Location Privacy Protection in Vehicle-Based Spatial Crowdsourcing via Geo-Indistinguishability

Chenxi Qiu, *IEEE Member,* Anna Squicciarini, Ce Pang, Ning Wang, and Ben Wu

Abstract—Nowadays, vehicles have been increasingly adopted in many spatial crowdsourcing (SC) applications. Similar to other SC applications, location privacy is of great concern to vehicle workers as they are required to disclose their own location to servers to facilitate the service utilities. Traditional location privacy protection mechanisms cannot be applied to vehicle-based SC since they assume workers' mobility on a 2-dimensional plane without considering the network-constrained mobility features of vehicles. Accordingly, in this paper, we aim at addressing issues related to Vehicle-based spatial crowdsourcing Location Privacy (VLP) over road networks. Our objective is to design a location obfuscation strategy to minimize the quality loss due to obfuscation with *geo-indistinguishability* satisfied. Considering the computational complexity of VLP, by resorting to discretization, we first approximate VLP to a linear programming problem that can be solved by well-developed approaches. To further improve the time-efficiency, we conduct constraint reduction for VLP by exploiting key features of geo-indistinguishability in road networks and problem decomposition based on VLP's constraint structure. Finally, we carry out both trace-driven simulation and real-world experiments, where our experimental results demonstrate the superiority of our approach over a known state-of-the-art location obfuscation strategy in terms of both quality-of-service and privacy.

Index Terms—Spatial crowdsourcing; geo-indistinguishability, location privacy; location obfuscation.

1 Introduction

Spatial crowdsourcing (SC) [1] has emerged as a new mode of crowdsourcing to enable requesters to outsource their *spatial tasks* (i.e., tasks related to a location) to a set of mobile workers. In SC, task requesters register through a centralized *server* and publish tasks with target locations or spatial routes. If a worker accepts a task, he/she needs to physically present at the task location to perform the task. To date, SC has been applied to many different domains, such as smart cities [2] and environmental sensing [3]. Particularly, with the advent of intelligent transportation systems (ITS), *vehicle-based spatial crowdsourcing (VSC)* is evolving rapidly [4]. For example, many recent studies have proposed to use vehicle crowdsourcing workers as mobile agents to help maintain vehicle ad hoc networks (VANETs), for tasks such as data dissemination and query processing (e.g., [4]–[6]). In some other vehicle-related applications, VSC has been used for data sharing and collection [7], or to improve traditional transportation systems such as Uber [4].

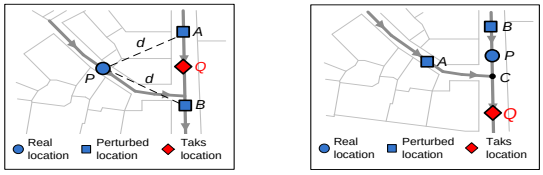
As workers have to physically move to tasks' location to complete tasks in VSC, a VSC server usually allocates tasks to the workers that have low *traveling cost* (e.g., traveling distance) in order to promote a cost-effective task assignment [8]–[10]. To this end, each worker is required to disclose his/her location to the server in real time. Such location information exposure, unfortunately, may lead to privacy breaches not only related to the whereabouts of the vehicle, but also related to other sensitive information, e.g., home address, sexual preferences, religious inclinations, etc [11].

In fact, location privacy research has gained great attention over the last decade and a variety of *location privacy protection mechanisms (LPPM)* have been developed, such as cryptographic strategies [12], [13], *k-anonymity* [14], [15], *cloaking* [16], [17], and *pseudonym* based methods [18]. Among these strategies, location obfuscation stands as one of the dominating LPPM paradigms in SC due to its high effectiveness of location privacy protection and low computation load for mobile devices [19]–[27]. In location obfuscation, each user is allowed to report a perturbed location instead of his/her real location to servers.

As a growing effort aims to address the location privacy issues via location obfuscation, a formal notion of location privacy, namely *Geo-Indistinguishability* (or *Geo-I*), was introduced by Andrés et al. recently [28]. Geo-I can be considered as a generalization of the statistical notion *differential privacy* [29]. According to Geo-I, if two locations are geographically close, they will have similar probabilities to generate a certain reported location. In the other words, the reported location will not provide enough information to an adversary to distinguish the true location among nearby ones. Subsequent to this notion of Geo-I, a variety of improved

• * Corresponding Author. Email: qiu@rowan.edu. Telephone number: (323) 3372758

Chenxi Qiu is with the Computer Science Department, Rowan University, Glassboro, NJ, 08028.
E-mail: qiu@rowan.edu.
Anna Squicciarini is with the College of Information Science and Technology, Pennsylvania State University, University Park, PA, 16801.
E-mail: asquicciarini@ist.psu.edu.
Ce Pang is with the Computer Science Department, Rowan University, Glassboro, NJ, 08028.
E-mail: pangc7@students.rowan.edu
Ning Wang is with the Computer Science Department, Rowan University, Glassboro, NJ, 08028.
E-mail: wangn@rowan.edu
Ben Wu is with the Electrical and Computer Engineering Department, Rowan University, Glassboro, NJ, 08028.
E-mail: wub@rowan.edu



(a) Example I (b) Example II

Fig. 1. Traveling cost distortion caused by obfuscated locations A and B (P: actual location; Q: task location).

Example I: A and B have the same Euclidean distance to Q, and hence they introduce the same traveling distance distortion from P on the 2D plane. However, both B and P need to take a detour to reach Q over roads, while A can reach Q with almost a straight path. In this case, the traveling cost distortion generated by A is much higher than that of B.

Example II: Compared with B, A has a shorter Euclidean distance to Q, indicating that A and B offer different traveling cost distortions from P on a 2D plane. However, A and B have the same traveling distance to Q over roads, so the traveling distance distortions generated by A and B over the road network are the same.

location obfuscation methods have been proposed [24]–[27]. Particularly, considering that users' location privacy is usually protected at the expense of *cost-effectiveness*, many of these works introduce optimization-based approaches (e.g., linear programming) to minimize the overall traveling cost for all the workers and still preserve privacy [24], [30].

However, most of these approaches consider workers' mobility in a 2 dimensional (2D) plane [31], where workers can move in every direction without any restriction [19], [20], [24]–[26], [28]. Clearly, such assumption is hardly to be applied in VSC as workers' mobility in VSC is more structured, i.e., vehicles have to operate over road networks and follow traffic regulations, which may disclose additional information to the adversary, and hence increase the risk of location exposure [32], [33]. More importantly, the approaches based on 2D is more likely to generate high traveling cost, as the sensitivity of traveling cost to location obfuscation over road networks varies much more significantly across different road segments compared with that of 2D (Fig. 1(a)(b) gives two examples).

Considering the different features of vehicle workers' mobility over a road network than in a 2D plane, in this paper, we aim to solve the *Vehicle based spatial Crowdsourcing Location Privacy (VLP)* problem over a *road network*. We model the road map by a *weighted directed graph* and assume that both workers' and tasks' are located on the graph. We consider a location obfuscation approach under which workers are allowed to report an obfuscated location instead of their true location, where the obfuscated location is probabilistically selected over the graph. Our objective is to determine the probability distribution of obfuscated location over the graph (so-called *location obfuscation strategy*), such that *1) the distortion of estimated traveling distance for each worker is minimized and 2) Geo-I is satisfied*. As for privacy, instead of adopting the Geo-I defined in [28], which is Euclidean distance based, we redefine the notion of Geo-I based on path distance in directed weighted graphs (details can be found in Definition 3.1). This correction, however, increases the complexity of VLP from an algorithmic perspective.

We start by discussing VLP in a general case, where the probability distribution of obfuscated location is defined in a continuous region. Considering the computational intractability of VLP, we then approximate the problem via discretization, i.e., each edge in the graph is partitioned into small intervals and the locations within each interval don't need to be differentiated. The approximated VLP, called *Discretized VLP* or *D-VLP*, can be then formulated as a *linear programming (LP)* problem. Nevertheless, D-VLP is hardly to be solved efficiently by using standard LP approaches due to its high number of Geo-I constraints ($O(K^3)$) and decision variables ($O(K^3)$), where K represents the number of intervals partitioned over the road network. As a solution, we first reduce the number of Geo-I constraints from $O(K^3)$ to $O(K^2)$ by exploiting the transitivity property of Geo-I over road network (Theorem 4.2). To further improve the time efficiency, by using the *angular block* structure of the constraint of D-VLP, we design a two-layer algorithm via optimization decomposition. The algorithm is composed of i) a master program (in the upper layer) to search the optimal solution and ii) a list of subproblems (in the lower layer) to check the optimality of the solution derived by the master program. The problems in both layers can be solved efficiently via standard LP methods and a near-optimal solution can be derived efficiently via the communication between the two levels. In addition, we provide a trade-off analysis between privacy and QoS in the D-VLP.

With respect to performance, we carry out both simulation (using a dataset of taxi cabs' trajectories in Rome, Italy [34]) and real-world experiment. The experimental results from both simulation and real-world test demonstrate that our approach outperforms a state-of-the-art location obfuscation mechanism [24] in terms of both quality loss and privacy. Specifically, compared with [24], in the simulation, our approach reduces the quality loss by 12.35% and increases the expected error from adversaries by 6.91%. In the real-world experiment, our method reduces the quality loss by 6.31% and increases the expected error from adversary by 9.68%. In addition, we show that D-VLP offers a reasonably good approximation with VLP in term of the quality loss in both simulation and real-world experiment.

We summarize our contributions as follows:
1) We formulated a new problem called *VLP* problem, of which the objective is to minimize the estimated traveling cost distortion without compromising location privacy.
2) Considering the computational intractability of VLP, we approximate VLP to a LP problem by discretization. To solve VLP in a time-efficient manner, we design a two-layer algorithm via optimization decomposition. We also provide a trade-off analysis between privacy and QoS in the VLP.
3) We conduct both simulation and real-world experiment to test the performance our approach. The experimental results demonstrate that our approach outperforms one existing 2D-based method in terms of both QoS and privacy and the two-layer algorithm can achieve the optimal closely with high time-efficiency.

The remainder of the paper is organized as follows: We introduce the model and the VLP problem in Section 3 and propose a time efficient solution in Section 4. In Section 5, we test the performance of our approach. Finally, we present related work in Section 6 and conclude in Section 7.

2 Framework

Fig. 2 shows the framework of our location obfuscation strategy in SC, which is composed of both worker and

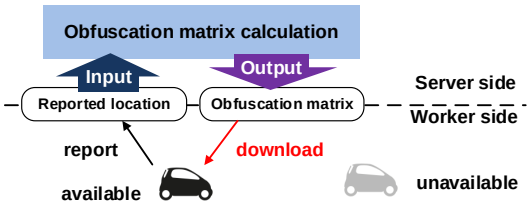


Fig. 2. Framework.

server sides. On the worker side, a worker can label his/her current status by either *available* or *unavailable*. 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, regardless of whether or not they are moving (driving). 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" on the platform.

To protect each worker's location privacy, instead of frequently requiring location updating, our framework only requires the worker to upload his/her obfuscated location before a snapshot of task assignment. Based on the worker's reported location, the server then determines which task to be assigned to this worker. The server is also responsible for generating an *obfuscation function (strategy)* that can be downloaded by the worker. With the obfuscation function, the worker takes his/her current location as the input and obtains a probability distribution of the obfuscated location as the output.

The obfuscation function is initialized when the system is set up. After initialization, the function is updated by the server based on the change of the worker's location distribution (estimated by the worker's reported location). Like [30], [35], we assume that the server may suffer from a *passive attack, where attackers can eavesdrop vehicles' reported locations breached by the server*. Note that although the server takes charge of generating the obfuscation function, the worker's location privacy is still guaranteed since 1) the obfuscated locations are problematically selected and 2) the obfuscation function is designed to satisfy the privacy criteria (Geo-I) even if the adversary obtains both workers' reported location and the obfuscation function from the server.

In a nutshell, the server determines the obfuscation function to achieve the following two objectives:
1) *Quality loss minimization*, i.e., to minimize the distortion of estimated distance based on obfuscated locations;
2) *Privacy guarantee*, i.e., geo-indistinguishability is satisfied.

Next, we will mathematically formalize the problem of generating obfuscation function (VLP) to achieve the above two objectives.

3 Model and Problem Formulation

In this section, we first introduce the model, including notations and assumptions that will be used throughout the paper in Section 3.1 and Section 3.2. Based on the model, we then formulate the VLP problem in Section 3.3.

