Time-Efficient Locally Relevant Geo-Location Privacy Protection

Chenxi Qiu
University of North Texas
chenxi.qiu@unt.edu

Ruiyao Liu
University of North Texas
ruiyaoliu@my.unt.edu

Primal Pappachan
Portland State University
primal@pdx.edu

Anna Squicciarini
The Pennsylvania State University
acs20@psu.edu

Xinpeng Xie
University of North Texas
xinpengxie@my.unt.edu

Abstract

Geo-obfuscation serves as a *location privacy protection mechanism (LPPM)*, enabling mobile users to share obfuscated locations with servers, rather than their exact locations. This method can protect users' location privacy when data breaches occur on the server side since the obfuscation process is irreversible. To reduce the utility loss caused by data obfuscation, *linear programming (LP)* is widely employed, which, however, might suffer from a polynomial explosion of decision variables, rendering it impractical in large-scale geo-obfuscation applications.

In this paper, we propose a new LPPM, called *Locally Relevant Geo-obfuscation (LR-Geo)*, to optimize geo-obfuscation using LP in a time-efficient manner. This is achieved by confining the geo-obfuscation calculation for each user exclusively to the *locally relevant (LR)* locations to the user's actual location. Given the potential risk of LR locations disclosing a user's actual whereabouts, we enable users to compute the LP coefficients locally and upload them only to the server, rather than the LR locations. The server then solves the LP problem based on the received coefficients. Furthermore, we refine the LP framework by incorporating an exponential obfuscation mechanism to guarantee the indistinguishability of obfuscation distribution across multiple users. Based on the constraint structure of the LP formulation, we apply Benders' decomposition to further enhance computational efficiency. Our theoretical analysis confirms that, despite the geo-obfuscation being calculated independently for each user, it still meets geo-indistinguishability constraints across multiple users with high probability. Finally, the experimental results based on a real-world dataset demonstrate that LR-Geo outperforms existing geo-obfuscation methods in computational time, data utility, and privacy preservation.

Keywords

Geo-obfuscation, location privacy, linear programming

1 Introduction

Among a variety of *location privacy protection mechanisms (LPPMs)*, geo-obfuscation has become the preferred paradigm for protecting individual location privacy against server-side data breaches [1]. Geo-obfuscation allows mobile users to report obfuscated locations instead of their exact locations to servers in location-based services (LBS). As the obfuscation process is irreversible [2], users' exact locations are well-protected even if the obfuscated locations are disclosed to attackers. This is achieved by satisfying certain privacy criteria, such as *geo-indistinguishability (Geo-Ind)* [3], which requires that, for any two locations geographically close, the probability distribution of their obfuscated locations should be sufficiently close so that it is difficult for an attacker to distinguish the two locations based on their obfuscated representations.

Although geo-obfuscation provides a strong privacy guarantee for users' locations, the location errors introduced by obfuscation can negatively impact the quality of LBS. Many recent efforts [1, 4–14] aim to address the quality issue caused by geo-obfuscation using *linear programming (LP)* [15], of which the objective is to minimize the utility loss with the privacy criterion like Geo-Ind guaranteed. For the sake of computational tractability, the LP-based methods typically discretize the location field into a finite set of discrete locations. Its decision variables, represented as an *obfuscation matrix*, determine the probability distributions of obfuscated locations given each possible real location.

Due to the intricate complexity of LP, generating the obfuscation matrix directly on users' mobile devices is not feasible. Instead, the matrix is calculated by a server, which optimizes the matrix before it is downloaded by the mobile devices [6]. Given that the server lacks knowledge of users' precise locations, the server typically considers every location within the target area when calculating the matrix, regardless of whether it is currently occupied by a user. After downloading the matrix, each user selects the specific row of the matrix that matches their actual location to determine the probability distribution of the obfuscated locations. Consequently, the LP formulation of the obfuscation matrix involves K^2 decision variables, where K denotes the number of discrete locations within the target region. This results in a significant challenge in accommodating a large array of locations. For instance, the inclusion of thousands of distinct locations within a modestly sized town escalates the number of decision variables into the millions [11]. As compared in Table 3 in Section 6 (Related Work), most current LP-based works limit the number of discrete locations K to up to 100.

Motivations. The traditional LP-based geo-obfuscation methods (e.g., [6, 11]) have a high computation overhead since the LP is formulated completely by the server side, which requires accounting for all locations within the target region. However, from an individual user's perspective, they engage only with the specific row that matches their actual location. Although this single row cannot be generated in isolation as it is linked to some other rows by "Geo-Ind", such constraints are only enforced between the nearby

Proceedings on Privacy Enhancing Technologies 2025(2), 5–22
© 2025 Copyright held by the owner/author(s).
https://doi.org/10.56553/popets-2025-0046

locations [3]. This indicates that, if the LP can be formulated locally by each user, they only need to consider "locally relevant" locations so that the computational cost can be significantly reduced. In practical terms, when a user chooses an obfuscated location, the relevance of how another user 100 kilometers away selects their obfuscated location due to the Geo-Ind constraints is minimal.

Motivated by the above observation, this paper introduces a new geo-obfuscation paradigm, termed *Locally Relevant Geo-obfuscation (LR-Geo) locations*. The core idea of LR-Geo is to allow each user to formulate the LP by themselves by focusing exclusively on their *Locally Relevant (LR)*, thereby streamlining the process of generating obfuscation matrices. Nevertheless, the development of LR-Geo presents several distinct challenges:

Challenge 1: How to determine the LR location set? First, it is important to note that even a location far from a user's location can have an indirect impact on the user's obfuscation distribution since the distant location can have higher relevance to other locations closer to the user by the Geo-Ind constraints. Considering such a "multi-hop" influence of Geo-Ind is hard to circumvent while pursuing the globally optimal solution, our approach focuses on striking a balance between optimizing the obfuscation matrix and enhancing computational efficiency, achieved by selecting an appropriate LR location set.

