbf{Z}$. $\Theta_1^{\mathrm{cri}}, ..., \Theta_K^{\mathrm{cri}}$ can be derived in parallel with the existing SA works [23]. Given a worker's real location, we define the *safe region* of obfuscation as the set of locations' with critical region covering the real location. Clearly, if each worker selects the obfuscated within the safe region, the assignment will achieve a near-optimal solution. Fig. 5 gives an example to compare the safe regions of one location ("A") with different workers distributed around, which implies that the worker has smaller "safe region" when the density of workers is higher over the region.

Critical region constraints. To ensure each obfuscated location to be within the safe region with a high probability, we require that for any candidate obfuscated location v_l, the posterior of real location covered by Θ_l^{cri}, $\Pr\left(P \in \Theta_l^{\mathrm{cri}} \middle| \tilde{P} = v_l\right)$, is no smaller than a threshold $1 - \eta$, defining the *critical region constraints*:

$$\Pr\left(P \in \Theta_l^{\mathrm{cri}} \middle| \tilde{P} = v_l\right) \geq 1 - \eta, \tag{8}$$

where $\eta \in [0, 1)$ is a predefined small constant. By plugging the posterior (Equ. 1) into Equ. (8), the critical region constraints can be also written as a set of linear constraints for $\mathbf{x}_l$ ($l = 1, ..., K$):

$$\textstyle\sum_j f_P(v_j) x_{j,l} - \sum_{v_k \in \Theta_l^{\mathrm{cri}}} f_P(v_k) x_{k,l}/(1 - \eta) \leq 0. \tag{9}$$

For simplicity, we use a $(K + 1)$-dimension vector $\mathbf{\Phi}_l^{\mathrm{Cr}}$ to represent the *critical region constraint vector* for $\mathbf{x}_l$, i.e., $\mathbf{\Phi}_l^{\mathrm{Cr}} \mathbf{x}_l \leq \mathbf{0}$, where

$$\mathbf{\Phi}_l^{\mathrm{Cr}} = [f_P(v_1), ..., \underbrace{f_P(v_l) - \textstyle\sum_{v_k \in \Theta_l^{\mathrm{cri}}} f_P(v_k)/(1 - \eta)}_{\text{the } l\text{th element}}, ..., f_P(v_K), 0]$$

Note that even with the critical region constraints, the optimality of the task assignment still cannot be guaranteed due to the following two reasons: 1) The safe region of obfuscated location for each worker is calculated separately, but without considering the uncertainty of other workers' obfuscated location. 2) The derived safe region is calculated based on workers' reported location in the last round and hence it is possibly "unsafe" in the current round.

The above two limitations are unavoidable. For 1), it is computational intractable to derive the safe regions for all the workers, as the number of possible combinations of estimated costs from workers increases exponentially with the number of workers. For 2), calculating the GO function with the reported location in the current round is infeasible, since workers cannot report their location before the GO function being generated. However, even with these two limitations, our approach still approximates the optimal assignment closely according to the experimental results (Fig. 10(b) in Section 6).

4.2 GO Function Generation Problem

The GO function is initialized when the system is first setup. After then, the server updates the GO function (matrix) at each round based on the workers' new reported location as well as the critical

regions derived from the assignment sensitivity analysis. By taking the GO matrix $\mathbf{X}$ as the decision variable, we formulate the problem of generating the GO matrix as a mathematical optimization problem, of which the objective is to maximize the overall expected inference error (EIE) $\sum_l x_{K+1,l}$ (Equ. (3)), while satisfying both *critical region constraints* (Equ. (9)) and *GI constraints* (Equ. (6)). According to Equ. (4), we have $x_{K+1,l} - \sum_k f_P(v_k) x_{k,l} d(v_r, v_k) \leq 0,\ \forall v_r \in \mathcal{V}$, which can be also written in the form of $\mathbf{\Phi}_l^{\text{In}} \mathbf{x}_l \leq \mathbf{0}$, where $\mathbf{\Phi}_l^{\text{In}}$ is a matrix with K rows and $K+1$ columns:

$$\mathbf{\Phi}_l^{\text{In}} = \begin{bmatrix} -f_P(v_1)d(v_1,v_1) & \cdots & -f_P(v_K)d(v_1,v_K) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ -f_P(v_1)d(v_K,v_1) & \cdots & -f_P(v_K)d(v_K,v_K) & 1 \end{bmatrix}$$

Given that the constraints $\mathbf{\Phi}_l^{\text{GI}} \mathbf{x}_l \leq \mathbf{0}$, $\mathbf{\Phi}_l^{\text{Cr}} \mathbf{x}_l \leq \mathbf{0}$, $\mathbf{\Phi}_l^{\text{In}} \mathbf{x}_l \leq \mathbf{0}$, and the objective function $\sum_l x_{K+1,l}$ are all linear, and the decision variables $\mathbf{X} = \{x_{k,l}\}_{K \times K}$ are defined in a continuous region, the *GO function generation (GFG)* problem can be formulated as an LP:

$$\max \quad \sum_l x_{K+1,l} \tag{10}$$

$$\text{s.t.} \quad \mathbf{\Phi}_l^{\text{GI}} \mathbf{x}_l \leq \mathbf{0},\ \mathbf{\Phi}_l^{\text{Cr}} \mathbf{x}_l \leq \mathbf{0},\ \mathbf{\Phi}_l^{\text{In}} \mathbf{x}_l \leq \mathbf{0},\ \forall l \tag{11}$$