3.1 Road Network Model

Like [30], [36], we represent the road network by a set of roads. When a road intersects, furcates, joins with other roads, or turns into a different direction, a connection is created (as shown in Fig. 3). These connections divide roads into multiple *road segments*, which only connect with other road segments at their end points. Accordingly, the road network can be represented by a *weighted directed graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the *connection set* and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the *route segment (edge) set*. Each edge $e \in \mathcal{E}$ is directed, presented by an ordered pair $(v_e^{\mathrm{s}}, v_e^{\mathrm{e}})$ $(v_e^{\mathrm{s}}, v_e^{\mathrm{e}} \in \mathcal{V})$, where v_e^{s} and v_e^{e} denote the starting and the ending connections of e, respectively. That is, vehicles can only move from v_e^{s} to v_e^{e} on e in the road network. Each e is allocated with a weight w_e, which can be interpreted as the shortest traveling distance [30], the lowest traveling time [37], or their combination [27]. In this paper, we consider traveling distance as the main factor for task assignment. In this case, w_e is defined as the traveling distance from v_e^{s} to v_e^{e}. Given different preferences from the requester, the model can be extended by redefining the weights of edges. For example, if the requester expect both low traveling distance and low latency, w_e can be redefined as a weighted sum of traveling distance and traveling time across the edge e.

To derive the traveling distance between the worker and the task, besides the road network information and the task location, the server also requires the worker to report his/her own location in real time. We assume both task and worker are located in the road network G, and let $\mathbf{p}$ and $\mathbf{q}$ denote the worker and the task's location, respectively. Each location point $\mathbf{p}$ (or $\mathbf{q}$) is represented by an ordered pair: $\mathbf{p} = (e, x)$, where e represents the edge that $\mathbf{p}$ is located in and x ($x \in (0, w_e]$) denotes the path distance from $\mathbf{p}$ to e's endpoint v_e^{e}.

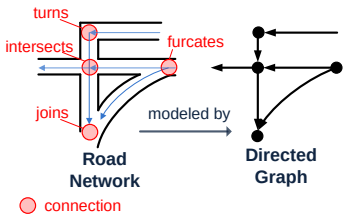


Fig. 3. Auxiliary graph.

TABLE 1
Main notations and definition

Notation	Description
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	The road network, where $\mathcal{V}$ and $\mathcal{E}$ denote its connection set and its edge set
w_e	The weight (length) of edge e
$\mathbf{p}$	The worker's true location
$\tilde{\mathbf{p}}$	The worker's obfuscated location
$\mathbf{q}$	The task location
v_e^{s} (v_e^{e})	The starting (ending) point of edge e
$e(\mathbf{p})$ $(e(\mathbf{q}))$	The edge that $\mathbf{p}$ $(\mathbf{q})$ is positioned in
$d_{\mathcal{G}}(\mathbf{v}, \mathbf{v}')$	The traveling distance from location $\mathbf{v}$ to location $\mathbf{v}'$ in $\mathcal{G}$ (in one direction)
$d_{\mathcal{G}}^{\min}(\mathbf{v}, \mathbf{v}')$	The ShTD between location $\mathbf{v}$ and location $\mathbf{v}'$ in $\mathcal{G}$ (in two directions)
$\Delta d_{\mathcal{G}}(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q})$	The difference between $d_{\mathcal{G}}(\mathbf{p}, \mathbf{q})$ and $d_{\mathcal{G}}(\tilde{\mathbf{p}}, \mathbf{q})$
$f_P(\mathbf{p})$ $(f_Q(\mathbf{q}))$	The prior PDF of $\mathbf{p}$ $(\mathbf{q})$
$f_{\tilde{P}}(\tilde{\mathbf{p}} \mid P = \mathbf{p})$	The conditional PDF of the obfuscated location $\tilde{\mathbf{p}}$ given the true location $\mathbf{p}$

Given any pair of locations $\mathbf{v}$ and $\mathbf{v}'$ in the road network $\mathcal{G}$, we let $d_{\mathcal{G}}(\mathbf{v}, \mathbf{v}')$ represent the *shortest traveling distance* (or *traveling distance* for simplicity) from $\mathbf{v}$ to $\mathbf{v}'$ in $\mathcal{G}$ (in one direction). Note that $d_{\mathcal{G}}(\mathbf{v}, \mathbf{v}')$ and $d_{\mathcal{G}}(\mathbf{v}', \mathbf{v})$ are possibly different since they measure the traveling distance of different paths. We let $d_{\mathcal{G}}^{\min}(\mathbf{v}, \mathbf{v}')$ denote the traveling distance between $\mathbf{v}$ and $\mathbf{v}'$ (in both directions):

$$d_{\mathcal{G}}^{\min}(\mathbf{v}, \mathbf{v}') = \min\{d_{\mathcal{G}}(\mathbf{v}, \mathbf{v}'), d_{\mathcal{G}}(\mathbf{v}', \mathbf{v})\}. \quad (1)$$

Table 1 lists the main notations and their descriptions used throughout this paper.

3.2 Threat Model

Like [28], [35], [38], instead of frequently requiring location reports from workers, our framework only requires workers to upload their obfuscated locations before a snapshot of task assignment. In such a scenario, the locations reported by each worker in different rounds are less correlated, which allows us to consider the inference attack in each round independently.

However, to check the performance of our privacy protection strategy when workers report their locations multiple times, we also develop a spatial correlation aware threat model by considering multiple reports from each worker.

3.2.1 Worker

To maintain location privacy, the worker will report an obfuscated location $\tilde{\mathbf{p}}$ instead of his/her true location $\mathbf{p}$, where $\tilde{\mathbf{p}}$ is probabilistically determined. We use random variables P and $\tilde{P}$ to represent the worker's true and obfuscated location, respectively. When reporting the obfuscated location, the worker's true location is given, i.e., $P = \mathbf{p}$, and hence the reported location distribution can be described by a conditional PDF $f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p})$, where

$$\sum_{e \in \mathcal{E}} \int_{[0, w_e]} f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p}) \mathrm{d}x = 1 \text{ (probability unit measure)}. \quad (2)$$

The *obfuscation function* is essentially the collection of conditional PDFs given all possible $\mathbf{p}$

$$\mathcal{F} = \{f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p}) | \mathbf{p} = (e, x), e \in \mathcal{E}, x \in (0, w_e]\}. \quad (3)$$

3.2.2 Adversary

When the vehicle reports its obfuscated location to the server, its true location is hidden from the attacker. However, the vehicle's obfuscated location is related to its true location by following a probability distribution determined by the vehicle's true location and the obfuscation matrix $\mathbf{Z}^t$. As such, the location of the targeted worker from the adversary can be estimated by a probabilistic model:

a) Bayesian inference attack (based on single location report). Given a reported location $\tilde{\mathbf{p}}$ from the worker, the adversary tries to find the true location $\mathbf{p}$ by calculating $\mathbf{p}$'s probability distribution. Here, we consider the scenario that the adversary has full information about the worker's obfuscation strategy $\mathcal{F}$ and the worker's prior PDF $f_P(\mathbf{p})$. Then, given the reported $\tilde{\mathbf{p}}$, the adversary can derive the PDF of the true location $\mathbf{p}$ by the Bayes' Theorem [39]:

$$f_P\left(\mathbf{p}|\tilde{P} = \tilde{\mathbf{p}}\right) = \frac{f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p}) f_P(\mathbf{p})}{\int_{\mathcal{G}} f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p}') f_P(\mathbf{p}') \mathrm{d}\mathbf{p}'} \quad (4)$$

b) Spatial correlation aware attack (based on multiple location reports). Due to the constraints of the road network environment and traffic conditions, if a vehicle reports its locations multiple times, each pair of adjacent location reports are possibly spatially correlated. The spatial correlation information can be possibly used by the adversary to improve the accuracy of location inference.

Specifically, from the adversary' point of view, *hidden Markov model (HMM)* offers a straightforward model to characterize the vehicles' network-constrained mobility feature. In each round t, the vehicle's true location P^t and reported location $\tilde{P}^t$ are considered as its *hidden* and *observable* states, respectively, where P^t follows a *Markov process*, i.e., P^t only depends on P^{t-1}. We let the transition matrix $\mathbf{H}^{t,t'} = \left\{h_{i,j}^{t,t'}\right\}_{K \times K}$ $(t < t')$ describe the transition probabilities between the vehicle's hidden states from round t to t', where $h_{i,j}^{t,t'} = \Pr\left(P^{t'} = \mathbf{p}_j | P^t = \mathbf{p}_i\right)$.

With HMM, given the observed sequence $\{\tilde{\mathbf{p}}^1, ..., \tilde{\mathbf{p}}^T\}$ and the transition matrix $\mathbf{H}^{1,2}, \mathbf{H}^{2,3}, ..., \mathbf{H}^{T-1,T}$, the task of the attacker is to derive the maximum likelihood estimate of the vehicle's true location sequence $\{\hat{\mathbf{p}}^1, ..., \hat{\mathbf{p}}^T\}$.

The Transition matrix in HMM can be learned via floating vehicle data [37]. The dataset usually records the vehicles' coordinates along with the timestamps. Given the dataset, we can calculate each the transition probability $h_{i,j}^{t-1,t}$ by

$$h_{i,j}^{t-1,t} = \frac{\text{\# of vehicles moving from } \mathbf{p}_i \text{ to } \mathbf{p}_j \text{ from round } t-1 \text{ to } t}{\text{\# of vehicles in } \mathbf{p}_i \text{ in round } t-1}. \quad (5)$$

After deriving the transition matrices in HMM, the attacker can use well-developed hidden state inference algorithms such as the Viterbi algorithm [40] to infer the vehicle's real location sequence $\{\hat{\mathbf{p}}^1, ..., \hat{\mathbf{p}}^T\}$.

3.3 Problem Formulation

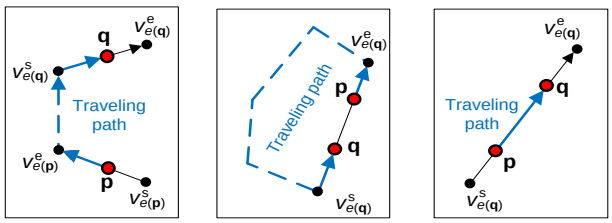
Before formulating the problem, we first define the two metrics that are considered for the location obfuscation strategy: *privacy* and *quality loss*.

Privacy. We aim to achieve *quasi-indistinguishability* or *Geo-Indistinguishability* (*Geo-I*) [28] for any pair of locations that are close to each other. Geo-I corresponds to a generalized version of the well-known concept of *differential privacy*. The idea of Geo-I on a 2D plane is to require a small change of a single user's location, measured by Euclidean distance, so as not to affect the distribution of his/her reported location too much. Following this idea, we redefine Geo-I on a weighted directed graph in Definition 3.1, where we measure the difference between any pair of locations by their traveling distance on graph, rather than their Euclidean distance. Particularly, for each pair of locations, we consider the traveling distance in both directions and pick up the shorter one as the measure of privacy.

Definition 3.1. A location obfuscation strategy satisfies (ϵ, r)-Geo-I if and only if for any pair of true locations $\mathbf{p}_i$ and $\mathbf{p}_l$ such that $d_{\mathcal{G}}^{\min}(\mathbf{p}_i, \mathbf{p}_l) \leq r$ and for any obfuscated location $\tilde{\mathbf{p}}$,

$$\frac{f_P\left(\mathbf{p}_i|\tilde{P} = \tilde{\mathbf{p}}\right)}{f_P\left(\mathbf{p}_l|\tilde{P} = \tilde{\mathbf{p}}\right)} \leq e^{\epsilon d_{\mathcal{G}}^{\min}(\mathbf{p}_i, \mathbf{p}_l)} \frac{f_P(\mathbf{p}_i)}{f_P(\mathbf{p}_l)}, \quad (6)$$

where r is the radius of the obfuscation area and ϵ is the parameter to quantify how much information of the true location will be disclosed according to the reported



(a) C1 (b) C2 (c) C3

Fig. 4. Derivation of $d_{\mathcal{G}}(\mathbf{p}, \mathbf{q})$ under three cases.

location, where higher ϵ implies more information to be disclosed.