Specifically, we introduce a *Geo-Ind graph* to describe the Geo-Ind constraints between each nearby location pair, which also enables us to quantify the "multi-hop" impact of Geo-Ind constraints through the path distance between nodes in the graph (see **Theorem 3.1**). Using the Geo-Ind graph, we determine the LR location set for each user as the collection of locations whose path distance from the user's actual location does not surpass a predefined threshold. Following this, we formulate the LP of LR-Geo for each user to focus exclusively on their selected LR location set.

Challenge 2: How to calculate LR-Geo? Despite having a relatively smaller LP size, the calculation of LR-Geo still needs to be migrated to the server since (i) the computational demands of LR-Geo remain relatively high for mobile devices, and (ii) LR-Geo's LP formulation involves assessing data utility for downstream decision-making, a task typically handled by the server rather than individual users [6]. However, each user needs to keep the LR location set hidden from the server, as these locations could potentially disclose the user's actual location. As a workaround, we enable each user to locally compute the coefficients of the LP formulation with server assistance and then upload these coefficients to the server. We demonstrate that the uploaded coefficients can be used by the server to solve the LP problems but cannot be reversed to unveil the LR location of the user (by examples in Section 4.7 and experimental results in Fig. 13 in Section 5).

Challenge 3: How to guarantee Geo-Ind across multiple users? Given that each user conceals their LR location set from the server, formulating Geo-Ind constraints across users in LP becomes another challenge for the server. To address this, we enable the server to apply *exponential distribution constraints* on a selected subset of obfuscated locations for each user. We demonstrate that adhering to these constraints ensures that the chosen obfuscated locations meet Geo-Ind constraints across users even though their obfuscation is calculated in an independent manner (see **Theorem 4.4**). Moreover, our experimental findings in Fig. 11 indicate that while unselected obfuscated locations do not theoretically guarantee Geo-Ind, they still possess a high probability (e.g. 99.81% on average) of meeting Geo-Ind constraints in practice. Additionally, by integrating the exponential mechanism with LP, the constraint matrix of LR-Geo follows a *ladder block structure*, making the problem well-suited to *Benders' decomposition*, which further improves the computation efficiency of solving LR-Geo.

Experimental results. Lastly, in our experiment, we assessed LR-Geo's performance by simulating its application to road map data sourced from Rome, Italy [16]. The results revealed that LR-Geo efficiently generates obfuscation matrices within 100 seconds for cases involving up to 1500 locations in the target area. This marks a substantial enhancement over existing LP-based geo-obfuscation techniques (as listed in Table 3), which can only handle up to 100 locations. Furthermore, our experimental results show that LR-Geo's obfuscation matrix not only adheres closely to the theoretical lower bound of expected cost, as established in **Theorem 4.7 and Theorem 4.6**, but also outperforms contemporary benchmarks [3, 11, 17] in terms of time efficiency and cost-effectiveness.

Contributions. In summary, the contributions of this paper are summarized as follows:

1. We introduce LR-Geo, a new geo-obfuscation approach that significantly reduces the computational overhead of geo-obfuscation while maintaining a high level of optimality.
2. We develop a remote computing framework that allows for the offloading of LR-Geo computations to a server while preserving the privacy of each user's LR location set.
3. To achieve Geo-Ind across multiple users' obfuscation matrices, we integrate exponential distribution constraints within the LP computational framework. Given LR-Geo's constraint structure, we apply Benders' decomposition to enhance computational time efficiency.
4. Our experimental results demonstrate that LR-Geo not only approximates optimal solutions with considerably lower computational costs but also outperforms existing state-of-the-art methods in time efficiency and cost-effectiveness.

The rest of the paper is organized as follows: The next section provides the preliminaries of geo-obfuscation. Section 3 describes the motivation and Section 4 designs the algorithm. Section 5 evaluates the algorithm's performance. Section 6 presents the related work and Section 7 makes a conclusion.

2 Preliminary

In this section, we introduce the preliminary knowledge of geo-obfuscation, including its framework in LBS in **Section 2.1**, its privacy criteria Geo-Ind in **Section 2.2**, and the LP formulation in **Section 2.3**. The main notations used throughout this paper can be found in Table 4 in **Section A in Appendix**.

2.1 Geo-Obfuscation in LBS

We consider an LBS system composed of a *server* and *a set of users*, where users need to report their locations to the server to receive the desired services. Like [3, 6, 9, 18], we assume that the server is not malicious, but it might suffer from a *passive attack where attackers can eavesdrop on the users' reported locations breached by the server.* In this case, users can hide their exact locations from the server using geo-obfuscation mechanisms [6].

In general, a geo-obfuscation mechanism can be represented as a *probabilistic function*, of which the input and the output are the user's real location and obfuscated location, respectively. For the sake of computational tractability, many existing works like [6, 9, 11, 19, 20] approximate the users' location field to a discrete location set $\mathcal{V} = \{v_1, ..., v_K\}$. In this case, the obfuscation function can be represented as a *stochastic obfuscation matrix* $\mathbf{Z} = \{z_{i,k}\}_{K\times K}$, where each $z_{i,k}$ denotes the probability of taking v_k as the obfuscated location given the actual location v_i.

Besides hiding the users' actual location, the obfuscation matrix $\mathbf{Z}$ is designed to minimize the *utility loss* (or *cost*) caused by geo-obfuscation. As an example, in this paper, we focus on a category of LBS where a mobile user needs to physically travel to a specified destination to receive service (e.g., hotel/restaurant recommendations [21]) or implement a task (e.g., spatial crowdsourcing [22, 23]). Typically, these LBS types strive to minimize travel expenses for users. Accordingly, *we define the cost resulting from geo-obfuscation as the distortion between the estimated travel distances (using obfuscated locations) and the actual travel distances incurred by users.* Note that our approach in this paper can be readily adapted in other LBS applications as long as the explicit relationship between cost and location obfuscation can be established.