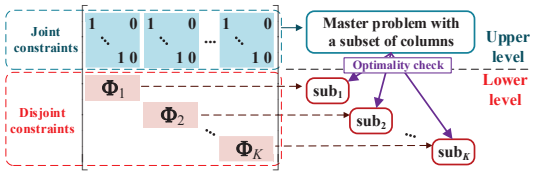
$$\sum_l x_{k,l} = 1,\ \forall k \text{ (prob. unit measure)} \tag{12}$$

Note that in the GI constraints (Equation (5)) and the critical regions constraints (Equation (9)) of GFG, distances between workers/tasks are defined over the road network and can be updated by the server in each round based on the traffic. Therefore, both road network topology and traffic dynamics have been considered by the solution of GFG. GFG can be solved by standard LP approaches such as the simplex methods [23]. This, however, introduces challenges with respect to time efficiency and scalability. The number of decision variables in the GO matrix $\mathbf{X}$ is quadratic to the number of discrete locations in $\mathcal{V}$, e.g., thousands of discrete locations will generate millions of decision variables in GFG, leading to an extremely high computation load. On the other hand, to account for realistic applications where worker location distribution changes all the time, the derivation of optimal $\mathbf{X}$ is supposed to be time-efficient to handle the highly dynamic inputs. To tackle this issue, in Section 5, we introduce how to generate the GO matrix in a scalable and time-efficient way.

5 GO FUNCTION GENERATION

A promising route to solve large-scale LP problems is to adopt *decomposition* techniques based on how decision variables in the problems are coupled [28]. For simplicity, we let $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1^\top & \dots & \mathbf{x}_K^\top \end{bmatrix}^\top$ and $\mathbf{\Phi}_l = \begin{bmatrix} \mathbf{\Phi}_l^{\text{GI}\top} & \mathbf{\Phi}_l^{\text{Cr}\top} & \mathbf{\Phi}_l^{\text{In}\top} \end{bmatrix}^\top$. The whole GFG constraint matrix $\mathbf{\Phi}$ for $\mathbf{x}$ (i.e., $\mathbf{\Phi}\mathbf{x} \leq \mathbf{0}$) is shown in Fig. 6(a), where a *block angular* structure can be found, i.e., 1) the constraint matrices $\mathbf{\Phi}_1, ..., \mathbf{\Phi}_K$ (for $\mathbf{x}_1, ..., \mathbf{x}_K$ respectively) are all disjoint; 2) only the joint constraints $\sum_l x_{k,l} = 1$ ($k = 1, ..., K$) link together the different decision vectors $\mathbf{x}_1, ..., \mathbf{x}_K$. Such block angular structure makes GFG well-suited to *Dantzig-Wolfe (DW) decomposition* [29].

DW decomposition relies on *column generation (CG)* to improve the tractability of large-scale LP [30]. By rewriting GFG in a DW formulation (defined in Equation (13)-(14)) and solving it via the revised simplex method [23], most extreme points of GFG are non-basic (i.e., the corresponding decision variables are set by zero) during the whole search process [31]. Therefore, the DW formulation can be solved by involving only a portion of extreme points.



(a) The block angular structure of GFG's constraint matrix (b) Two levels of DW decomposition

Figure 6: DW decomposition.

5.1 DW Formulation

We let Λ_l denote the polyhedron defined by the constraint matrix Φ_l ($l = 1, ..., K$) and let $\mathcal{X}_l = \left\{\hat{\mathbf{x}}_l^1, ..., \hat{\mathbf{x}}_l^{T_l}\right\}$ denote the set of extreme points of Λ_l. Then, any decision vector $\mathbf{x}_l \in \Lambda_l$ can be represented as a convex combination of $\hat{\mathbf{x}}_l^1, ..., \hat{\mathbf{x}}_l^{T_l}$ (Minkowski-Weyl's Theorem [23]): $\mathbf{x}_l = \sum_{t=1}^{T_l} \lambda_{l,t} \hat{\mathbf{x}}_l^t$, where $\sum_{t=1}^{T_l} \lambda_{l,t} = 1$ and $\lambda_{l,t} \geq 0$. Replacing $\mathbf{x}_l$ by $\sum_{t=1}^{T_l} \lambda_{l,t} \hat{\mathbf{x}}_l^t$, GFG can be rewritten as the following *master program (MP)*:

$$\max \quad \sum_l \sum_t \lambda_{l,t} \hat{x}_{K+1,l}^t \tag{13}$$

$$\text{s.t.} \quad \sum_l \sum_t \lambda_{l,t} \hat{x}_{k,l}^t = 1, \forall k, \sum_{t=1}^{T_l} \lambda_{l,t} = 1, \lambda_{l,t} \geq 0, \forall l \tag{14}$$

The decision variables in MP are $\lambda_{l,t}$ ($t = 1, ..., T_l, l = 1, ..., K$) and each $\lambda_{l,t}$ corresponds to an extreme point in the polyhedron Λ_l. Since the total number of extreme points in all the polyhedrons are exponential to K (the number of discrete locations in V), MP itself does not decrease the time complexity if it is solved directly by standard LP approaches. Fortunately, most extreme points in DW-formulated MPs are non-basic when searching the optimal [31], indicating that the idea of CG can be applied.