According to Equ. (4), Equ. (6) can be rewritten as

$$f_{\tilde{P}}\left(\tilde{\mathbf{p}}|P=\mathbf{p}_i\right) \leq e^{\epsilon d_{\mathcal{G}}^{\min}(\mathbf{p}_i,\mathbf{p}_l)} f_{\tilde{P}}\left(\tilde{\mathbf{p}}|P=\mathbf{p}_l\right). \quad (7)$$

Quality loss. Given the worker's obfuscated location $\tilde{\mathbf{p}}$, his/her true location $\mathbf{p}$, and the task location $\mathbf{q}$, we measure the quality loss by the *estimated traveling distance distortion (ETDD)* to the task location $\mathbf{q}$, which, more precisely, is defined as the difference between the estimated traveling distance $d_{\mathcal{G}}(\tilde{\mathbf{p}}, \mathbf{q})$ and the true traveling distance $d_{\mathcal{G}}(\mathbf{p}, \mathbf{q})$:

$$\Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right) = \left|d_{\mathcal{G}}(\mathbf{p}, \mathbf{q}) - d_{\mathcal{G}}(\tilde{\mathbf{p}}, \mathbf{q})\right|. \quad (8)$$

As opposed to 2D plane, the quality loss in a vehicle road map is highly impacted by the topology of the network. We derive $d_{\mathcal{G}}(\mathbf{p}, \mathbf{q})$ by considering the following two cases (we use $e(\mathbf{q})$ and $e(\mathbf{p})$ to represent the edges that $\mathbf{q}$ and $\mathbf{p}$ are located in).

C1: When $e(\mathbf{q}) \neq e(\mathbf{p})$ (Fig. 4(a)), or when $e(\mathbf{q}) = e(\mathbf{p})$ but $\mathbf{p}$ has shorter traveling distance to its endpoint of $e(\mathbf{p})$, $v^{\mathrm{e}}_{e(\mathbf{p})}$, than $\mathbf{q}$ (Fig. 4(b)):

In this case, the worker's traveling path has to first reach the current edge's endpoint $v^{\mathrm{e}}_{e(\mathbf{p})}$, then the starting point of $\mathbf{q}$'s edge, $v^{\mathrm{s}}_{e(\mathbf{q})}$, and finally the destination location $\mathbf{q}$. Hence, the traveling distance from $\mathbf{p}$ to $\mathbf{q}$ is calculated by

$$\begin{aligned} d_{\mathcal{G}}(\mathbf{p}, \mathbf{q}) &= d_{\mathcal{G}}\left(\mathbf{p}, v^{\mathrm{e}}_{e(\mathbf{p})}\right) + d_{\mathcal{G}}\left(v^{\mathrm{e}}_{e(\mathbf{p})}, v^{\mathrm{s}}_{e(\mathbf{q})}\right) + d_{\mathcal{G}}\left(v^{\mathrm{s}}_{e(\mathbf{q})}, \mathbf{q}\right) \\ &= d_{\mathcal{G}}\left(v^{\mathrm{e}}_{e(\mathbf{p})}, v^{\mathrm{s}}_{e(\mathbf{q})}\right) + x_{\mathbf{p}} + l_{e(\mathbf{q})} - x_{\mathbf{q}}. \end{aligned} \quad (9)$$

C2: When $e(\mathbf{q}) = e(\mathbf{p})$ and $\mathbf{p}$ has longer path distance to the edge's endpoint $v^{\mathrm{e}}_{e(\mathbf{p})}$ than $\mathbf{q}$ (Fig. 4(c)), the traveling distance from $\mathbf{p}$ to $\mathbf{q}$ is

$$d_{\mathcal{G}}\left(\mathbf{p}, \mathbf{q}\right) = d_{\mathcal{G}}\left(\mathbf{p}, v^{\mathrm{e}}_{e(\mathbf{p})}\right) - d_{\mathcal{G}}\left(\mathbf{q}, v^{\mathrm{e}}_{e(\mathbf{p})}\right) = x_{\mathbf{p}} - x_{\mathbf{q}}. \quad (10)$$

A similar derivation can be applied to $d_{\mathcal{G}}(\tilde{\mathbf{p}}, \mathbf{q})$. Considering the possible $(\mathbf{p}, \mathbf{q})$ (and $(\tilde{\mathbf{p}}, \mathbf{q})$) in the above two cases, we can derive $\Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right)$ as

$$\begin{aligned} &\Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right) \quad (11) \\ &= \begin{cases} \left|d_{\mathcal{G}}(v^{\mathrm{e}}_{e(\mathbf{p})}, v^{\mathrm{s}}_{e(\mathbf{q})}) - d_{\mathcal{G}}(v^{\mathrm{e}}_{e(\tilde{\mathbf{p}})}, v^{\mathrm{s}}_{e(\mathbf{q})}) + x_{\mathbf{p}} - x_{\tilde{\mathbf{p}}}\right| & C(1,1) \\ \left|d_{\mathcal{G}}(v^{\mathrm{e}}_{e(\mathbf{p})}, v^{\mathrm{s}}_{e(\mathbf{q})}) + x_{\mathbf{p}} + l_{e(\mathbf{q})} - x_{\tilde{\mathbf{p}}}\right| & C(1,2) \\ \left|d_{\mathcal{G}}(v^{\mathrm{e}}_{e(\tilde{\mathbf{p}})}, v^{\mathrm{s}}_{e(\mathbf{q})}) + x_{\tilde{\mathbf{p}}} + l_{e(\mathbf{q})} - x_{\mathbf{p}}\right| & C(2,1) \\ \left|x_{\tilde{\mathbf{p}}} - x_{\mathbf{p}}\right| & C(2,2) \end{cases}, \end{aligned}$$

where $C(i, j)$ represents when $(\mathbf{p}, \mathbf{q})$ is in case i and $(\tilde{\mathbf{p}}, \mathbf{q})$ is in case j.

In addition, we assume that the task location $\mathbf{q}$ won't be exposed to the worker before the worker selects his/her obfuscated location. The worker however has the prior distribution of the task, $f_Q(\mathbf{q})$, based on the historical record, where Q denotes the random variable to describe the task location. Then, given the location obfuscation approach $\mathcal{F}$, the worker can obtain ETDD (quality loss):

$$\begin{aligned} &\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right) \quad (12) \\ &= \int\int\int \Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right) f_Q(\mathbf{q}) f_P(\mathbf{p}) f_{\tilde{P}}(\tilde{\mathbf{p}}|P=\mathbf{p}) \mathrm{d}\tilde{\mathbf{p}}\mathrm{d}\mathbf{q}\mathrm{d}\mathbf{p} \end{aligned}$$

Problem Formulation. Based on the definition of Geo-I (Equ. (7)) and the quality loss (Equ. (12)), the probability unit measure (Equ. (2)), we formulate the *VSC Location privacy Protection (VLP)* problem as:

$$\begin{aligned} \min \quad & \mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right) & (13) \\ \text{s.t.} \quad & f_{\tilde{P}}\left(\tilde{\mathbf{p}}|P=\mathbf{p}_i\right) \leq e^{\epsilon d_{\mathcal{G}}^{\min}(\mathbf{p}_i,\mathbf{p}_l)} f_{\tilde{P}}\left(\tilde{\mathbf{p}}|P=\mathbf{p}_l\right), \\ & \forall \mathbf{p}_i, \mathbf{p}_l, \tilde{\mathbf{p}} \text{ in } \mathcal{G}, \text{ with } d_{\mathcal{G}}^{\min}(\mathbf{p}_i, \mathbf{p}_l) \leq r, & (14) \\ & \sum_{e\in\mathcal{E}} \int_{[0,w_e]} f_{\tilde{P}}(\tilde{\mathbf{p}}|P=\mathbf{p}) \mathrm{d}x = 1, \forall \mathbf{p}_i \in \mathcal{G}. & (15) \end{aligned}$$

The objective of VLP is to determine each location obfuscation strategy $f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p})$ in $\mathcal{F}$ such that ETDD is minimized (Equ. (13)) and (ϵ, r)-Geo-I is satisfied (Equ. (14)).

In VLP, although our target is to minimize the error of estimated traveling distance for a single vehicle, achieving such goal can also reduce the overall traveling cost for multiple vehicle-task assignment. Given accurate estimated traveling cost for every single vehicle, the server is more likely to assign tasks to their nearest vehicles and hence achieve a lower total traveling cost.

However, finding the optimal $f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p})$ is computational intractable as it is difficult to describe $f_{\tilde{P}}(\tilde{\mathbf{p}}|P = \mathbf{p})$ (a general continuous function) with finite number of decision variables. As workers are mostly highly dynamic, and hence are required to update their location information in a timely fashion, it is of great importance to find a location obfuscation strategy that can *achieve near-minimum quality loss with low time complexity*.

TABLE 2
Additional notations and definition in Section 4

Notation	Description
$\mathbf{u}_k$	The kth interval partitioned in $\mathcal{G}$
$\mathbf{u}^{\mathrm{s}}_k$ ($\mathbf{u}^{\mathrm{e}}_k$)	The starting (ending) point of $\mathbf{u}_k$
$\mathcal{U}$	The set of intervals $\mathcal{U} = \{\mathbf{u}_1, ..., \mathbf{u}_K\}$
δ	The length of each partitioned interval $\mathbf{u}_k$
$\delta(\mathbf{p})$	The relative location of $\mathbf{p}$
$\mathcal{G}'$ $= (\mathcal{U}', \mathcal{E}')$	The auxiliary graph describing the interval set $\mathcal{U}$, where $\mathcal{U}'$ corresponds $\mathcal{U}$ and $\mathcal{E}'$ corresponds the distance between adjacent intervals in $\mathcal{U}$
$\mathbf{u}'_k$	The vertex corresponding to $\mathbf{u}_k$ in $\mathcal{U}'$
$\overset{G(\epsilon,r)}{\simeq}$	$\mathbf{u}'_i \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_l$ if Geo-I is satisfied in the direction from $\mathbf{u}'_i$ to $\mathbf{u}'_l$ (Definition 4.2)

4 Algorithm Design and Analysis

In this section, we aim to design a time efficient algorithm for VLP. The basic idea is to approximate VLP to a LP problem via discretization (Section 4.1). After that, by exploring key features of Geo-I over road networks, we design an approach that can further reduce the complexity of the

descretized VLP via constraint reduction (Section 4.2) and optimization decomposition (Section 4.3). Table 2 lists the additional notations used in this section.

4.1 Problem Approximation

The obfuscation function in VLP is defined in a continuous region, i.e., the road network, which cannot be represented by finite number of decision variables. As a solution, we approximate VLP by discretization, in which we only need to consider the obfuscated location probability in intervals instead of in a continuous region. We denote the discretized problem by *discretized-VLP* or *D-VLP*. More precisely, we formulate D-VLP from VLP by the following three steps:

Step I: Each edge is partitioned into route intervals with length δ (as depicted in Fig. 5). We let $\mathcal{U} = \{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_K\}$ ($K = |\mathcal{U}|$) denotes the set of intervals in the road network $\mathcal{G}$, and let $\mathbf{u}_k^{\mathrm{s}} = \left(e, x_{\mathbf{u}_k}^{\mathrm{s}}\right)$ and $\mathbf{u}_k^{\mathrm{e}} = \left(e, x_{\mathbf{u}_k}^{\mathrm{e}}\right)$ denote the two endpoints of each $\mathbf{u}_k$ (in edge e), where $x_{\mathbf{u}_k}^{\mathrm{s}} - x_{\mathbf{u}_k}^{\mathrm{e}} = \delta$. For a true location $\mathbf{p} = (e, x)$ that is in $\mathbf{u}_k$, we call $\delta(\mathbf{p}) = x - x_{\mathbf{u}_k}^{\mathrm{e}}$ the *relative location* of $\mathbf{p}$ in $\mathbf{u}_k$ ($0 \le \delta(\mathbf{p}) \le \delta$) [1].

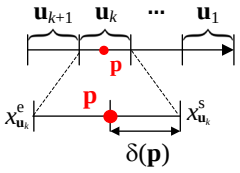


Fig. 5. Edge partition.

Step II: The obfuscated location $\tilde{\mathbf{p}}$ is required to have the same relative location with its true location $\mathbf{p}$, i.e., $\delta(\mathbf{p}) = \delta(\tilde{\mathbf{p}})$ whichever interval is $\tilde{\mathbf{p}}$ in.

Step III: For any pair of true locations $\mathbf{p}_1$ and $\mathbf{p}_2$ that are in the same interval $\mathbf{u}_i$, the probabilities of their obfuscated location $\tilde{\mathbf{p}}_1$ and $\tilde{\mathbf{p}}_2$ in each interval $\mathbf{u}_l$ are the same, i.e.,

$$\Pr(\tilde{\mathbf{p}}_1 \in \mathbf{u}_l | P = \mathbf{p}_1) = \Pr(\tilde{\mathbf{p}}_2 \in \mathbf{u}_l | P = \mathbf{p}_2) \quad (16)$$

where $l = 1, ..., K$.