To calculate the traveling costs, global LBS information such as traffic conditions and destination distribution is needed. Since global information is hard to maintain by individuals, many existing works like [1, 6, 7, 10, 11] let the server manage the computation of the obfuscation matrix. Specifically, before reporting the location to the server, each privacy-aware user downloads the obfuscation matrix $\mathbf{Z}$ from the server. Given the current location v_i, the user finds the corresponding row $\mathbf{z}_i = [z_{i,1}, ..., z_{i,K}]$ in the obfuscation matrix, based on which the user then randomly selects an obfuscated location to report. In what follows, we call $\mathbf{z}_i$ the *obfuscation vector* of the location v_i.

2.2 Geo-Indistinguishability

Although the server takes charge of generating the obfuscation matrix, the users' exact locations are still hidden from the server since the obfuscated locations are selected in a probabilistic manner [6]. In particular, the obfuscation matrix $\mathbf{Z}$ is designed to satisfy the privacy criterion *Geo-Ind*, indicating that even if an attacker has obtained the users' reported (obfuscated) location and $\mathbf{Z}$ from the server, it is still hard for the attacker to distinguish the users' exact locations from the nearby locations.

We use d_{v_i,v_j} to denote the *Haversine distance* (the angular distance on the surface of a sphere) between v_i and v_j. Given a threshold $\gamma > 0$, we call two locations v_i and v_j "neighboring locations" if their distance $d_{v_i,v_j} \leq \gamma$. *Geo-Ind* is formally defined in *Def. 2.1* [3]:

Definition 2.1. *(Geo-Ind) An obfuscation matrix* $\mathbf{Z}$ *satisfies* (ϵ, γ)*-Geo-Ind if, for each pair of neighboring locations* $v_i, v_j \in \mathcal{V}$ *with* $d_{v_i,v_j} \leq \gamma$*, the following constraints are satisfied*

$$z_{i,k} - e^{\epsilon d_{v_i,v_j}} z_{j,k} \leq 0, \ \forall v_k \in \mathcal{V}, \tag{1}$$

i.e., the probability distributions of the obfuscated locations of v_i *and* v_j *are sufficiently close. Here,* ϵ *is called the privacy budget. Higher* ϵ *implies a lower privacy level.*

In what follows, we use $\mathcal{E} = \left\{(v_i, v_j) \in \mathcal{V}^2 | d_{v_i,v_j} \leq \gamma\right\}$ to denote the set of neighboring locations in $\mathcal{V}$.

Note that the existing works like [6, 9, 11] do not require Geo-Ind to be satisfied only between locations that are within a distance smaller than γ. Here, we consider γ as it forms a more general model, i.e., $\gamma = \infty$ when γ is not included in *Definition 2.1.* In practical, the choice of γ depends on the user's privacy requirements, defining the range within which the user's location should be indistinguishable. For example, if a student wants to obscure their location within a campus, selecting γ to cover the entire campus would be sufficient.

2.3 LP Problem Formulation

Constraints. In addition to satisfying Geo-Ind in Equ. (1), for every real location v_i, the total probability of its obfuscated locations should be equal to 1:

$$\textstyle\sum_{k=1}^{K} z_{i,k} = 1, \ \forall v_i \in \mathcal{V} \text{ (probability unit measure).} \tag{2}$$

Objective function. Given the target location v_l, the real location v_i, and the obfuscated location v_k, we define the *cost* of LBS as the discrepancy between the estimated travel cost tc_{v_i,v_l} and the actual travel cost tc_{v_k,v_l} to reach v_l

$$\delta_{v_i,v_k,v_l} = \left|\mathrm{tc}_{v_i,v_l} - \mathrm{tc}_{v_k,v_l}\right|. \tag{3}$$

We assume that the server has the prior distribution of the target locations $\mathbf{q} = [q_1, ..., q_K]$, where q_l $(l = 1, ..., K)$ denotes the probability that a target's location is at v_l. The objective is to minimize the *expected cost* caused by the obfuscation matrix $\mathbf{Z}$:

$$\mathcal{L}(\mathbf{Z}) = \textstyle\sum_{i=1}^{K} p_i \sum_{k=1}^{K} \sum_{l=1}^{K} q_l \delta_{v_i,v_k,v_l} z_{i,k} = \sum_{i=1}^{K} \mathbf{c}_i \mathbf{z}_i^\top, \tag{4}$$

where p_k $(k = 1, ..., K)$ denotes the prior probability that a user's real location is at v_k, $\mathbf{c}_i = \left[c_{v_i,v_1}, ..., c_{v_i,v_K}\right]$ denote the cost (cost) coefficients of $\mathbf{z}_i$ in the objective function, and each c_{v_i,v_k} is given by

$$c_{v_i,v_k} = p_i \textstyle\sum_{l=1}^{K} q_l \delta_{v_i,v_k,v_l} \ (i = 1, ..., K). \tag{5}$$

In related works such as [3, 18], utility loss is typically defined as the distance between original and obfuscated locations. However, this metric does not fully capture the utility loss in many vehicle-based applications, as they fail to consider the constraints imposed by vehicle mobility [11]. Therefore, in this paper, we define utility loss based on downstream decision-making in data processing. Specifically, in our experiments, we focus on spatial crowdsourcing, where utility loss of $\mathbf{Z}$ is measured by the expected estimation error in travel cost caused by $\mathbf{Z}$.