5.2 The Column Generation Algorithm

The algorithm is composed of the steps **S1-S3** (the pseudo code can be found in the technical report [25]):

S1: Initialization. By considering only a subset of extreme points in MP, a *restricted MP (RMP)* (Definition 5.1) is formulated.

Definition 5.1. *(RMP) Given a subset of extreme points $\overline{\mathcal{X}}_l$ ($\overline{\mathcal{X}}_l \subseteq \mathcal{X}_l$) in each polyhedron Λ_l, we define the corresponding RMP, denoted by RMP($\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K$), as the MP with only $\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K$ being considered:*

$$\overline{\boldsymbol{\lambda}}^* = \left\{ \begin{array}{ll} \max & \sum_l \sum_{t \in \overline{\mathcal{X}}_l} \lambda_{l,t} \hat{x}_{K+1,l}^t \\ \text{s.t.} & \sum_l \sum_{t \in \overline{\mathcal{X}}_l} \lambda_{l,t} \hat{x}_{k,l}^t = 1, \forall k, \sum_{t=1}^{T_l} \lambda_{l,t} = 1, \lambda_{l,t} \geq 0, \forall l \end{array} \right\}$$

where $\overline{\boldsymbol{\lambda}}^*$ denotes the optimal solution of RMP($\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K$).

Here, we also define the dual problem of RMP (D-RMP), which will be used in **S2**.

Definition 5.2. *(D-RMP) The dual problem of RMP($\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K$) [23], denoted by D-RMP($\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K$), is defined as:*

$$(\overline{\boldsymbol{\pi}}^*, \overline{\boldsymbol{\mu}}^*) = \left\{ \begin{array}{ll} \min & \sum_k \pi_k + \sum_l \mu_l \\ \text{s.t.} & \sum_k \hat{x}_{k,l}^t \pi_k + \mu_l \geq \hat{x}_{K+1,l}^t, \forall t \in \overline{\mathcal{X}}_l,\ l = 1, ..., K. \end{array} \right\}$$

where $(\overline{\pi}^*, \overline{\mu}^*)$ $(\overline{\pi}^* = [\overline{\pi}_1^*, ..., \overline{\pi}_K^*]$ *and* $\overline{\mu}^* = [\overline{\mu}_1^*, ..., \overline{\mu}_K^*])$ *denotes the optimal solution of D-RMP*$(\overline{\mathcal{X}}_1, ..., \overline{\mathcal{X}}_K)$.

S2: Optimality test. We solve the RMP and test whether its solution $\overline{\lambda}^*$ achieves MP's optimal based on Proposition 5.1.

Proposition 5.1. *(Optimality test criteria) To test* $\overline{\lambda}^*$*'s optimality in MP, it is sufficient to test whether* $(\pi^*, \overline{\mu}^*)$ *defined in Definition 5.2, satisfies* $\min_{t \in \mathcal{X}_l} \left\{ \sum_k \hat{x}_{k,l}^t \overline{\pi}_k^* + \overline{\mu}_l^* - \hat{x}_{K+1,l}^t \right\} \geq 0$ $(l = 1, ..., K.)$*, where the derivation of* $\min_{t \in \mathcal{X}_l} \left\{ \sum_k \hat{x}_{k,l}^t \pi_k + \mu_l - \hat{x}_{K+1,l}^t \right\}$ *is essentially an LP problem (labeled by* sub_l*) with the decision variables* $\mathbf{x}_l$ *constrained in the polyhedron* Λ_l*:*

$$\text{sub}_l : \ \overline{\mathbf{x}}_l^* = \{ \ \min \quad \sum_k x_{k,l} \overline{\pi}_k^* + \overline{\mu}_l^* - x_{K+1,l} \text{ s.t. } \mathbf{x}_l \in \Lambda_l. \ \}$$

where $\overline{\mathbf{x}}_l^*$ *is the optimal solution of* sub_l*. The detailed proof of Proposition 5.1 can be found in our technical report in [25].*

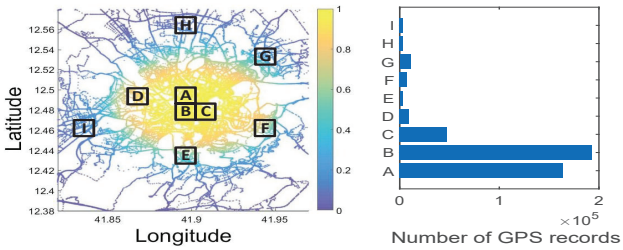
S3: Column generation. We use ζ_l to denote the objective value of sub_l, i.e., $\zeta_l = \min_{\mathbf{x}_l \in \Lambda_l} \left\{ \sum_k x_{k,l} \overline{\pi}_k^* + \overline{\mu}_l^* - x_{K+1,l} \right\}$. If $\exists \zeta_l < 0$, the optimal of MP hasn't been achieved. Then, the corresponding sub_l will suggest a new extreme point (column) to add to the RMP to improve the objective value. After that, we move to **S2** to test the optimality of the new solution.

S3 and **S2** are repeated until the MP's optimal is found. As Proposition 5.1 indicates, the process of optimality test can be partitioned into a list of subproblems sub_1, ..., sub_K, where each sub_l $(l = 1, ..., K)$ has its decision variables $\mathbf{x}_l$ only constrained in the polyhedron Λ_l. As the decision variables $\mathbf{x}_1$, ..., $\mathbf{x}_K$ are fully decoupled in sub_1, ..., sub_K, they can be derived in parallel. As Fig. 6(b) shows, the process of **S2** and **S3** follows a two-layer framework: a RMP in the upper layer and sub_1, ..., sub_K in the lower layer. The two layers communicate with each other and are updated in each iteration, until the RMP's optimal solution converges to the MP's optimal.