We note that Step II and Step III introduce additional constraints to VLP, and hence they shrink the feasible region of VLP [41], indicating that the optimal solution of D-VLP offers an *upper bound* of the minimum quality loss in VLP.

Proposition 4.1. Suppose that a pair of true locations $\mathbf{p}_1$ and $\mathbf{p}_2$ are in the same interval $\mathbf{u}_i$. Then, in D-VLP:

A) Given any task location $\mathbf{q}$, $\mathbf{p}_1$ and $\mathbf{p}_2$ have the same ETDD to $\mathbf{q}$, i.e.,

$$\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(\mathbf{p}_1, \tilde{P}_1; \mathbf{q}\right)\right) = \mathbf{E}\left(\Delta d_{\mathcal{G}}\left(\mathbf{p}_2, \tilde{P}_2; \mathbf{q}\right)\right). \quad (17)$$

B) A location obfuscation function satisfies (ϵ, r)-Geo-I for $\mathbf{p}_1$ if only if it satisfies the constraint for $\mathbf{p}_2$.

Proof The detailed proof can be found in [30].

Proposition 4.1 indicates that there is no need to differentiate any pair of true locations within the same interval when calculating quality loss or checking Geo-I. Accordingly, we can rewrite the objective function in VLP (Equ. (13)) based on Proposition 4.1-A:

$$\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right) = \sum_i \sum_l c_{i,l} z_{i,l} \quad (18)$$

1. Due to the variety of edge length, there exists some intervals with length smaller than δ. But as δ is small enough, we won't discuss these intervals in the following part considering the tractability of our solution.

where $z_{i,l}$ represents the probability that the obfuscated location $\tilde{\mathbf{p}}$ is in $\mathbf{u}_l$ given the true location $\mathbf{p}$ in $\mathbf{u}_i$ and

$$c_{i,l} = \int_{\mathbf{u}_i} \int_{\mathbf{u}_l} \int \Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right) f_Q(\mathbf{q}) f_P(\mathbf{p}) \mathrm{d}\mathbf{q} \mathrm{d}\tilde{\mathbf{p}} \mathrm{d}\mathbf{p} \quad (19)$$

is a constant (note that $\Delta d_{\mathcal{G}}\left(\mathbf{p}, \tilde{\mathbf{p}}; \mathbf{q}\right)$, $f_Q(\mathbf{q})$, $f_P(\mathbf{p})$ are all known). According to Proposition 4.1-B, the Geo-I constraint in VLP (Equ. (14)) can be rewritten by

$$z_{i,j} - e^{\epsilon d_{\mathcal{G}}^{\min}(\mathbf{u}_i^{\mathrm{e}}, \mathbf{u}_l^{\mathrm{e}})} z_{l,j} \le 0, \forall \mathbf{u}_i, \mathbf{u}_j, \mathbf{u}_l \text{ s.t. } d_{\mathcal{G}}^{\min}(\mathbf{u}_i, \mathbf{u}_l) \le r \quad (20)$$

where $d_{\mathcal{G}}^{\min}(\mathbf{u}_i, \mathbf{u}_l) = d_{\mathcal{G}}^{\min}(\mathbf{u}_i^{\mathrm{s}}, \mathbf{u}_l^{\mathrm{e}})$, and the probability unit measure (Equ. (15)) can be rewritten by

$$\textstyle\sum_j z_{i,j} = 1, \ \forall i \quad (21)$$

Eventually, D-VLP can be written as a LP problem:

$$\min \sum_i \sum_l c_{i,l} z_{i,j} \ \text{ s.t. Equations (20)-(21) is satisfied.}$$

where the decision variables are $\mathbf{Z} = \{z_{i,j}\}_{K \times K}$. For simplicity, we let $\mathbf{z}_j = [z_{1,j}, ..., z_{K,j}]$ ($j = 1, ..., K$). In [30] (Proposition 3.3), we have derived a lower bound of the minimum quality loss in VLP to check how close the solution of D-VLP can achieve the optimal of VLP (e.g., experimental results are shown in Fig. 10(a)(b)).

4.2 Constraint Reduction

According to the definition of (ϵ, r)-Geo-I (Equ. (20)), given any obfuscated interval $\mathbf{u}_j$ ($j = 1, ..., K$), we need to set a constraint for each pair of $z_{i,j}$ and $z_{l,j}$ ($i, l = 1, ..., K$), which generates $O(K^3)$ constraints to D-VLP in total. Although LP is solvable by many existing approaches, it is crucial to reduce the huge number of constraints, which highly affects the time-efficiency for solving the LP problem [41].

Fortunately, there are some features of partitioned intervals in road networks that can be exploited to reduce the number of inequality constraints in D-VLP. Along these features, we find that, to constrain all pairs of intervals partitioned in the road network, it is sufficient to apply Geo-I to pairs of intervals that are "adjacent" (Definition 4.1), achieving *constraint reduction*.

Before describing the constraint reduction, we first introduce Definition 4.1–4.2, Property 4.1, and Theorem 4.2.

Definition 4.1. (*Auxiliary graph*) We build a weighted directed auxiliary graph $\mathcal{G}' = (\mathcal{U}', \mathcal{E}')$, where the vertex set $\mathcal{U}' = \{\mathbf{u}_1', \mathbf{u}_2', ..., \mathbf{u}_{|\mathcal{U}|}'\}$ corresponds $\mathcal{U}$, and if any pair of $\mathbf{u}_i$ and $\mathbf{u}_l$ are adjacent in the road network and the worker can directly travel from $\mathbf{u}_i$ to $\mathbf{u}_l$, then we build a directed edge from $\mathbf{u}_i'$ to $\mathbf{u}_l'$ with weight δ in $\mathcal{G}'$ (Fig. 6 gives an example).

The auxiliary graph $\mathcal{G}'$ is used to describe the relationship among intervals in $\mathcal{U}$, where the traveling distance between any pair of vertices, say $\mathbf{u}_i'$ and $\mathbf{u}_l'$, equals to the traveling distance between the corresponding intervals $\mathbf{u}_i$ and $\mathbf{u}_l$. Accordingly, checking Geo-I between $\mathbf{u}_i$ and $\mathbf{u}_l$ is equivalent to checking Geo-I for $\mathbf{u}_i'$ and $\mathbf{u}_l'$. With the auxiliary graph, we can directly

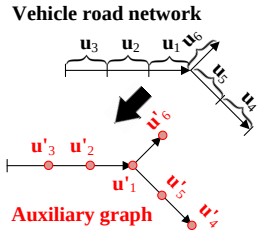


Fig. 6. Auxiliary graph.

apply some existing data structures (e.g., shortest path trees) to help implement the constraint reduction.

Definition 4.2. We use $\mathbf{u}'_i \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_l$ to denote that Geo-I is satisfied in the direction from $\mathbf{u}'_i$ to $\mathbf{u}'_l$. More precisely, $\mathbf{u}'_i \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_l$ if $z_{i,j} - e^{\epsilon d_{\mathcal{G}}(\mathbf{u}^e_i, \mathbf{u}^e_l)} z_{l,j} \leq 0,\ \forall \mathbf{u}_j$.

According to Definition 3.1 and Definition 4.2, it is trivial to obtain Property 4.1:

Property 4.1. $\mathbf{u}'_i$ and $\mathbf{u}'_l$ satisfies (ϵ, r)-Geo-I constraint if only if $\mathbf{u}'_i \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_l$ and $\mathbf{u}'_l \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_i$.

In Theorem 4.2, we introduce the *transitivity* of Geo-I over road networks:

Theorem 4.2. (*Transitivity*) Given any pair of vertices $\mathbf{u}'_1$ and $\mathbf{u}'_K$, suppose that a *shortest path* from $\mathbf{u}'_1$ to $\mathbf{u}'_K$ is composed of $K-1$ edges in $\mathcal{G}'$: $(\mathbf{u}'_1, \mathbf{u}'_2) \rightarrow ... \rightarrow (\mathbf{u}'_{K-1}, \mathbf{u}'_K)$, then

$$\mathbf{u}'_k \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_{k+1},\ (k = 1, ..., K-1) \Rightarrow \mathbf{u}'_1 \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_K \quad (22)$$

Proof The detailed proof can be found in [30].

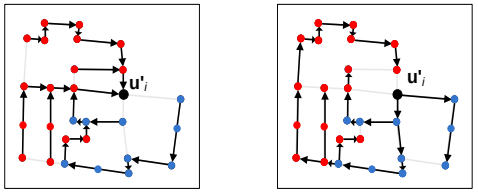
According to Theorem 4.2, to provide a sufficient condition of the Geo-I constraint for any pair of vertices $\mathbf{u}'_i$ and $\mathbf{u}'_l$ in $\mathcal{U}'$, we can first 1) find the shortest path between $\mathbf{u}'_i$ and $\mathbf{u}'_l$ in both directions (from $\mathbf{u}'_i$ to $\mathbf{u}'_l$ and from $\mathbf{u}'_l$ to $\mathbf{u}'_i$), and then 2) select the shortest path between the two paths, say $\mathcal{P}$, and set the Geo-I constraint $\mathbf{u}'_k \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_{k+1}$ for each pair of adjacent vertices $\mathbf{u}'_k$ and $\mathbf{u}'_{k+1}$ in $\mathcal{P}$. We repeat such process for all pairs of vertices in $\mathcal{G}'$.

In the above process of constraint reduction, any constraint constructed by adjacent vertices, say $\mathbf{u}'_k \overset{G(\epsilon,r)}{\simeq} \mathbf{u}'_{k+1}$, won't shrink the feasible region of the D-VLP, since $\mathbf{u}'_k$ and $\mathbf{u}'_{k+1}$ themselves also need to satisfy the Geo-I constraint (according to Property 4.1). Hence, *the optimality of D-VLP will not be lost by the constraint reduction.*

Since each pair of adjacent vertices in $\mathcal{P}$ must be the two endpoints of an edge in $\mathcal{G}'$, the number of possible adjacent vertices in all shortest paths cannot exceed M (M denotes the number of edges in the auxiliary graph $\mathcal{G}'$, i.e., $M = |\mathcal{E}'|$). For each obfuscated vertex $\mathbf{u}'_j$ ($j = 1, ..., K$) (i.e., the obfuscated location is in $\mathbf{u}_j$), instead of building constraint for each pair of vertices in $\mathcal{U}'$, we only need to build up to M constraints for the pairs that are adjacent. Hence, the total number of constraints to be added is $O(KM)$. Since the auxiliary graph $\mathcal{G}'$ is close to a planar graph in the real-world, M is close to K. Accordingly, the number of constraints in D-VLP can be reduced from $O(K^3)$ to approximately $O(K^2)$.

Algorithm 1 provides the pseudo code of our constraint reduction method: We use an indicator matrix $\mathbf{U}_{\text{con}} = \{u_{i,j}\}_{K \times K}$ to represent whether a constraint for $\mathbf{u}'_i$ and $\mathbf{u}'_j$ is added: if the pair $\{\mathbf{u}'_i, \mathbf{u}'_j\}$ needs a constraint, $u_{i,j} = 1$; otherwise, $u_{i,j} = 0$. To find the shortest path between all pairs of vertices in $\mathcal{G}'$, we build the *shortest path trees (SPTs)* [42] rooted at each vertex $\mathbf{u}'_i$ ($i = 1, ..., |\mathcal{U}|$), respectively.

We note that the shortest path between any pair of vertices can be in both directions and we only need to check Geo-I for the shorter one. Here, for each $\mathbf{u}'_i$, we build two SPTs: SPT-Out(i) and SPT-In(i) (as shown in Fig. 7(a)(b)), in



(a) SPT-Out(i) (b) SPT-In(i)

Fig. 7. Example: the two types of SPTs for the vertex $\mathbf{u}'_i$ ($\mathcal{U}'_{\text{In},i}$ and $\mathcal{U}'_{\text{Out},i}$ are respectively marked by red color and blue color).

which all the paths take $\mathbf{u}'_i$ as the source and the destination, respectively (line 4). After building a SPT (can be either SPT-Out(i) or SPT-In(i)), it is unnecessary to find the path for each vertex to (or from) $\mathbf{u}'_i$ in the tree, since the vertex may have a shorter path with $\mathbf{u}'_i$ in the other tree. Hence, before finding the paths, we categorize all the vertices in $\mathcal{U}' \backslash \mathbf{u}'_i$ into two subsets $\mathcal{U}'_{\text{In},i}$ and $\mathcal{U}'_{\text{Out},i}$ based on whether each vertex in $\mathcal{U}' \backslash \mathbf{u}'_i$ has a shorter path to (or from) $\mathbf{u}'_i$ in SPT-In(i) than in SPT-Out(i) (line 5-9). After the vertex categorization, in SPT-In(i), all the paths from the vertices in $\mathcal{U}'_{\text{In},i}$ to $\mathbf{u}'_i$ are collected; and in SPT-Out(i), all the paths from $\mathbf{u}'_i$ to the vertices in $\mathcal{U}'_{\text{Out},i}$ are collected. Finally, for each pair of adjacent vertices in the collected paths, we add the corresponding constraint to D-VLP (line 10–13).