Problem formulation. To satisfy the constraints of Geo-Ind (Equ. (1)) and the probability unit measure (Equ. (2)), and minimize $\mathcal{L}(\mathbf{Z})$ (Equ. (4)), the problem of *LR obfuscation matrix generation (OMG)* can be formulated as the following LP problem:

$$\min \quad \mathcal{L}(\mathbf{Z}) = \textstyle\sum_{i=1}^{K} \mathbf{c}_i \mathbf{z}_i^\top \tag{6}$$

$$\text{s.t.} \quad \text{Equ. (1)(2) are satisfied.} \tag{7}$$

3 Motivations and Observations

Although the OMG problem outlined in Equ. (6)(7) can be solved using classical LP algorithms such as the simplex method [15], it is hampered by high computational costs. The time complexity of an LP problem depends on the number of decision variables and the number of linear constraints [15]. In OMG, the decision matrix

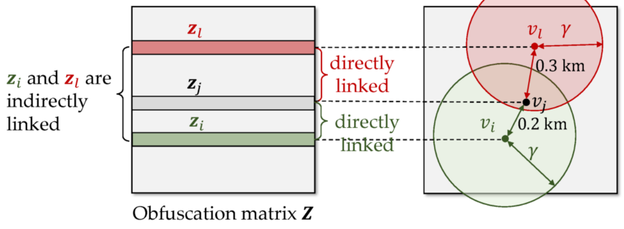


Figure 1: Directly/indirectly linked obfuscation vectors.

Z encompasses $O(|\mathcal{V}|^2)$ decision variables, where it must adhere to the Geo-Ind constraints for every pair of neighboring locations in $\mathcal{V}$, resulting in $O(|\mathcal{E}||\mathcal{V}|)$ linear constraints. This substantial computational demand renders the LP-based geo-obfuscation impractical for scenarios with a large number of locations. Therefore, **enhancing the computational efficiency of solving LP-based geo-obfuscation is the primary goal of this paper**.

Observations. When a user at location v_i seeks to obfuscate their actual location, they use only the ith row $\mathbf{z}_i = [z_{i,1}, ..., z_{i,K}]$ instead of the entire matrix $\mathbf{Z}$. However, determining $\mathbf{z}_i$ in isolation is not feasible within OMG as $\mathbf{z}_i$ is linked to other rows (obfuscation vectors) of $\mathbf{Z}$ by the Geo-Ind constraints.

As depicted in Fig. 1, Geo-Ind requires that obfuscation vector $\mathbf{z}_i$ is directly connected to another vector $\mathbf{z}_j$ if their corresponding locations v_i and v_j are neighbors. Additionally, a location v_l, even if distant from v_i, indirectly influences $\mathbf{z}_i$ through its neighborly relation with v_j. Considering that it is hard to circumvent such "multi-hop" influence of Geo-Ind between locations while pursuing the globally optimal solution, we aim to balance the optimality of geo-obfuscation and computational efficiency by focusing on a selectively identified set of locations that exert a significant influence on v_i. This approach is based on the intuition that locations nearer to v_i have obfuscation vectors $\mathbf{z}_l$ with a more pronounced effect on $\mathbf{z}_i$. The pivotal question then becomes how to quantify the extent of influence between $\mathbf{z}_i$ and other vectors, such as $\mathbf{z}_l$.

To this end, we introduce the concept of the *Geo-Ind graph* in **Def. 3.1**. We then detail the application of this graph to measure the Geo-Ind connection between $\mathbf{z}_i$ and $\mathbf{z}_l$ in **Theorem 3.1**.

Definition 3.1. *(Geo-Ind Graph) Geo-Ind graph is defined as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to describe the Geo-Ind constraints between locations within a set $\mathcal{V}$. Here, $\mathcal{V}$ represents the set of nodes, each corresponding to a distinct location, and $\mathcal{E}$ denotes the set of edges. Each edge $(v_i, v_j) \in \mathcal{E}$ indicates that the locations v_i and v_j are neighbors (i.e. $d_{v_i,v_j} \leq \gamma$) with the edge weight equal to the distance d_{v_i,v_j} between them.*

Theorem 3.1. *Consider two locations v_i and v_j are connected through at least one path in the Geo-Ind graph $\mathcal{G}$. Let the path distance D_{v_i,v_j} represent the sum of weights of the edges forming the shortest path between v_i and v_j. Their probabilities of selecting location v_k as the obfuscated location is constrained by:*

$$z_{i,k} \leq e^{\epsilon D_{v_i,v_j}} z_{j,k}. \tag{8}$$

Proof. Detailed proof can be found in Section B in Appendix. □

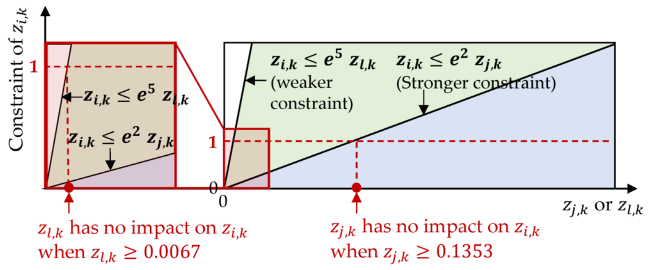


Figure 2: Strongly & weakly linked decision vectors.

Theorem 3.1 implies the extent to which a pair of obfuscation vectors, $\mathbf{z}_i$ and $\mathbf{z}_j$, are linked through Geo-Ind constraints depends on the path distance D_{v_i,v_j} between their respective locations v_i and v_j in the Geo-Ind graph $\mathcal{G}$. A higher path distance between locations implies a weaker linear constraint between their obfuscation vectors.