We use the superscript $^{(n)}$ to denote the values set/derived in iteration n. Note that RMP and each sub_l can be solved efficiently in each iteration, as they only contain $O(K)$ decision variables (i.e., $(\overline{\pi}, \overline{\mu})$ and $\mathbf{x}_l$). When the algorithm converges to the near optimal, we need to solve RMP$(\overline{\mathcal{X}}_1^{(n)}, ..., \overline{\mathcal{X}}_K^{(n)})$, which has at most nK decision variables, as we only add up to 1 column for each polyhedron in each iteration. Hence, the next question is how many iterations (denoted by L) are needed for convergence.

Convergence analysis. To improve the speed of convergence, in **S1**, we initialize each $\overline{\mathcal{X}}_l$ by the extreme point $\mathbf{e}_l$ (i.e., a $K+1$ dimension vector with the lth entry equal to 1 and all the other entries equal to 0). It means that the initial RMP only includes the extreme points $\mathbf{e}_1$, ..., $\mathbf{e}_K$. By selecting $\mathbf{e}_1$, ..., $\mathbf{e}_K$, the feasible region of the RMP is guaranteed to be non-empty, i.e., there is always a feasible solution $\overline{\lambda}$ with $\lambda_l^1 = 1$ and $\lambda_l^t = 0 \, \forall t > 1$ $(l = 1, ..., K)$, which ensures D-RMP to be bounded and hence improves the algorithm convergence at the beginning [30].

Nevertheless, in CG, there is possibly a long tail of the convergence (pointed out by our experimental results in Fig. 8(a)). As a solution, we set a negative threshold ξ with small magnitude, such that the algorithm will be ended immediately once $\min_l \{\zeta_l\}$ reaches ξ. Our experimental results indicate that, with proper value set to ξ, the convergence of CG will be improved significantly (e.g., use up to 5 iterations in Fig. 8(d) and Fig. 12(c)) with the objective



(a) Heap map of GPS records over Rome. (b) # of records over regions.

Figure 7: The Rome taxi cab dataset.

value (EIE) sacrificed a little (e.g., by up to 6.37% in Fig. 10). More details will be discussed in Section 6.1. For theoretical interests, we give an upper (dual) bound of the MP's optimal in Theorem 5.2 to check how close our solution can achieve the optimal:

Theorem 5.2. *In each iteration* n *of the CG algorithm,* $\omega^{(n)} = \sum_k \overline{\pi}_k^{*(n)} + \sum_l \left(\overline{\mu}_l^{*(n)} - \zeta_l^{(n)} \right)$ *offers an upper bound of MP's optimal. The detailed proof can be found in our technical report [25].*

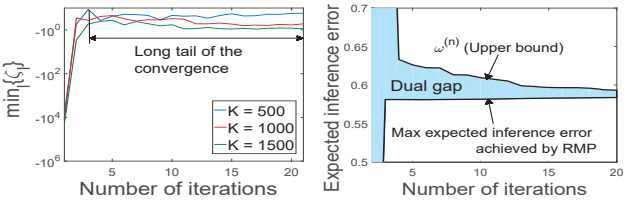
6 PERFORMANCE EVALUATION

In this section, we turn our attention to practical applications of our geo-obfuscation approach. We carry out an extensive evaluation of our method using a real dataset of over one million vehicle GPS records in Section 6.1, and report experimental results with our system prototype in Section 6.2. The main metrics we measure include:
(i) *Privacy level: Expected inference errors (EIE)* defined by Equ. (3).
(ii) *Total traveling cost*, defined as the total traveling distance of all the participating workers to the task location. We primarily consider "traveling distance" as the cost in the experiments as other related metrics, e.g. traveling time, are hard to measure in the dataset. The traveling distance from each worker to his/her task is calculated using the Dijkstra's algorithm [19].
(iii) *Number of iterations* to derive the GO function in CG.

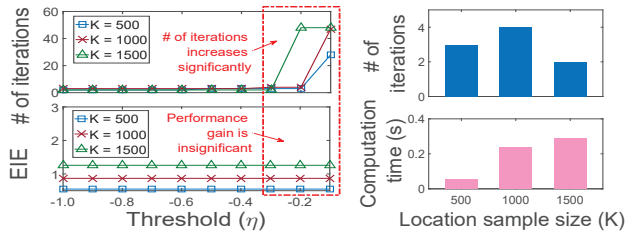
6.1 Trace-driven Simulation

Dataset. We conduct simulations by using a publicly available taxi cab trajectory dataset in Rome [24]. We select to use a taxi dataset since taxi services can be also considered as a type of SC operating over the road network, where a customer's "pickup location" can be considered as the task location. The dataset contains GPS coordinates of approximately 290 taxis in Rome collected over 30 days. Fig. 7(a) depicts the heat map of all taxi cabs' recorded location. As shown, the taxi cabs' location records are not evenly distributed over the city, e.g., taxi cabs are more likely located in downtown rather than in the suburbs. We grid the whole map, and select 9 regions "A", "B", ..., "I" (Fig. 7(a) shows the regions on the map) to check how the workers' density can impact both privacy and QoS, e.g. "A"–"C" are in downtown with high-density workers, while regions "D"–"I" are in suburbs with low-density workers. Fig. 7(b) compares the number of GPS records in different regions.