Algorithm 1: Pseudo-code of constraint reduction.

```
   input  : G = (V, E)
   output : U_con = {u_{i,j}}_{|U|×|U|}
1  Initialize U_con by 0;
2  Initialize the sets U'_{In,1}, ..., U'_{In,|U|}, U'_{Out,1}, ..., U'_{Out,|U|} by
     empty;
3  for each u_i ∈ V do
4      Build both SPT-In(i) and SPT-Out(i);
5      for each u_j ∈ V \ u_i do
6          if ShPD(u_i, u_j) ≤ ShPD(u_j, u_i) then
7              add u'_j to U'_{Out,i};
8          otherwise do
9              add u'_j to U'_{In,i};
10     for each u'_j ∈ U'_{Out,i} do
11         Traverse the edges in the path from u'_j to u'_i and let
             u_{l,k} = 1 if (u'_l, u'_k) is an edge in the path;
12     for each u'_j ∈ U'_{I,i} do
13         Traverse the edges in the path from u'_j to u'_i and let
             u_{l,k} = 1 if (u'_l, u'_k) is an edge in the path;
14 return U_con;
```

Time complexity analysis of the constraint reduction. The constraint reduction mainly includes SPT building (line 4), vertex categorization (line 5–9), and constraint addition (line 10–13) for each $\mathbf{u}'_i$ ($i = 1, ..., |\mathcal{U}|$). We adopt a well-developed method Dijkstra [42] to build the SPTs, of which the time complexity is $O(M + K \log K)$. Vertex categorization requires to compare the length of each vertex's two paths with $\mathbf{u}'_i$, taking up to K comparisons. Constraint addition requires to traverse all the edges in the two SPTs, both of which have up to K edges. Eventually, the time complexity of the constraint reduction method can be calculated by

$$\begin{aligned} T_{\text{cr}} &= O(K) \times (\underbrace{O(M + K \log K)}_{\text{line 4}} + \underbrace{O(K)}_{\text{line 5-9}} + \underbrace{O(K)}_{\text{line 10–13}}) \\ &= O\left(MK + K^2 \log K + K^2\right). \end{aligned} \quad (23)$$

4.3 Optimization Decomposition

With the constraint reduction in Section 4.2, the number of constraints is reduced from cubic to approximately quadratic. Nevertheless, the computation load is still extremely high, e.g., thousands of discrete locations will generate millions of decision variables in D-VLP. To further improve the time efficiency of the algorithm, we adopt *decomposition* techniques to decompose D-VLP to smaller problems that can be processed in parallel [43]. We find that in D-VLP, 1) the Geo-I constraints for each $\mathbf{z}_j = [z_{1,j}, ..., z_{K,j}]$ are all disjoint; 2) only the constraints of the probability unit measure (Equ. (21)) link together the different decision vectors $\mathbf{z}_1$, ..., $\mathbf{z}_K$. With such constraint structure, D-VLP is well-suited to *Dantzig-Wolfe (DW) decomposition* [44].

4.3.1 Dantzig-Wolfe (DW) formulation

Since all the Geo-I constraints are linear, the Geo-I constraints for each decision vector $\mathbf{z}_l$ $(l = 1, ..., K)$ define a polyhedron Λ_l in a K dimensional space. We let $\mathcal{Z}_l = \left\{\hat{\mathbf{z}}_l^1, ..., \hat{\mathbf{z}}_l^{T_l}\right\}$ denote the set of extreme points of Λ_l. Then, any decision vector $\mathbf{z}_l \in \Lambda_l$ can be represented as a convex combination of $\hat{\mathbf{z}}_l^1, ..., \hat{\mathbf{z}}_l^{T_l}$, i.e., $\mathbf{z}_l = \sum_{t=1}^{T_l} \lambda_{l,t} \hat{\mathbf{z}}_l^t$, where $\sum_{t=1}^{T_l} \lambda_{l,t} = 1, \lambda_{l,t} \geq 0$. Accordingly, D-VLP can be rewritten as the following DW formulation:

$$\min \quad \sum_l \sum_i c_{i,l} \sum_t \lambda_{l,t} \hat{z}_{i,l}^t \tag{24}$$

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

where $\lambda_{l,t}$ $(t = 1, ..., T_l, l = 1, ..., K)$ are the decision variables in the DW formulation.

4.3.2 Column Generation (CG) Algorithm

The number of decision variables (extreme points) in the DW formulation is exponential to K, so the DW formulation itself does not improve the time-efficiency if we directly solve it standard LP approaches. According to [45], most extreme points in the DW formulation are inactive, i.e., are not visited during the searching process in the algorithms such as the revised simplex method [41]. As such, we apply CG [46] to solve D-VLP in the DW formulation, which is composed of two steps (Fig. 8 shows the framework of CG):

Step I (Initialization): We start with a *reduced DW formulation (RDW)*, where only a subset of extreme points $\overline{\mathcal{Z}}_1, ..., \overline{\mathcal{Z}}_K$ $(\overline{\mathcal{Z}}_l \subseteq \mathcal{Z}_l (l = 1, ..., K))$ are considered in the DW formulation. We use $\overline{\boldsymbol{\lambda}}^*$ to represent the solution of RDW. When selecting $\overline{\mathcal{Z}}_1, ..., \overline{\mathcal{Z}}_K$, we need to ensure the feasible region of RDW to be non-empty[2] for the sake of convergence of CG [46]. Note that $\overline{\boldsymbol{\lambda}}^*$ derived from RDW does not necessarily achieve the optimal of the DW formulation. Hence, we need to test $\overline{\boldsymbol{\lambda}}^*$'s optimality in DW in **Step II**.

Step II (Optimal test): We first introduce Proposition 4.3 for the optimal test:

Proposition 4.3. $\overline{\boldsymbol{\lambda}}^*$ achieves the optimal if only if $(\overline{\boldsymbol{\pi}}^*, \overline{\boldsymbol{\mu}}^*)$ is dual feasible for the DW formulation, i.e., $\forall l = 1, ..., K$,

$$\mathsf{sub}_1: \ \min_{t \in \mathcal{Z}_l} \left\{ \sum_k \hat{z}_{k,l}^t \overline{\pi}_k^* + \overline{\mu}_l^* + \sum_i c_{i,l} \hat{z}_{i,l}^t \right\} \geq 0. \tag{25}$$

2. Here, we initialize each $\overline{\mathcal{Z}}_l^{(1)}$ by the extreme point $\mathbf{e}_l$, where $\mathbf{e}_l$ is a $K+1$ dimension vector with the lth entry equal to 1 and all the other entries equal to 0.

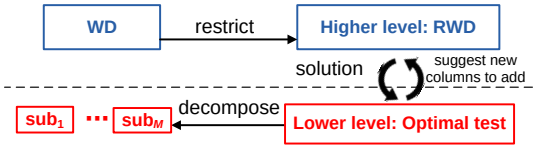


Fig. 8. Frame of the column generation algorithm.

Proof The detailed proof can be found in [47].

Note that checking each sub_l in Equ. (25) is essentially to solve a LP problem with K decision variables, i.e., $\mathbf{z}_l = [z_{1,l}, ..., z_{K,l}]$. The decision variables in sub_1, ..., sub_K are fully decoupled, and hence sub_1, ..., sub_K can be solved in parallel. In sub_l, there exists an extreme point $\mathbf{z}_l' = [z_{1,l}', ..., z_{K,l}']$ such that

$$\sum_k z_{k,l}' \overline{\pi}_k^* + \overline{\mu}_l^* + \sum_i c_{i,l} z_{i,l}' < 0, \tag{26}$$

then $\overline{\boldsymbol{\lambda}}^*$ has not achieved the optimal and we need to add the extreme point (column) $\mathbf{z}_l'$ to the RDW to improve $\overline{\boldsymbol{\lambda}}^*$. This process is repeated until the optimal objective value of each sub_l is non-negative.

4.3.3 Time-efficiency of CG

We first analyze the time complexity of RDW and sub_l in each iteration n:

1) RDW: The number of decision variables in RDW is at most nK, as in each iteration we only add up to 1 column for each polyhedron. Based on the experimental results (Fig. 13(d)), it takes CG up to 5 iterations to reach a near-optimal solution ($n \leq 5$), indicating that RDW contains $O(K)$ decision variables per iteration.

2) sub_l: Each sub_l has K decision variables in $\mathbf{z}_l$.

Accordingly, the numbers of decision variables in RDW and sub_l are both $O(K)$, indicating that both problems can be solved efficiently with standard LP methods.

The next question is how many iterations are needed to achieve the optimal solution, namely the *convergence* of CG. In fact, CG in DW formulation has been proved to have *finite convergence* [45]. Nevertheless, as pointed by our experimental results in Fig. 13(a)), there is possibly a long tail of the convergence. To increase the time-efficiency, we set ξ by a negative value with small magnitude, such that the algorithm will be ended immediately once the objective value of each sub_l is at least ξ. The experimental results demonstrate that, with proper value set to ξ, the computation time of CG will be reduced significantly with the objective value (ETDD) sacrificed a little (e.g., by up to 3.12% in Fig. 13(c)). More details will be discussed in Section 5. Theorem 4.4 gives an lower (dual) bound of the DW's optimal to check how close CG can achieve the optimal:

Theorem 4.4. In each iteration of CG,

$$\omega = \sum_k \overline{\pi}_k^* + \sum_l (\overline{\mu}_l^* - \zeta_l) \tag{27}$$

offers a lower bound of DW's optimal, where

$$\zeta_l = \min_{\mathbf{z}_l \in \Lambda_l} \left\{ \sum_k \hat{z}_{k,l}^t \overline{\pi}_k^* + \overline{\mu}_l^* + \sum_i c_{i,l} \hat{z}_{i,l}^t \right\}. \tag{28}$$

Proof According to Equ. (28),

$$\min_{\mathbf{z}_l \in \Lambda_l} \left\{ \sum_k \hat{z}_{k,l}^t \overline{\pi}_k^* + (\overline{\mu}_l^* - \zeta_l) + \sum_i c_{i,l} \hat{z}_{i,l}^t \right\} = 0, \tag{29}$$

indicating that $\overline{\pi}_1^*$, ..., $\overline{\pi}_K^*$, $(\overline{\mu}_1^* - \zeta_1)$, ..., $(\overline{\mu}_K^* - \zeta_K)$ construct a feasible solution of the dual problem of RDW. Therefore,

the corresponding objective value in the dual problem, i.e., $\sum_k \overline{\pi}_k^* + \sum_l (\overline{\mu}_l^* - \zeta_l)$ offers a lower bound of the optimal solution of RDW (according to weak duality [41]).

4.4 Trade-off Analysis of QoS and Privacy

In addition, we note that it is difficult to decrease the quality loss (ETDD $\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right)$) and increase the privacy level ϵ at the same time. Hence, in practice, it is important to balance QoS and privacy based on users' preferences. For theoretical interests, Proposition 4.5 describes a relationship between $\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right)$ and ϵ by a closed-form expression:

Proposition 4.5. $\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right)$ is lower bounded by $\max_l \kappa_l^{\max}(\epsilon)$, where $\kappa_l^{\max}(\epsilon) = \max_j \{\kappa_{l,j}(\epsilon)\}$ and $\kappa_{l,j}(\epsilon) = \sum_i c_{i,j} e^{-\epsilon d_{\mathcal{G}}^{\min}(\mathbf{u}_i^e, \mathbf{u}_l^e)}$.