In Fig. 2, we follow the example of Fig. 1, and check specifically how the obfuscation vector $\mathbf{z}_i$ is impacted by the decision vectors $\mathbf{z}_j$ and $\mathbf{z}_l$ according to the conclusion of Theorem 3.1. As Fig. 2 shows, given the path distances $D_{v_i,v_j} = 0.2$km and $D_{v_i,v_l} = 0.5$km, and the privacy budget $\epsilon = 10.0\text{km}^{-1}$, each entry of $\mathbf{z}_i$ follows the following linear constraints according to Theorem 3.1:

$$z_{i,k} \leq e^{\epsilon D_{v_i,v_j}} z_{j,k} \Rightarrow z_{i,k} \leq e^2 z_{j,k} \text{ (stronger constraints)} \tag{9}$$

$$z_{i,k} \leq e^{\epsilon D_{v_i,v_l}} z_{l,k} \Rightarrow z_{i,k} \leq e^5 z_{l,k} \text{ (weaker constraints)} \tag{10}$$

indicating that $z_{l,k}$ **enforces a weaker constraint on** $z_{i,k}$ **compared to** $z_{j,k}$. Given that $z_{i,k}$ represents a probability measure and therefore cannot exceed 1, the condition $z_{i,k} \leq e^5 z_{l,k}$ becomes irrelevant for instances where $z_{l,k}$ is just marginally greater than 0 (when $z_{l,k} \geq 0.0067$). This is because the upper limit of $z_{i,k} \leq 1$ naturally satisfies the condition $z_{i,k} \leq e^5 z_{l,k}$ under these circumstances. Conversely, the constraint $z_{i,k} \leq e^2 z_{j,k}$ is more stringent and remains applicable unless $z_{j,k} \geq 0.1353$. As $z_{j,k}$ increases beyond 0.1353, the condition $z_{i,k} \leq 1$ is adequate to fulfill the constraint of $z_{i,k} \leq e^2 z_{j,k}$.

Overall, **the insight obtained from Theorem 3.1 and the example in Fig. 2 lead us to focus on a set of "locally relevant locations" that are within a specified path distance threshold from v_i in the Geo-Ind graph.** By focusing the LP problem on this narrowed-down LR location set, we can substantially decrease the computational demands associated with solving the LP problem.

In the next section, we introduce the details of our method.

4 Methodology

In this section, we present *Locally Relevant Geo-Obfuscation (LR-Geo)*, detailing its main concepts and problem formulation in **Section 4.1**, the computational framework in **Section 4.2–4.5**, the theoretical performance analysis in **Section 4.6**, and a discussion of potential threats in **Section 4.7**.

4.1 Locally Relevant Geo-Obfuscation

We consider a scenario where M users $\{1, ..., M\}$ need to obfuscate their locations. Without loss of generality, we assume each user m is located at v_m ($m = 1, ..., M$). Therefore, each user m needs to use the mth row of the obfuscation matrix $\mathbf{Z}$, denoted by $\mathbf{z}_m$ =

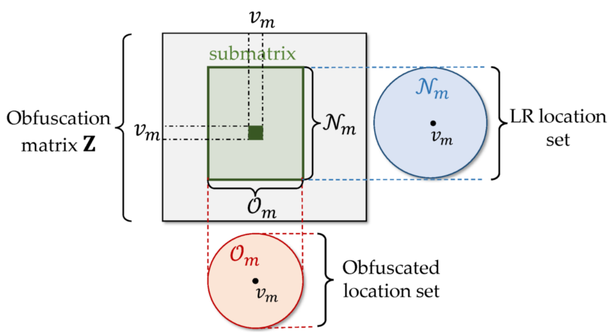


Figure 3: Generate a submatrix instead of the whole matrix.

$[z_{m,1}, ..., z_{m,K}]$, to determine the probability distribution of v_m's obfuscated locations.

Inspired by the insights discussed in Section 3, the underlying concept of *LR-Geo* is to generate an obfuscation matrix focusing solely on the "locally relevant (LR) locations" surrounding each user's actual location v_m to reduce the computational overhead. According to **Theorem 3.1**, within the Geo-Ind graph $\mathcal{G}$, locations with shorter path distances to v_m exhibit stronger connections of their obfuscation vectors to $\mathbf{z}_m$ through Geo-Ind constraints. Therefore, we identify the LR location set based on their path distance to v_m in the Geo-Ind graph, as described in **Definition 4.1**:

Definition 4.1. *(LR location set) The LR location set of v_m, denoted by $\mathcal{N}_m$, is defined as the set of locations with their path distances to v_m no greater than a predetermined threshold Γ:*

$$\mathcal{N}_m = \left\{v_j \in \mathcal{V} \middle| D_{m,j} \leq \Gamma \right\}, \tag{11}$$

where the constant Γ is called the LR distance threshold.

4.1.1 LR Location Set Searching. For each user m, the LR location set $\mathcal{N}_m$ can be found locally by the user. Specifically, given the coordinates of the locations in $\mathcal{V}$ and the neighbor threshold γ, the user first creates the *Geo-Ind graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by checking whether the *Haversine distance* between each pair of locations is no higher than γ. The user then builds a *shortest path tree* rooted at v_i in $\mathcal{G}$ using Dijkstra's algorithm [24]. The shortest path tree provides the path distance between v_i and each $v_j \in \mathcal{V}$, $D_{i,j}$, based on which the user then determines whether v_j should be included in the LR location set $\mathcal{N}_m$ based on Equ. (11).

Time complexity. To build $\mathcal{G}$, the user needs to compare the *Haversine distance* between each pair of locations with γ. This process involves a total of $O(|\mathcal{V}|^2)$ comparisons. The time complexity of building a shortest path tree using Dijkstra's algorithm is $O(|\mathcal{V}|^2)$. Therefore, the time complexity of LR location set identification is $O(|\mathcal{V}|^2 + |\mathcal{V}|^2) = O(|\mathcal{V}|^2)$.