Benchmarks. We compare our geo-obfuscation strategy with two representative geo-obfuscation algorithms:
1) *2D-based approach (2D)* [12], which aims to minimize the total traveling cost with geo-indistinguishability satisfied.
2) *VSC-Based approach (VSC)* [13], which aims to protect location privacy of vehicles in SC, with vehicles' network-constrained mobility features considered. Similar to our method, VSC determines



(a) CG convergence with different K (b) Dual gap convergence ($K = 1000$)



(c) The number of iterations and performance gain with different ξ (d) # of iterations and computation time with different K.

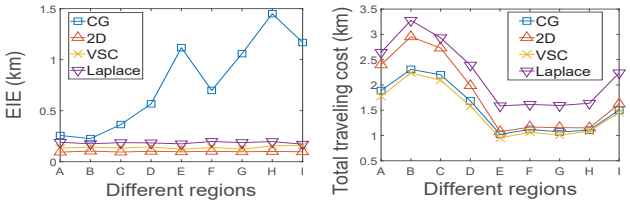
Figure 8: Time efficiency of CG.

the obfuscation function by following an LP framework. But the objective of VSC is to minimize the cost estimation error of a single worker without considering the worker distribution over the region. 3) *Laplacian obfuscation (Laplace)*, where the obfuscation probabilities are calculated by $f_{\tilde{P}|P=v_k}(v_l) \propto e^{-\epsilon \frac{d(v_l,v_k)}{D_{\max}}}$, and $D_{\max}$ is the maximum distance between any two locations in the target region.

Time-efficiency. In Corollary 3.1, we have theoretically proved that the number of GI constraints is reduced to $O(KH)$ by using constraint reduction, where K and H denote the number of nodes and the number of edges in the graph $\mathcal{G}$, respectively. We now test how the number of GI constraints is actually reduced in the real-world road map. We sample a set of discrete locations in each region, where every 10 *road segments*[1] have at least one location point sampled. We then build the weighted directed graph given the sample in each region. Table 1 shows the ratio of H to K in $\mathcal{G}$ across different regions as well as the percentage of GI constraints reduced by the constraint reduction. The table demonstrates that 1) H is not significantly higher than K in any region, e.g., H/K is at most 1.42; 2) the number of GI constraints is significantly reduced by constraint reduction, i.e., on average it is reduced by 99.5%.

We next evaluate the time efficiency of CG. Here, we only depict the results for region "A" as a representative, which has relative higher number of road segments (9,861 segments) and taxis' GPS records (163,938 records), which tends to generate higher computation load. Fig. 8(a) shows the change of $\min_l\{\zeta_l\}$ over iterations (i.e., the algorithm achieves the optimal when $\min_l\{\zeta_l\} = 0$ (Proposition 5.1)). We have two observations from the figure: 1) $\min_l\{\zeta_l\}$ converges faster when the location sample size K is smaller, and 2) after a fast convergence of $\min_l\{\zeta_l\}$ in first 3 or 4 iterations, there is a long tail in the convergence. In Fig. 8(b), we also show the dual gap of the algorithm, i.e., the gap between $\omega^{(n)}$ (derived in Theorem 5.2) and the maximum EIE achieved by the RMP. As the optimal EIE is within the dual gap, the figure indicates that our approach can achieve near-optimal after the 4th iteration, where

[1]Road segment is defined as the segment without furcation, turn, joining with other road segments [13]



(a) Privacy (b) Cost

Figure 9: Comparison with 2D and VSC.

the *approximation ratio* (the ratio of the optimal EIE to the EIE achieved by our approach) is up to 1.064.

Table 1: Constraints reduction.

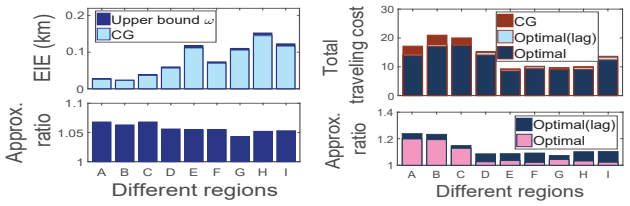
Regions	A	B	C	D	E	F	G	H	I
H/K ratio	1.31	1.42	1.19	1.28	1.08	1.21	1.22	1.27	1.22
Pct. of constraints reduced	99.9	99.8	99.6	99.7	99.5	99.3	99.5	99.0	99.4

As indicated by Fig. 8(a), the algorithm convergence will slow down after $\min_l\{\zeta_l\}$ reaches a certain level. Hence, it is unnecessary to wait until $\min_l\{\zeta_l\} = 0$. Instead, we choose to improve the time efficiency of our algorithm by slightly sacrificing the optimality of the GO function. We select a negative number $\xi < 0$ that is close to 0 as a threshold of $\min_l\{\zeta_l\}$, i.e., the algorithm is terminated once $\min_l\{\zeta_l\} \geq \xi$. Clearly, a higher value for ξ enforces the derived GO function to better approximate the optimal, but tends to generate a higher computation load. Fig. 8(c) shows the number of iterations of the algorithm and the corresponding EIE values, with ξ values increased from -1.0 to -0.1. From the figure, we can see that when ξ reaches a peak (i.e., $\xi > -0.3$ when $K = 1500$ and $\xi > -0.2$ when $K = 500$ or 1000), the number of required iterations increases rapidly (by 10 to 20 times), but the corresponding EIE gain is insignificant. Accordingly, we set ξ such that the number of iterations is maintained at a low level without significantly affecting EIE (e.g., $\xi = -0.3$ for region "A") in the following experiment.