Proof According to Equation (6), for each pair of $z_{i,j}$ and $z_{l,j}$, we have $\frac{z_{i,j}}{z_{l,j}} \geq e^{-\epsilon d_{\mathcal{G}}^{\min}(\mathbf{u}_i^e, \mathbf{u}_l^e)}$. Given that $\sum_i c_{i,j} z_{i,j} = \sum_i c_{i,j} z_{l,j} \frac{z_{i,j}}{z_{l,j}}$, we can obtain

$$\sum_i c_{i,j} z_{i,j} \geq \underbrace{\sum_i c_{i,j} e^{-\epsilon d_{\mathcal{G}}^{\min}(\mathbf{u}_i^e, \mathbf{u}_l^e)}}_{\kappa_{l,j}(\epsilon)} z_{l,j} \tag{30}$$

and hence

$$\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right) = \sum_j \sum_i c_{i,j} z_{i,j} \geq \sum_j \underbrace{\kappa_{l,j}(\epsilon)}_{\text{has to be smaller than } \kappa_l^{\max}(\epsilon), \forall j} z_{l,j}$$

$$\Rightarrow \mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right) \geq \max_l \kappa_l^{\max}(\epsilon). \tag{31}$$

The proof is completed.

Note that $\max_l \kappa_l^{\max}(\epsilon)$ $(l = 1, ..., K)$ is a function that decreases monotonically. Hence, Proposition 4.5 offers a reference to balance privacy and QoS in our scheme: 1) to achieve a target privacy level, the quality loss cannot be lower than a derived bound, and 2) lower ϵ (i.e., higher privacy level) enforces a higher lower bound of $\mathbf{E}\left(\Delta d_{\mathcal{G}}\left(P, \tilde{P}; Q\right)\right)$, indicating that a higher quality loss is likely to generate. In Section 5.1, we will further analyze how quality loss is impacted by ϵ by trace-driven simulation (in Fig. 11(a)).

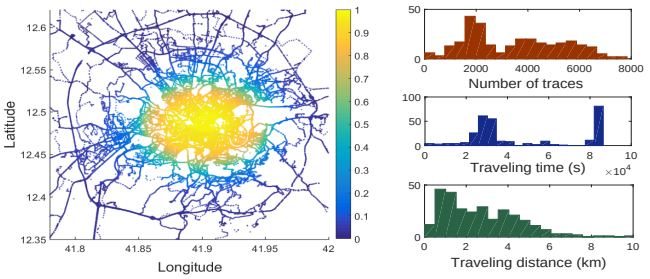
5 Performance Evaluation

In this section, we turn our attention to practical applications of our location obfuscation approach.

5.1 Simulation

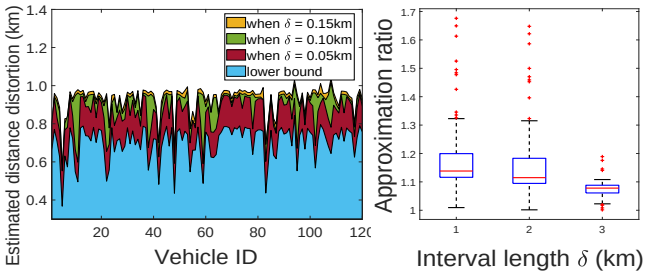
The two main metrics we will measure include:
1) *ETDD*, which is defined in Equ. (12). We use this metric to reflect the quality loss of location obfuscation strategies.
2) *The best guess of the adversary given the reported*, or *AdvError* for short [24]. Here we assume the adversary use the *optimal inference attack* [27]. We adopt this metric to reflect the *privacy level* that our approach can achieve, where higher AdvError indicates higher privacy level.

Dataset. We test the performance of our approach with a real-world dataset provided by [34], which records the trajectories of taxi cabs in Rome, as taxi services are considered as a type of VSC [34]. The dataset contains GPS coordinates of approximately 290 taxis collected over 30 days. Fig. 9 depicts the heat map of all taxi cabs' recorded location, from which we can observe that, on average taxi cabs are more likely located in downtown than in the suburbs. Also, different vehicle has different number of recorded location points, traveling time, and path distance, where Fig. 9(b) gives the histogram of these three metrics. Here, we select the 120 cabs with the highest number of records in the trace and estimate each cab's prior probability distribution $f_P(\mathbf{p})$ based on its own records. We conduct a simulation for each single cab, where we randomly pick up a location on the road network based on the vehicle's $f_P(\mathbf{p})$, and find the obfuscated location with our approach. In addition, we assume that the task's (customer's) location has the same probability distribution with the location of all cabs [37].



(a) Taxis location density (b) Statistics of taxis trace

Fig. 9. Taxi cabs' trajectories in Rome, Italy.

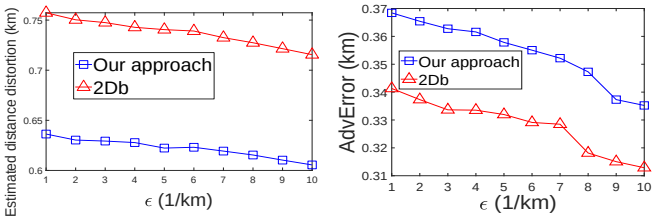


(a) Comparison with the lower bound (b) Approximation ratio

Fig. 10. Performance of our approach with different δ (simulation).

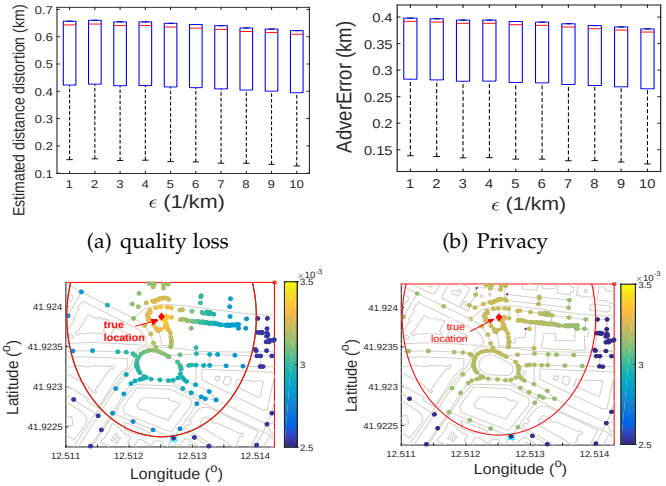
Comparison with the lower bound. As the minimum quality loss in VLP is within the gap between its lower bound (derived in Proposition 3.3. in [30]) and the quality loss of our approach, it is interesting to check how close our approach can achieve the optimal by comparing the QoS achieved by our approach with the lower bound. Fig. 10(a) compares the quality loss of our approach and the lower bound for 120 cabs, where we set the interval size δ in the D-VLP by 0.05km, 0.1km, and 0.15km, respectively. From the figure, we can observe that the denser we partition the road network in the approximation algorithm, the closer our solution can achieve the lower bound. This observation is also confirmed in Fig. 10(b), which gives the box plot of the approximation ratio of our approach (i.e,, the ratio between our approach's quality loss and the lower bound) for 120 cabs with different δ. Note that when the approximation ratio = 1, the solution achieves the optimal.

Comparison with 2D-plane based methods. We also compare our approach with the existing 2D-based location obfuscation methods. Here, we pick a state of the art mechanism introduced in [24] as baseline. Note that [24] is also a global optimization framework: given the privacy constraint, the object is to minimize the quality loss. Different from our approach, this 2D-based method (or simply *2Db*) assume locations on a 2D plane and both quality loss



(a) quality loss (b) Privacy

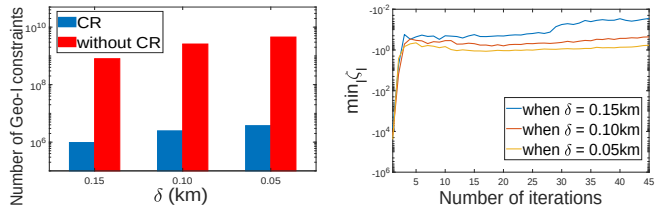
Fig. 11. Comparison of our approach with 2Db (simulation).



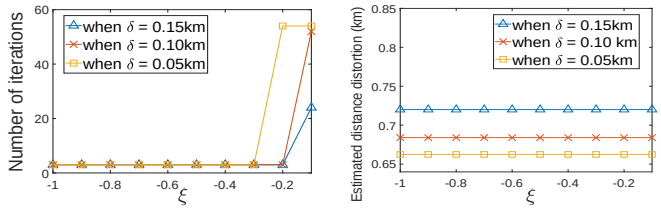
(a) quality loss (b) Privacy

(c) Obfuscation location probability distribution when $\epsilon = 10/\text{km}$ (d) Obfuscation location probability distribution when $\epsilon = 2/\text{km}$

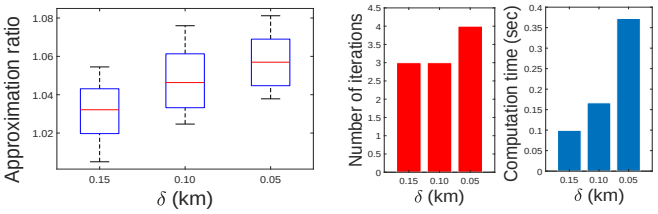
Fig. 12. Performance of our approach with different ϵ (simulation).



(a) Number of Geo-I constraints with and without CR (b) Convergence of CG with different δ



(c) Number of Geo-I constraints with and without CR (d) Convergence of CG with different δ



(e) Approximation ratio of CG with different δ (f) Number of iterations and computation time of CG

Fig. 13. Time-efficiency improved by constraint reduction and the column generation.

and privacy are measured by the Euclidean distance[3]. Fig. 11(a)(b) compare the average quality loss and AdvError of 120 cabs by using our approach and 2Db, respectively. Not surprisingly, our approach outperforms 2Db in both metrics since 2Db neglects the structure feature of the road network. For example, a pair of locations with shorter Euclidean distance may take longer path distance in the road network.

Performance given different threshold values for ϵ. Besides testing our approach's effectiveness, we check how the parameter ϵ in Geo-I will impact the quality loss and privacy of our method in Fig. 12(a)(b), in which ϵ is changed from $1/\text{km}$ to $10/\text{km}$. From the figures, we observe that larger ϵ generates lower quality loss and lower AdvError. According to the definition of (ϵ, r)-Geo-I (Definition 3.1), with higher ϵ (e.g., $\epsilon = 10/km$), the obfuscated location probability can be less evenly distributed over the road network. Hence, the obfuscated location around the true location will have higher probability to be selected, as shown in the heat map in Fig. 12(c), leading to a lower quality loss and a lower AdvError. In contrast, when ϵ is lower (e.g., $\epsilon = 2/km$), the obfuscated location probability is required to be more evenly distributed, as shown in the heat map in Fig. 12(d). Then, obfuscated locations with relatively higher traveling distance from (to) the true location will have higher probability to be selected, which causes higher quality loss and AdvError. According to Fig. 11(c)(d), we also find that it is difficult to increase both QoS and privacy at the same time. As expected, it is very important to take a trade-off between the two objectives based on workers' preference.

Time-efficiency improved by constraint reduction. Next, we test how constraint reduction (CR) introduced in Section 4.2 can reduce the number of Geo-I constraints in Fig. 13(a). From the figure, we find that CR largely reduces the number of Geo-I constraints in D-VLP. More precisely, when $\delta = 0.15km, 0.1km, 0.05km$, the number of constraints are reduced by and 99.86%, 99.87%, and 99.88%, respectively. Here, the number of edges in the auxiliary graph are only 56.7%, 28.4%, and 18.9% higher than the number of vertices in the graph when $\delta = 0.15km, 0.1km, 0.05km$, indicating that CR reduces the number of constraints in D-VLP from cubic to approximately quadratic with the respect to the number of intervals partitioned in D-VLP.

Convergence of the Column Generation (CG) algorithm. we next evaluate the convergence of CG (introduced in Section 4.3.2). Fig. 13(b) shows how $\min_l\{\zeta_l\}$ changes over iterations when $\delta = 0.15km, 0.1km, 0.05km$ (i.e., when $\min_l\{\zeta_l\} = 0$, the algorithm ends with the optimal solution). From the figure we find that 1) $\min_l\{\zeta_l\}$ converges faster when δ is smaller (which generates more decision variables in D-VLP), and 2) there is a long tail in the convergence before $\min_l\{\zeta_l\}$ achieving 0. Therefore, it is not time-efficient to wait until $\min_l\{\zeta_l\} = 0$. As solution, We determine 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 $\min_l\{\zeta_l\} \geq \xi$ such that the time-efficiency of CG is guaranteed and the optimality of D-VLP is only sacrificed slightly. Clearly, higher ξ enforces ETDD to achieve the optimal more closely via CG,

3. Note that 2Db may choose an obfuscated location that is not included in any edge in the network, so given an obfuscated location calculated by 2Db, we assume that the adversary will take the location in the road network that has the shortest Euclidean distance to this obfuscated location as the "reported location" from the worker

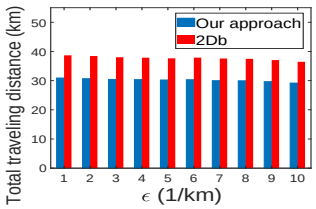


Fig. 14. Multi-vehicle task assignment (simulation).