4.1.2 Obfuscation Range. In the original OMG formulation (Equ. (6)(7)), the obfuscation range convers the entire location set $\mathcal{V}$, even though many of these obfuscated locations receive zero probability assignments from the LP algorithm due to their high cost. To further reduce the computation cost, we limit the selection of obfuscated locations to a smaller range. Given a real location v_i, we consider its obfuscated location range as a circle $C(v_i, r_{\text{obf}})$ centered at v_i with radius r_{obf}. Then, the set of the obfuscated locations of v_i, denoted by O_m ($O_m \subseteq \mathcal{V}$), can be calculated by

$$O_m = \left\{v_k \in \mathcal{V} \middle| d_{v_i,v_k} \leq r_{\text{obf}} \right\}. \tag{12}$$

For each obfuscated location $v_k \notin O_m$, we assign a small value ξ to the probability $z_{i,k}$. Here, ξ is a small value and we will specify how to determine ξ in Equ. (19) in Section 4.2.

As Fig. 3 shows, after deriving both $\mathcal{N}_m$ and O_m, the user only needs to download a submatrix of $\mathbf{Z}$, of which the rows and the columns cover $\mathcal{N}_m$ and O_m, respectively.

4.1.3 Problem Formulation. Given each user m's LR location set $\mathcal{N}_m$, we define the user's LR obfuscation matrix as

$$\mathbf{Z}_{\mathcal{N}_m} = \left\{z_{i,k}^{(m)}\right\}_{\mathcal{N}_m \times \mathcal{V}}, \tag{13}$$

which includes the obfuscation vectors of all the relevant locations $\mathcal{N}_m$. The cost caused by $\mathbf{Z}_{\mathcal{N}_m}$ is defined by

$$\mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right) = \sum_{v_i \in \mathcal{N}_m}^{K} \mathbf{c}_i \mathbf{z}_i^{(m)\top} \tag{14}$$

where $\mathbf{c}_i = \left[c_{v_i,v_1}, ..., c_{v_i,v_K}\right]$ are the cost coefficients (defined by Equ. (5)) of the obfuscation vector $\mathbf{z}_i^{(m)} = \left[z_{i,1}^{(m)}, ..., z_{i,K}^{(m)}\right]$.

The objective of each user m is to minimize $\mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right)$ while guaranteeing the Geo-Ind constraints among the obfuscation vectors of relevant locations $\mathcal{N}_m$ and the probability unit measure constraint for each obfuscation vector $\mathbf{z}_i$ in $\mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right)$. Given the LR location set $\mathcal{N}_m$ and the obfuscated location set O_m, we let each user m formulate the *Locally Relevant Obfuscation Matrix Generation (LR-OMG)* problem as the following LP problem:

$$\min \quad \mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right) = \sum_{v_i \in \mathcal{N}_m} \mathbf{c}_i \mathbf{z}_i^{(m)\top} \tag{15}$$

$$\text{s.t.} \quad \frac{z_{i,k}^{(m)}}{z_{j,k}^{(m)}} \leq e^{\epsilon d_{v_i,v_j}}, \forall v_i, v_j \in \mathcal{N}_m \; s.t. \; d_{v_i,v_j} \leq \gamma, \forall v_k \tag{16}$$

$$\sum_k z_{i,k} = 1, \forall v_i \in \mathcal{N}_m \tag{17}$$

$$z_{i,k} = \xi, \forall v_k \notin O_m. \tag{18}$$

4.2 Computation Framework

Considering the limited computation capability of users, like most related works [6, 9, 11], we migrate the computation load of LR-OMG to the servers. Note that, for each user m, directly uploading the LR location set $\mathcal{N}_m$ and the obfuscated location set O_m to the server might cause additional privacy leakage, as both $\mathcal{N}_m$ and O_m can be leveraged to infer the user's real location v_m. As a solution shown in Fig. 4, we let each user m ($m = 1, ..., M$) upload the coefficient of the formulated LP problem (Equ. (15)–(18))) to the server, including the **distance matrix** $\mathbf{D}_{\mathcal{N}_m^2}$ and the **cost matrix** $\mathbf{C}_{\mathcal{N}_m,O_m}$ to the server, instead of $\mathcal{N}_m$ and O_m.

(1) Distance matrix $\mathbf{D}_{\mathcal{N}_m^2}$: The user m calculates the Haversine distance d_{v_i,v_j} between each pair of locations $v_i, v_j \in \mathcal{N}_m$. Then, the user uploads the distance matrix $\mathbf{D}_{\mathcal{N}_m^2} = \left\{d_{v_i,v_j}\right\}_{(v_i,v_j) \in \mathcal{N}_m^2}$ to the server, which uses each distance value d_{v_i,v_j} to establish the Geo-Ind constraints for each pair of decision variables $z_{i,k}$ and $z_{j,k}$ in Equ. (1). Note that $\mathbf{D}_{\mathcal{N}_m^2}$ solely provides information about the relative positions of the locations within $\mathcal{N}_m$ without disclosing their specific coordinates.