Finally, in Fig. 8(d), we list the number of iterations of the algorithm and the corresponding computation time, when the location sample size (K) equals 550, 1000, and 1500. Fig. 8(d) shows that the number of iterations is at most 4 and the highest computation time is 0.28s. Fig. 8(d) also indicates that 1) the number of iterations is not impacted by K significantly, while 2) the computation time increases with the increase of K, as a higher K leads to higher computation load for each subproblem in the lower layer of CG.

Privacy and cost. We next evaluate our approach in terms of both privacy and traveling cost. We ran the simulation for the 9 regions separately. Each simulation lasts for 60 minutes (from 00:00:00 to 01:00:00 in the trace). We set the parameter ϵ by 1/km for ϵ-GI (Definition 3.3). We first show the *EIE* and the *total traveling cost* achieved by our approach (labeled by CG) in different regions in Fig. 9(a)(b), with the comparison of the three benchmarks 2D, VSC, and Laplace. All 2D, VSC, and Laplace require ϵ-GI (ϵ is set by 1/km as well). The two figures indicate that, with higher density of workers, CG in A, B, and C achieve higher total traveling cost and lower EIE than other regions. When the density of workers is higher, to guarantee the optimality of task assignment, the safe region of obfuscated location needs to be smaller (see Fig. 5(a)(b)). This, on average, makes the selected obfuscated location closer to the actual worker's location, leading to a lower EIE from the adversary.

Moreover, Fig. 9(a) demonstrates that CG is more effective in protecting worker location privacy than 2D, VSC, and Laplace. Besides



(a) Privacy (b) Cost

Figure 10: Comparison with bounds/idealized scenario.

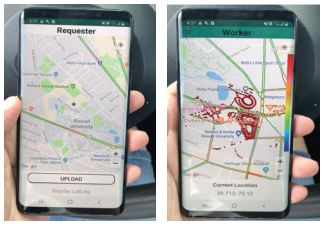
achieving ϵ-GI, CG aims to minimize EIE. ϵ-GI does not always generate higher EIE, since ϵ-GI primarily aims to control posterior information exposure and hence to obfuscate location such that different real locations are hard to differentiate. While, methods based on EIE tend to select obfuscated location with higher distortion from the real location. Fig. 9(b) indicates that the total traveling cost follows CG ≈ VSC < 2D < Laplace. CG can better reduce the total traveling cost compared with 2D and Laplace because 1) CG considers the workers' mobility features over roads, 2) CG derives a safe region for each obfuscated location by considering the worker distribution over the region, and 3) the GO function in CG is defined in a fine-grained location set due to the high-efficiency of CG's computation framework (e.g., 2D samples 1 location per 1km×1km grid, while the average distance between neighbor sample point in CG is less than 100m). While VSC has slightly lower cost than CG, VSC facilitates cost-effective task assignment by unnecessarily minimizing cost estimation error at the expense of privacy.

Fig. 10(a) compares the EIE achieved by CG with a theoretical upper bound (derived in Theorem 5.2), where the ratio of the upper bound to the EIE in CG ranges from 1.043 to 1.068 across the different regions. As the optimal solution is no higher than the upper bound, the approximation ratio of CG is at most 1.068, indicating that CG approximates the optimal EIE closely.

Even though CG achieves lower cost than 2D, it still cannot guarantee the optimality of task assignment as the safe region of each worker's obfuscation is derived separately with lag information (as analyzed in Section 4.1). Hence, it is interesting to check how close CG can achieve the actual lowest cost. Here, we derive the lowest traveling cost that can be achieved in the following two scenarios as the benchmarks: 1) when the optimal traveling cost of workers is estimated by workers' actual location in the last round, labeled by "OPT(lag)"; and 2) when the optimal traveling cost of workers is estimated by workers' actual location in the current round, labeled by "OPT". Fig. 10(b) compares the total traveling cost of OPT(lag), OPT, and CG, and also lists the approximation ratios of CG to OPT(lag) and OPT, respectively. We have two observations in the figure: 1) CG in "A"–"C" with higher-density workers, suffer a larger gap from "OPT(lag)", since the optimality of task assignment is subject to change when the worker's safe region is smaller. 2) In contrast, the gap between "OPT(lag)" and "OPT" is smaller in "A"–"C", due to the higher spatial correlation of workers' location in these regions, i.e., workers move relatively more slowly in the downtown area.

6.2 Pilot Study based on Prototype

In addition, we have built a prototype of SC, including the functions of task request/assignment and geo-obfuscation. We have developed an Android APP on smartphones based on the Google map API, where Fig. 11(a)(b) shows the user interface. The APP allows users



(a) Requester (b) Worker

Figure 11: User interface.

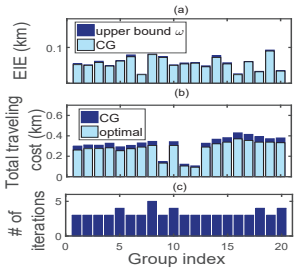


Figure 12: Performance.

to register/log in as a requester/worker. With the APP, a requester can upload his/her task with the location specified, and a worker can download a GO matrix from the server. According to the GO matrix, a worker can select the obfuscated location, and may receive a task assigned by the server. After then, the worker can accept the task by clicking "accept" button, and a route will be displayed on the map to navigate this worker to the task location.