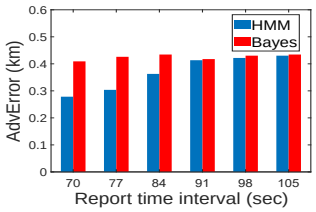


Fig. 15. Location spatial correlation aware attack (simulation).

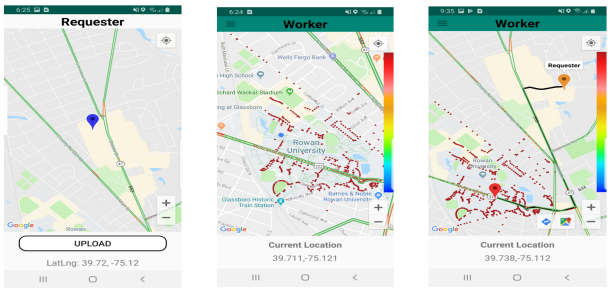
but will generate a higher computation load.

Fig. 13(b) indicates that the convergence of CG will slow down after $\min_l\{\zeta_l\}$ reaches a certain level. We choose to end the algorithm before $\min_l\{\zeta_l\}$ reaches the long tail, which slightly sacrifices the optimality of the CG's solution but significantly improves its time-efficiency. We implement it by setting a threshold ξ for $\min_l\{\zeta_l\}$ (ξ is a negative value close to 0) and end the algorithm once $\min_l\{\zeta_l\} \geq \xi$. Here, higher ξ generates a solution closer to the optimal, but is more likely to cause a higher computation load. Fig. 13(c)(d) show the number of iterations to end the algorithm and the ETDD, with ξ increased from -1.0 to -0.1. The figure indicates that when ξ reaches a certain value (e.g., $\xi = -0.2$ when $\delta = 0.05km$ and $\xi = -0.1$ when $\delta = 0.10km$ or $0.15km$), the number of iterations increases rapidly (by up to 20 times), but the corresponding quality loss (ETDD) is maintained at the same level. As such, we set $\xi = -0.3$ in the algorithm to lower the number of iterations without significantly affecting the quality loss.

Fig. 13(e) shows the approximation ratio of CG (i.e., the ratio of ETDD achieved by CG to the lower bound derived in Theorem 4.4) given different δ. The figure indicates that CG can achieve a near-optimal solution, i.e., when $\delta = 0.15km, 0.1km, 0.05km$, the average approximation ratios of CG are 1.031, 1.048, and 1.059, respectively. Fig. 13(f) lists the number of iterations and the corresponding computation time in CG, where the number of iterations is at most 4 and the highest computation time is 0.36s.

Multi-vehicle task assignment. Note that when ETDD for every single vehicle is lower, tasks are more likely to be matched to their nearby workers and hence the overall traveling cost of all the vehicles is lower. To test the cost-effectiveness of the GO function for the multi-vehicle task assignment, we randomly deploy 20 tasks and 30 vehicles over the maps, and let the server assign the tasks to the vehicles. Here, the vehicles' locations are obfuscated by our approach and 2Db, respectively. Fig. 14 compares the total traveling distance of all the vehicles using the two obfuscation methods, with ϵ increased from 1 to 10. The figure shows that our approach has a lower traveling cost than 2Db, since the server can estimate the traveling cost between vehicles and tasks more accurately using our approach, leading to more efficient task assignments.

Spatial correlation aware inference attack. Next, we test the privacy level achieved by our obfuscation algorithm when vehicles report their locations multiple times. We consider two types of threat models introduced in Section 3.2: Bayesian inference attack (denoted by Bayes) and spatial correlation aware attack (denoted by HMM). Fig. 15 compares AdvError achieved by our approach under the attack of Bayes and HMM, when the report time interval is increased from 70 to 105 seconds[4]. From the figure, we can find that our obfuscation strategy achieves a lower privacy level under HMM than under Bayes, particularly when reporting time interval is lower than 84 seconds. It is because that lower report time interval indicates a higher spatial correlation between adjacent reported locations, which helps HMM improve the accuracy of the inference attack. The figure also indicates that when the report time interval is high enough, e.g., higher than 91 seconds, the correlation of adjacent reports from vehicles is insignificant, and our approach has a similar accuracy under HMM and Bayes. We can also find that the accuracy of our approach is not impacted by report interval under Bayes since Bayes infers vehicles' locations in each round independently without considering the correlation between adjacent reports.



(a) Requester (b) Worker (I) (c) Worker (II)

Fig. 16. User interface of the prototype.

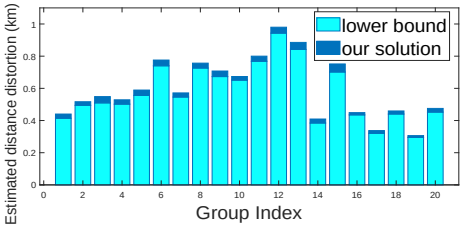


Fig. 17. Comparison with the lower bound (real implementation).

5.2 Real Implementation

Finally, we build a prototype of our obfuscation approach, including functions such as task request/assignment and location obfuscation. On the user-side, an Android APP on smartphones has been developed based on the Google map API. Fig. 16(a)(b)(c) show the user interfaces, where each user can register as either a worker or a requester:

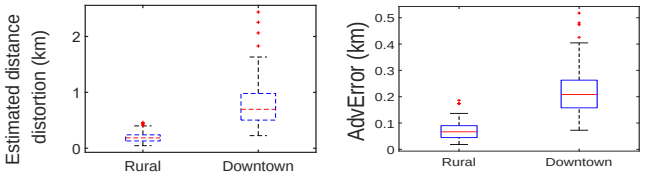
Requester: As shown in Fig. 16(a), a requester can upload his/her task with the task location specified.

Worker: The APP downloads the obfuscation matrix from the server, based on which the worker can report his/her obfuscated location to the server. Fig. 16(b) shows the distribution of obfuscated locations. After receiving workers' reported location, the server sends a list of task requests to the worker, along with the traveling cost estimated based on the worker's obfuscated location. By selecting a task request, a route will be displayed on the map to navigate this worker to the selected task location, as shown in Fig. 16(c).

Based on this prototype, we then carried out a pilot study to test the performance of our obfuscation strategy.

4. In [34], each taxi reports location every 7 seconds. To create a trajectory with the report time interval equal to $7n$, we take 1 sample from every n reports in the trace.

Fig. 18. Different regions in Glassboro (A: Rural area. B: Downtown)



(a) quality loss (b) Privacy

Fig. 19. Comparison of our approach with a 2D-based method in region A and B (real implementation).

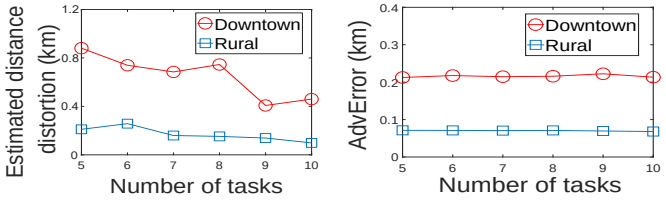
We select the Rowan campus as the target region and set the length of road partitioned interval $\delta = 0.05km$.

First, we randomly deploy 5 tasks over the campus. A vehicle participant equipped with a smartphone (considered as the worker) drove around the campus and reported the location every 20–30 seconds. We conduct 20 groups of tests. Fig. 17 compares ETDD achieved by our approach with the corresponding lower bound derived in Theorem 4.4. The figure demonstrates that our approach can achieve the optimal closely, where the approximation ratio is up to 1.14 among the 20 groups.

Moreover, to test the impact of the road network on the performance of our approach, we select two regions that have different road network features (as Fig. 18 shows):
1) **Region A (a rural area)**: where road segments are sparsely distributed, with less one-way streets, and
2) **Region B (Glassboro downtown)**: where road segments are densely distributed, with more one-way streets.
We randomly deploy 50 tasks in each region and run the experiment in each region, respectively.

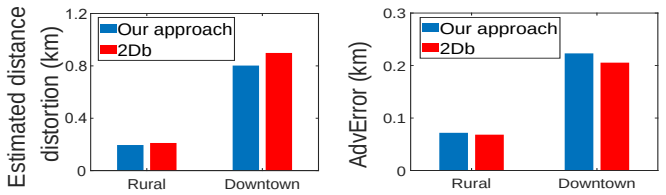
We first let the participant generate obfuscated locations using our approach, where ETDD and AdvError in regions A and B are depicted in Fig. 19(a)(b). From the figures, we can observe that, on average, ETDD and AdvError in region B are 310.68% and 210.52% higher than in region A. ETDD in downtown (region B) has higher ETDD because, with a higher number of road segments and one-way streets, the sensitivity of traveling distance to obfuscation is higher, i.e., obfuscation is more likely to generate high ETDD. Also, AdvError is higher in the downtown area due to the more complicated road network topology, where the deviation between the locations estimated by the adversary and the real locations is higher over roads.

We then change the number of tasks from 5 to 10 in both regions A and B, and depict the average ETDD and the average AdvError in Fig. 20(a)(b). The figures imply that in both regions 1) ETDD decreases with the increase of the number of tasks, while 2) AdvError is maintained at the same level as the number of tasks increases. For 1), it is because when more tasks are distributed in the region, the average distance to the nearest task is smaller, leading to a smaller ETDD on average. For 2), both our LP approach



(a) quality loss (b) Privacy

Fig. 20. Comparison of our approach with a 2D-based method with different task load (real implementation).



(a) quality loss (b) Privacy

Fig. 21. Comparison of our approach with a 2D-based method with different task load (real implementation).

(which generates the obfuscated location) and the Bayesian inference attack (Equ. (4), which takes obfuscated locations as inputs and outputs the estimated location) don't take the number of tasks as an input. Therefore, the adversary's estimated locations and AdvError are not impacted by the number of tasks.

Finally, we let the vehicle participate generate obfuscated locations using the 2D-based method, and compare its ETDD and AdvError with our approach in Fig. 21(a)(b), respectively. The two figures demonstrate that our approach outperforms the 2D-based method in terms of both ETDD and AdvError, which is consistent with the simulation results in Fig. 11(a)(b). Particularly, in region A, ETDD and AdvError of our approach are 7.41% lower and 5.21% higher than that of the 2D-based method. In region B, our approach's ETDD and AdvError are 10.71% lower and 8.64% higher than the 2D-based method.

6 Related Work

During the last decade, a variety of location privacy protection approaches have been developed. Many of these methods allow users to hide his identity from the server, such as *k-anonymity* (i.e., a user's location cannot be distinguished with the other $k-1$ users) [14], [15], *cloaking* (i.e., the accurate location is hidden in a obfuscated region) [16], [17]. Some other works let users use pseudonyms to interact with the system, where the users can change their pseudonyms without being traced by the system [18]. These approaches cannot be applied to VSC, since in VSC workers' identity has to be known by the server for task distribution. Although *obfuscation* has been also widely used for protecting location privacy [19]–[23], [27]. It introduces errors to location-based services, and hence one key problem is how to establish a trade-off between QoS and privacy. For example, the strategies introduced in [24]–[26] follow a global optimization framework, which QoS (or privacy) constraints are satisfied.

Differential privacy [48]–[50] has also been applied to address location privacy issues, though many of these works are used to protect aggregate location information [51]–[53], which is much different from the problem we discuss

in this paper. As apposed to requiring low sensitivity of aggregate output to a single individual change, the notion of Geo-I we adopt in this paper sets constraints such that any small change of location will not have a significant effect on adversary's observation. Following this notion, many Geo-I based location obfuscation strategies have been proposed [24]–[27]. Recently, some researchers have started working on location privacy issues of some specific location based services (LBS), such as mobile spatial crowdsourcing (SC) [8]–[10]. Similar to our work, most of these methods target maximizing the reacheability from workers to spatial tasks without compromising workers' location privacy. However, all these works assume both workers' and tasks' location on a 2D plane. As we have demonstrated that 2D-based strategy cannot effectively achieve high QoS and high privacy in a vehicle road network, these existing approaches cannot be applied to VSC.