It is important to note that if locations are unevenly distributed, an attacker could potentially narrow down the search range of a target user's location based on the user's distance matrix. For

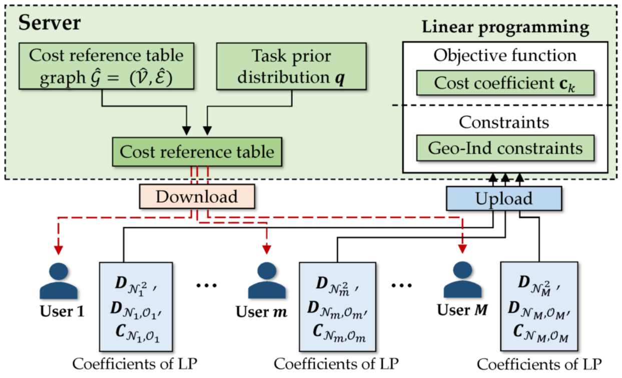

Server
Cost reference table graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}})$
Task prior distribution $\boldsymbol{q}$
Cost reference table
Linear programming
Objective function
Cost coefficient $\mathbf{c}_k$
Constraints
Geo-Ind constraints
Download
Upload
User 1
$\boldsymbol{D}_{\mathcal{N}_1^2}$, $\boldsymbol{D}_{\mathcal{N}_1,\mathcal{O}_1}$, $\boldsymbol{C}_{\mathcal{N}_1,\mathcal{O}_1}$
User m
$\boldsymbol{D}_{\mathcal{N}_m^2}$, $\boldsymbol{D}_{\mathcal{N}_m,\mathcal{O}_m}$, $\boldsymbol{C}_{\mathcal{N}_m,\mathcal{O}_m}$
User M
$\boldsymbol{D}_{\mathcal{N}_M^2}$, $\boldsymbol{D}_{\mathcal{N}_M,\mathcal{O}_M}$, $\boldsymbol{C}_{\mathcal{N}_M,\mathcal{O}_M}$
Coefficients of LP
Coefficients of LP
Coefficients of LP

Figure 4: Obfuscation matrix calculation.

example, if users' locations are represented by the "connections" of the road network (retrievable via OpenStreetMap [23]), users located in downtown areas would tend to generate a distance matrix with lower values due to the higher density of connections in those areas. In this paper, we focus primarily on locations that are evenly distributed. Specifically, similar to existing works [1, 6, 9], we represent the location field using a grid map composed of equally sized grid cells, where each cell is a discrete location. In this case, the distance matrix generated by each user have the same distance values.

(2) Cost matrix $\mathbf{C}_{\mathcal{N}_m,\mathcal{O}_m} = \left\{c_{v_i,v_k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ includes the cost coefficients c_{v_i,v_k} caused by each obfuscated location $v_k \in \mathcal{O}_m$ given each location $v_i \in \mathcal{N}_i$, from which the server can specify the objective function in Equ. (15). Note that, to compute the cost coefficient c_{v_i,v_k} in Equ. (5), it requires knowledge of the target locations $\mathbf{q} = [q_1, ..., q_K]$. However, retaining this information at the user's end presents challenges, primarily due to the dynamically changing target distribution that introduces additional communication costs. Moreover, privacy concerns, particularly regarding the confidentiality of targets (e.g., passengers in Uber-like platform [11]), further complicate the maintenance of such information.

To facilitate the estimation of c_{v_i,v_k}, as Fig. 4 shows, we let each user download a *cost reference* table maintained by the server. This table associates each pair (v_i, v_k) with a value approximating the estimated cost caused by v_k when the real location is v_i, all while keeping the actual target locations confidential. The process of constructing the cost reference table, ensuring the privacy of both users and targets, will be elaborated in **Section 4.5**. In this section, we assume that **users can accurately obtain each** c_{v_i,v_k} **using the cost reference table**. Further analysis of the performance guarantee utilizing the cost reference table will be presented in **Section 4.6**. We will also illustrate that the cost matrix cannot be reversed to unveil the LR location of the user in **Section 4.7** and experiment in Fig. 13.

4.3 Exponential Mechanism Integration

Given the coefficient matrices $\mathbf{D}_{\mathcal{N}_m^2}$ and $\mathbf{C}_{\mathcal{N}_m,\mathcal{O}_m}$ ($m = 1, .., M$), the server can compute $\mathbf{Z}_{\mathcal{N}_m}$ for each user. However, as each LR matrix is generated independently, the obfuscation vectors, such as $\mathbf{z}_i^{(m)}$ and $\mathbf{z}_j^{(n)}$ from different LR matrices $\mathbf{Z}_{\mathcal{N}_m}$ and $\mathbf{Z}_{\mathcal{N}_n}$, may not satisfy the Geo-Ind constraints. Conversely, jointly deriving $\mathbf{Z}_{\mathcal{N}_1}, ..., \mathbf{Z}_{\mathcal{N}_M}$ using only LP not only incurs high computational overhead but also necessitates the disclosure of $\mathcal{N}_m$ and $\mathcal{O}_m$, which should be hidden from the server.

As a solution, we integrate the exponential geo-obfuscation mechanism into LR-Geo, allowing obfuscated locations that follow an exponential distribution to satisfy Geo-Ind across users without imposing the corresponding linear constraints in the LP formulation. Note that it is hard for the server to detect which obfuscation probabilities of the LR locations violate the Geo-Ind constraints, as checking such constraints require users' real locations, which are hidden from the server. Instead, like [17], we let each user locally determine which obfuscated locations should follow the exponential mechanism. We define an indicator matrix $\mathbf{Q} = \left\{q_{i,k}\right\}_{(v_i,v_k)\in\mathcal{V}^2}$ to indicate whether the probability distribution of obfuscated location v_k needs to follow the exponential distribution when the real location is v_i. If $q_{i,k} = 1$, we enforce the following constraint (exponential distribution) for each obfuscated location $v_k \in \mathcal{V}$

$$z_{i,k} = \begin{cases} y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} & \text{if } v_k \in \mathcal{O}_m \\ \xi = y_k e^{-\frac{\epsilon r_{\text{obf}}}{2}} & \text{if } v_k \notin \mathcal{O}_m \end{cases}. \tag{19}$$

where $y_k \geq 0$ is a decision variable. In what follows, we let $\mathbf{y} = [y_1, ..., y_K]$. Note that in Equ. (18) we have set $z_{i,k} = \xi$ when $v_k \notin \mathcal{O}_m$, and here in Equ. (19), we specify $\xi = y_k e^{-\frac{\epsilon r_{\text{obf}}}{2}}$.