We conduct 20 groups of test, where in each group we deployed 5 workers and 3 tasks (tasks are randomly distributed over the Rowan campus). We sample 1640 discrete locations over the local road network. Every time a worker reports the location, the APP approximates the worker's current location by its nearest sampled discrete location (i.e., measured by the Euclidean distance). Fig. 12(a) and Fig. 12(b) show the EIE and the total traveling cost in different groups, with the comparison of the EIE's upper bound ω and the actual lowest traveling cost, respectively. Fig. 12(c) lists the number of iterations in CG in each group. The figure demonstrates that our approach achieves a near-optimal EIE (i.e., with approximation ratio up to 1.07) with low computation load (i.e., up to 5 iterations). The approximation ratio of the traveling cost is relatively high (i.e., up to 1.171) as approximating each true location to its nearest sampled location inevitably introduces errors to the task assignment.

7 RELATED WORK

In this section, we summarize the existing works relevant to ours, including *location privacy criteria* and *obfuscation based strategies*.

Location privacy criteria. The discussion of location privacy criteria can date back to more than ten years ago, when Gruteser and Grunwald [32] first introduced the notion *location k-anonymity* based on the well-known concept of *k-anonymity* [33]. While different from our approach, location k-anonymity protects users' privacy by hiding their identities (i.e., it is indistinguishable among a set of k users' identities given their location reports). Later, two privacy notions for location obfuscation have been proposed based on statistical quantification of attack resilience: EIE [22] and GI [10]. Both privacy notions have their own limitations and are complementary to each other: EIE based approaches assume certain types of prior information that the adversary may obtain, but require no restriction on the posterior information gain from the exposure of obfuscated locations. GI-based approaches limit the posterior information leakage through a *differential privacy* based criteria, but they are susceptible to the inference attacks using prior knowledge. As such, recent works (e.g., [14]) have proposed to strategically combine the two notions to double shield users' location privacy.

Obfuscation based approach. Based on the notions of EIE and GI, a large body of obfuscation based approaches have been proposed to achieve either of these two privacy criteria (e.g., [10–12, 22, 34])

or their combination [14]. As location information error introduced by geo-obfuscation may lead to quality loss in LBS, a key issue that has been discussed in obfuscation based approaches is how to trade-off privacy and QoS. For example, Shokri et al. [22] advocated an optimal geo-obfuscation mechanism to maximize the EIE given the quality loss constraint, where quality loss is measured by the expected distortion from obfuscated location to actual location. Following by the optimization framework in [22], Theodorakopoulos et al. [34] proposed to maximize EIE with considering the privacy leakage due to sequential correlation of locations in user's trajectory. GI has been also adopted by many recent works [15] as a privacy constraint. For instance, besides proposing the notion of GI, Andrés et al. [10] developed a location perturbation technique to achieve GI by adding noise to actual location, drawn from a polar Laplacian distribution. Given the restriction of GI, Bordenabe et al. [11] proposed an optimization framework for geo-obfuscation to minimize the quality loss (i.e., expected distortion between obfuscated and true locations) for each single user, while Wang et al. [12] considered the quality loss generated by all the users (workers) as a whole and proposed a location privacy-preserving task assignment algorithm to minimize the total traveling cost.

In a nutshell, the strategies [11, 12, 22, 34] are all based on the 2D model, which is hardly to be applied to SC over road networks. In addition, although all these techniques follow an optimization framework like ours, they rely on centralized approaches that have to deal with $O(K^2)$ decision variables in LP, which generates extremely high computation load considering the frequently changed inputs (e.g., highly dynamic traffic) in the optimization. Moreover, these approaches apply uniform privacy criteria over the whole target region without considering the different privacy requirements due to users' (workers') uneven density over the region." To date, the strategy closest to ours is our prior work [13], which also obfuscates vehicles' locations by following an LP framework, with the vehicles' network-constrained mobility features considered. However, the LP formulated in [13] is to minimize the cost estimation error of a single worker. As a result, the derived obfuscation function in [13] does not consider the distribution (density) of multiple workers, leading to a uniform privacy level (GI) over the region.

8 CONCLUSIONS

In this paper, we have developed a new geo-obfuscation strategy to protect workers' locations over road networks in SC. We modeled workers' mobility with considering the road network topology and dynamic traffic conditions. Our proposed geo-obfuscation approach follows an LP framework, of which the objective is maximize the EIE from adversary with the constraints of task assignment cost and geo-indistinguishability (GI) satisfied. Considering the highly dynamic inputs of the LP in SC, we devise a time-efficient algorithm by resorting to DW decomposition. The trace-driven simulation results have demonstrated the effectiveness of our approach over the state of the arts in terms of both privacy and QoS.

We see several promising directions for this research. First, our current work accounts only for homogeneous workers (e.g., either vehicles or pedestrians), without considering heterogeneous mobile workers with different mobility features (e.g., a mixture of vehicles and pedestrians). Also, this work can be extended to general LBS applications, where service utilities are defined in different ways.

9 ACKNOWLEDGEMENTS

This work was partly supported by U.S. NSF grant NSF-1453080.