Recently, there are also some strategies proposed to protect location privacy by considering the network constrained mobility features [54], [55]. However, these works are designed under different scenarios with ours. For example, [54] proposes a location protection mechanism to achieve l-diversity over roads, i.e., the user's location cannot be distinguished with other $l-1$ dummy locations. However, l-diversity is hardly to achieve in many applications as it is based on a strong assumption that dummy locations are equally likely to be the real location from the adversary's point of view [27], [28]. By leveraging environmental factors such as road network and traffic conditions, [54] introduces a threat model that tracks vehicles' trajectories according to users' driving behaviors (harsh braking/accelerating), which is different from our work as we assume that adversaries' inference attack is based on vehicles' obfuscated locations.

To date, the work closest to ours is [24]. [24] proposes an optimal location obfuscation mechanism with regard to Geo-I based on LP. However, their approach still assumes users' locations on a 2D plane and hence cannot be applied in VSC. On the other hand, although [24] also proposes an approximation technique to reduce the number of constraints in LP, their approach may not guarantee optimal solutions as it shrinks the LP's feasible region. Instead, by exploiting some unique features of Geo-I on road networks, our constraint reduction approach can significantly reduce the computation time without losing the optimality of the original problem. Moreover, our approach is more time-efficient as we apply decomposition techniques by using the constraint structure of the LP problem.

7 Conclusions

In this paper, we designed a location obfuscation strategy to minimize the quality loss of task distribution without compromising workers' location privacy, as defined by Geo Indisitiguishability constraints. Through discretization, we approximated our obfuscation problem as a LP problem that can be solved and further reduce its complexity by constraint reduction and optimization decomposition. Finally, our experimental results demonstrate that our approach can well approximate the optimal QoS, and also outperforms state-of-the-art location obfuscation strategy in terms of both QoS and privacy.

We see a number of promising directions for this research work. For exmaple, we plan to further investigate VSC privacy frameworks in heterogeneous settings, in which users may have different QoS preferences over different regions in the road network, e.g., some workers may tolerate less quality loss in downtown than in suburban areas. Moreover, we plan to consider different threat models where the information disclosed to adversary is not only users' uploaded location (e.g., mobile devices' accelerometer and gyroscope).

References

[1] L. Kazemi and C. Shahabi. Geocrowd: Enabling query answering with spatial crowdsourcing. In *Proc. of ACM SIGSPATIAL*, pages 189–198, 2012.

[2] Wei Li, Haiquan Chen, Wei-Shinn Ku, and Xiao Qin. Scalable spatiotemporal crowdsourcing for smart cities based on particle filtering. In *Proc. of ACM SIGSPATIAL*, 2017.

[3] Yongliang Sun, Yejun Sun, and Kanglian Zhao. *Spatial Crowdsourcing-Based Sensor Node Localization in Internet of Things Environment*. 2018.

[4] X. Wang, X. Zheng, Q. Zhang, T. Wang, and D. Shen. Crowdsourcing in ITS: The state of the work and the networking. *IEEE T-ITS*, 17(6):1596 – 1605, 2016.

[5] Di Wu, Yuan Zhang, Lichun Bao, and Amelia Regan. Location-based crowdsourcing for vehicular communication in hybrid networks. *IEEE T-ITS*, 14:837–846, 2013.

[6] Z. Ou and et al. Utilize signal traces from others? a crowdsourcing perspective of energy saving in cellular data communication. *IEEE TMC*, 14(1):194–207, 2015.

[7] A. Misra, A. Gooze, K. Watkins, M. Asad, , and C. A. Le Dantec. Crowdsourcing and its application to transportation data collection and management. *Transportation Research Record: Journal of the Transportation Research Board*, 2414(1):1 – 8, 2014.

[8] 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.

[9] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen. Online mobile micro-task allocation in spatial crowdsourcing. In *Proc. of IEEE ICDE*, pages 49–60, 2016.

[10] H. To, C. Shahabi, and L. Xiong. Privacy-preserving online task assignment in spatial crowdsourcing with untrusted server. In *Proc. of IEEE ICDE*, 2018.

[11] B. Liu, L. Chen, X. Zhu, Y. Zhang, C. Zhang, and W. Qiu. Protecting location privacy in spatial crowdsourcing using encrypted data. In *Proc. of EDBT*, 2017.

[12] A. Khoshgozaran and C. Shahabi. Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy. In *Proc. of Advances in Spatial and Temporal Databases*, pages 239–257, 2007.

[13] G. Ghinita et al. Private queries in location based services: Anonymizers are not necessary. In *Proc. of ACM SIGMOD*, 2008.

[14] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of MobiSys*, pages 31–42, 2003.

[15] B. Gedik and Ling Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. of IEEE ICDCS*, pages 620–629, June 2005.

[16] C. Chow. *Cloaking Algorithms for Location Privacy*, pages 93–97. Springer US, Boston, MA, 2008.

[17] Z. Huang and M. Xin. A distributed spatial cloaking protocol for location privacy. *TrustCom*, 2010.

[18] A. R. Beresford and F. Stajano. Location privacy in pervasive computing. In *In Proc. of IEEE PerCom*, pages 46–55, 2003.

[19] P. Shankar, V. Ganapathy, and L. Iftode. Privately querying location-based services with sybilquery. In *Proc. of UbiComp*, pages 31–40, 2009.

[20] Richard Chow and Philippe Golle. Faking contextual data for fun, profit, and privacy. In *Proc. of WPES*, pages 105–108, 2009.

[21] V. A. Kachore, J. Lakshmi, and S. K. Nandy. Location obfuscation for location data privacy. In *Proc. of IEEE World Congress on Services*, 2015.

[22] J Zhang, K Liu, and J Yang. An obfuscation-based approach for location privacy protection. *Journal of Computational Information Systems*, 2013.

[23] M. Li, S. Salinas, A. Thapa, and P. Li. n-cd: A geometric approach to preserving location privacy in location-based services. In *Proc. of IEEE INFOCOM*, 2013.

[24] N. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proc. of ACM CCS*, 2014.

[25] K. Chatzikokolakis, C. Palamidessi, and M. Stronati. Constructing elastic distinguishability metrics for location privacy. *PoPETs*, 2015:156–170, 2015.

[26] R. Shokri. Privacy games: Optimal user-centric data obfuscation. *Proc. on Privacy Enhancing Technologies*, 2015(2):299 – 315, 2015.

[27] L. Yu, L. Liu, and C. Pu. Dynamic differential location privacy with personalized error bounds. In *Proc. of ACM NDSS*, 2017.

[28] M. Andrés et al. Geo-indistinguishability: Differential privacy for location-based systems. In *Proc. of ACM CCS*, pages 901–914, 2013.

[29] C. Dwork, , F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. Springer Berlin Heidelberg, 2006.

[30] C. Qiu and A. C. Squicciarini. Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability. In *Proc. of IEEE ICDCS*, 2019.

[31] D. B. Johnson and D. A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*. Springer US, Boston, MA, 1996.

[32] Rinku Dewri, Prasad Annadata, Wisam Eltarjaman, and Ramakrishna Thurimella. Inferring trip destinations from driving habits data. In *Proc. of ACM WPES*, pages 267–272, 2013.

[33] Xianyi Gao and et al. Elastic pathing: Your speed is enough to track you. In *Proc. of ACM UbiComp*, pages 975–986, 2014.

[34] L. Bracciale et al. CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from https://crawdad.org/roma/taxi/20140717, July 2014.

[35] 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.

[36] S. Narain, T. D. Vo-Huu, K. Block, and G. Noubir. Inferring user routes and locations using zero-permission mobile sensors. In *2016 IEEE Symposium on Security and Privacy (SP)*, 2016.

[37] 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.

[38] K. Fawaz and K. G. Shin. Location privacy protection for smartphone users. In *Proc. of ACM CCS*, pages 239–250, 2014.

[39] George Casella and Roger Berger. *Statistical Inference*. Duxbury Resource Center, June 2001.

[40] G. D. Forney. The viterbi algorithm. *Proc. of the IEEE*, 61(3):268–278, 1973.

[41] F. S. Hillier. *Linear and Nonlinear Programming*. Stanford University, 2008.

[42] Harsh Bhasin. *Algorithms: Design and Analysis*. Oxford Univ Press, 2015.

[43] D. Palomar and M. Chiang. A tutorial on decomposition methods for network utility maximization. *IEEE JSAC*, 24(8):1439–1451, Aug 2006.

[44] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations Research*, (8):101–111, 1960.

[45] 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.

[46] N. Maculan, M. Passini, B. Moura, and I. Loiseau. Column-generation in integer linear programming. *RAIRO*, 37(2):67–83, 2003.

[47] 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. In *Proc. of Conference on Information and Knowledge Management (CIKM)*, 2020.

[48] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Proc. of Automata, Languages and Programming*, pages 1–12. Springer Berlin Heidelberg, 2006.

[49] B. Palanisamy, C. Li, and P. Krishnamurthy. Group differential privacy-preserving disclosure of multi-level association graphs. In *Proc. of IEEE ICDCS*, 2017.

[50] F. Ahmed, A. X. Liu, and R. Jin. Social graph publishing with privacy guarantees. In *Proc. of IEEE ICDCS*, 2016.

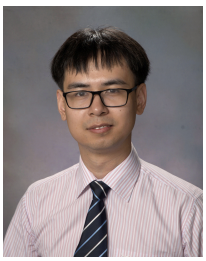
[51] R. Dewri. Local differential perturbations: Location privacy under approximate knowledge attackers. *IEEE TMC*, 12(12):2360–2372, 2013.

[52] C. Yin, J. Xi, R. Sun, and J. Wang. Location privacy protection based on differential privacy strategy for big data in industrial internet of things. *IEEE Transactions on Industrial Informatics*, 2018.

[53] Raluca Ada Popa, Andrew J. Blumberg, Hari Balakrishnan, and Frank H. Li. Privacy and accountability for location-based aggregate statistics. In *Proc. of ACM CCS*, 2011.

[54] T. Wang and L. Liu. Privacy-aware mobile services over road networks. *VLDB Endow.*, 2(1):1042–1053, August 2009.

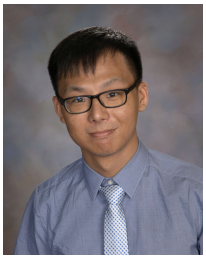
[55] L. Zhou, S. Du, H. Zhu, C. Chen, K. Ota, and M. Dong. Location privacy in usage-based automotive insurance: Attacks and countermeasures. *IEEE TIFS*, 14(1):196–211, 2019.



Chenxi Qiu received a BS degree in Telecommunication Engineering from Xidian University, China, in 2009 and a Ph.D. degree in Electrical and Computer Engineering in Clemson University in 2015. He worked as a Postdoc scholar at Penn State University from 2016 to 2018. Currently, he is an assistant professor in the Department of Computer Science at Rowan University, NJ, United States. His research interests include studying privacy&security issues in context of the Internet of Things (IoT).

Anna Cinzia Squicciarini is an associate Professor in the College of Information Sciences and Technology at the Pennsylvania State University. She received her PhD in Computer Science from the University of Milan. From February 2006 to December 2007, Squicciarini was a post-doctoral fellow at Computer Science Department of Purdue University. Her main research interests are in the area of data security and privacy, with emphasis on access control mechanisms.

Ce Pang is currently an undergraduate student in the Department of Computer Science at Rowan University, NJ, United States. Mr. Pang's current research focuses on addressing location privacy issues in spatial crowdsourcing systems. He also has experience in research topics related to data visualization.



Ning Wang is an assistant professor in the Department of Computer Science with a joint appointment in the Department of Electrical and Computer Engineering at Rowan University. He received his Ph.D. from Temple University in 2018. Before that, he received his B.S. in Electrical Engineering from University of Electronic Science and Technology of China in 2013. Dr. Wang focuses on solving networking problems in Internet-of-Things Systems and Smart Cities through algorithmic approaches.

Ben Wu received Ph.D. (with distinction) in electrical engineering from Princeton University, Princeton, NJ in 2015, and received B.Sci. in optoelectronics from Nankai University, Tianjin, China in 2008. He is currently an assistant professor at Rowan University, department of electrical and computer engineering. From 2015 to 2016, he was a postdoctoral researcher in Princeton University. He was a summer research intern at NEC Laboratories America Inc, Princeton, NJ in 2015. His current research interests include optical stealth transmission, optical encryption and photonic neuromorphic signal processing.