REFERENCES

[1] L. Kazemi and C. Shahabi. Geocrowd: Enabling query answering with spatial crowdsourcing. In *Proc. of ACM SIGSPATIAL*, pages 189–198, 2012.
[2] L. Kazemi, C. Shahabi, and L. Chen. Geotrucrowd: Trustworthy query answering with spatial crowdsourcing. In *Proc. of ACM SIGSPATIAL*, pages 314–323, 2013.
[3] M. A.-Naseri, P. Chakraborty, A. Sharma, S. B. Gilbert, and M. Hong. Evaluating the reliability, coverage, and added value of crowdsourced traffic incident reports from waze. *Transportation Research Record*, 2672(43):34–43, 2018.
[4] MediaQ. https://imsc.usc.edu/platforms/mediaq/, 2019. Accessed: 2019-07-22.
[5] Y. Tong, L. Chen, and C. Shahabi. Spatial crowdsourcing: Challenges, techniques, and applications. *VLDB Endow.*, 10(12):1988–1991, August 2017.
[6] H. To, G. Ghinita, L. Fan, and C. Shahabi. Differentially private location protection for worker datasets in spatial crowdsourcing. *IEEE TMC*, pages 934–949, 2017.
[7] V. Bindschaedler, R. Shokri, and C. Gunter. Plausible deniability for privacy-preserving data synthesis. *VLDB Endow.*, 10(5):481–492, January 2017.
[8] K. Chatzikokolakis, C. Palamidessi, and M. Stronati. Constructing elastic distinguishability metrics for location privacy. *PoPETs*, 2015:156–170, 2015.
[9] G. Ghinita et al. Private queries in location based services: Anonymizers are not necessary. In *Proc. of ACM SIGMOD*, 2008.
[10] M. Andrés et al. Geo-indistinguishability: Differential privacy for location-based systems. In *Proc. of ACM CCS*, pages 901–914, 2013.
[11] N. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proc. of ACM CCS*, 2014.
[12] L. Wang, D. Yang, X. Han, T. Wang, D. Zhang, and X. Ma. Location privacy-preserving task allocation for mobile crowdsensing with differential geo-obfuscation. In *Proc. of WWW*, 2017.
[13] C. Qiu and A. C. Squicciarini. Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability. In *Proc. of IEEE ICDCS*, 2019.
[14] L. Yu, L. Liu, and C. Pu. Dynamic differential location privacy with personalized error bounds. In *Proc. of ACM NDSS*, 2017.
[15] K. Fawaz and K. G. Shin. Location privacy protection for smartphone users. In *Proc. of ACM CCS*, pages 239–250. ACM, 2014.
[16] H. To, G. Ghinita, and C. Shahabi. A framework for protecting worker location privacy in spatial crowdsourcing. *VLDB Endow.*, 7(10):919–930, June 2014.
[17] A. Sarker, C. Qiu, H. Shen, H. Uehara, and K. Zheng. Brake data-based location tracking in usage-based automotive insurance programs. In *2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 229–240, 2020.
[18] D. B. Johnson and D. A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks.* Springer US, Boston, MA, 1996.
[19] Harsh Bhasin. *Algorithms: Design and Analysis.* Oxford Univ Press, 2015.
[20] H. Huang, G. Gartner, J. M. Krisp, M. Raubal, and N. Weghe. Location based services: ongoing evolution and research agenda. *JLBS*, 12(2):63–93, 2018.
[21] L. Yan, H. Shen, J. Zhao, C. Xu, F. Luo, and C. Qiu. Catcharger: Deploying wireless charging lanes in a metropolitan road network through categorization and clustering of vehicle traffic. In *Proc. of IEEE INFOCOM*, 2017.
[22] R. Shokri, G. Theodorakopoulos, C. Troncoso, J. Hubaux, and J. L. Boudec. Protecting location privacy: Optimal strategy against localization attacks. In *Proc. of ACM CCS*, pages 617–627, 2012.
[23] F. S. Hillier. *Linear and Nonlinear Programming.* Stanford University, 2008.
[24] L. Bracciale et al. CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from https://crawdad.org/roma/taxi/20140717, July 2014.
[25] Chenxi Qiu, Anna Squicciarini, Zhuozhao Li, Ce Pang, and Li Yan. Time-Efficient Geo-Obfuscation to Protect Worker Location Privacy over Road Networks in Spatial Crowdsourcing. https://chenxiq1986.github.io/files/locationprivacy_techreport_CIKM.pdf.
[26] H. To, C. Shahabi, and L. Xiong. Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server. In *Proc. of IEEE ICDE*, 2018.
[27] R. Shokri et al. Quantifying location privacy. In *Proc. of IEEE S&P*, 2011.
[28] D. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE JSAC*, 24(8):1439–1451, Aug 2006.
[29] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, (8):101–111, 1960.
[30] N. Maculan, M. Passini, B. Moura, and I. Loiseau. Column-generation in integer linear programming. *RAIRO*, 37(2):67–83, 2003.
[31] J. Puchinger, P. Stuckey, M. Wallace, and S. Brand. Dantzig-wolfe decomposition and branch-and-price solving in g12. *Constraints*, 16(1):77–99, Jan 2011.
[32] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of ACM MobiSys*, 2003.
[33] L. Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):571–588, 2002.
[34] G. Theodorakopoulos et al. Prolonging the hide-and-seek game: Optimal trajectory privacy for location-based services. In *Proc. of ACM WPES*, 2014.