Like [17], we adopt a heuristic strategy to determine the indicator matrix $\mathbf{Q}$. In particular, we assign $q_{i,k} = 1$ when the distance d_{v_i,v_k} exceeds r_{exp}, where r_{exp}, a predefined threshold, is less than or equal to the obfuscation range r_{obf}. This approach is based on the rationale of applying the exponential mechanism more extensively to obfuscated locations that are significantly distant from the actual location. Such locations are often associated with lower probability values, thereby minimizing their influence on the expected cost. Given that LP-based methods tend to yield lower costs, incorporating an exponential mechanism for these distant locations can reduce cost impacts more effectively. However, our framework can accommodate alternative methods for determining $\mathbf{Q}$.

In our experimental setup in Section 5, we set $r_{\text{obf}} = 4$km, resulting in a Geo-Ind violation ratio of up to 0.13% (as shown in Fig. 10), i.e., the Geo-Ind constraint is likely to be satisfied with a sufficiently high ratio.

Proposition 4.1. *[17] Given any $y_k \in \mathbb{R}^+$, if the constraints in Equ. (19) are satisfied, then for each pair of $z_{i,k}$ and $z_{j,k}$ with $q_{i,k} = q_{j,k} = 1$, the Geo-Ind constraint $z_{i,k} - e^{\epsilon d_{v_i,v_j}} z_{j,k} \leq 0$ is satisfied (The detailed proof can be found in the proof of* ***Proposition 1*** *in [17]).*

To enable the server to establish the constraints of the exponential distribution in Equ. (19), each user m computes the distance matrix $\mathbf{D}_{\mathcal{N}m,\mathcal{O}m} = \left\{d_{v_i,v_k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ and uploads the matrix to the server. It's important to note that $\mathbf{D}_{\mathcal{N}_m,\mathcal{O}_m}$ contains only the relative distances between locations within $\mathcal{N}_m$ and $\mathcal{O}_m$, and does not provide enough information to deduce the specific locations in either $\mathcal{N}_m$ or $\mathcal{O}_m$.

Problem formulation. After collecting the coefficient matrices $\mathbf{D}_{\mathcal{N}_m^2}$, $\mathbf{D}_{\mathcal{N}_m,\mathcal{O}_m}$ and $\mathbf{C}_{\mathcal{N}_m,\mathcal{O}_m}$ ($m = 1, .., M$) and adding the constraints of exponential mechanism in Equ. (19), we can formulate the following *Central LR-Geo (CLR-Geo)* problem at the server side:

$$\begin{aligned} \min \quad & \textstyle\sum_{m=1}^{M} \mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right) && (20)\\ \text{s.t.} \quad & \text{Equ. (16)(17) are satisfied for each } \mathcal{N}_m && (21)\\ & \text{Equ. (19) is satisfied } \forall v_k \in \mathcal{V} \text{ with } q_{i,k}=1. && (22) \end{aligned}$$

4.4 Benders' Decomposition to Enhance Computation Efficiency

4.4.1 Problem reformulation of CLR-Geo. We rewrite the objective function in Equ. (20) as

$$\begin{aligned} \textstyle\sum_{m=1}^{M} \mathcal{L}\left(\mathbf{Z}_{\mathcal{N}_m}\right) &= \textstyle\sum_{m=1}^{M}\sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{V}} \underbrace{c_{v_i,v_k} z_{i,k} q_{i,k}}_{\text{following Equ. (19)}} \\ &+ \textstyle\sum_{m=1}^{M}\sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{V}} c_{v_i,v_k} z_{i,k}(1-q_{i,k}) \\ &= \textstyle\sum_{k=1}^{K}\alpha_k y_k + \sum_{m=1}^{M} \mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} \qquad (23) \end{aligned}$$

where each $\alpha_k = \sum_{m=1}^{M}\sum_{v_i\in\mathcal{N}_m} q_{i,k} c_{v_i,v_k} e^{-\frac{\epsilon d_{v_i,v_k}}{2}}$ is a constant, and in $\mathbf{c}'_{\mathcal{N}_m}$, each $c'_{i,k} = c_{v_i,v_k} z_{i,k}(1-q_{i,k})$.

We rewrite the constraints of Equ. (16)(17) and Equ. (19) as

$$\mathbf{A}_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} + \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y}) \geq \mathbf{b}_{\mathcal{N}_m} \qquad (24)$$

where

$$\mathbf{A}_{\mathcal{N}_m} = \begin{bmatrix} \mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m} \\ \mathbf{A}^{\text{unit}}_{\mathcal{N}_m} \\ -\mathbf{A}^{\text{unit}}_{\mathcal{N}_m} \end{bmatrix}, \mathbf{B}_{\mathcal{N}_m} = \begin{bmatrix} \mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m} \\ \mathbf{B}^{\text{unit}}_{\mathcal{N}_m} \\ -\mathbf{B}^{\text{unit}}_{\mathcal{N}_m} \end{bmatrix}, \mathbf{b}_{\mathcal{N}_m} = \begin{bmatrix} \mathbf{b}^{\text{GeoI}}_{\mathcal{N}_m} \\ \mathbf{b}^{\text{unit}}_{\mathcal{N}_m} \\ -\mathbf{b}^{\text{unit}}_{\mathcal{N}_m} \end{bmatrix} \qquad (25)$$

and (1) $\mathbf{z}'_{\mathcal{N}_m}$ (resp. $\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})$) includes the obfuscation probabilities $z_{i,k}$ *not adhering to* (resp. *adhering to*) the exponential mechanism, where $v_i \in \mathcal{N}_m$ (note that each entry in $\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})$ follows Equ. (19), therefore the vector is written as a function of $\mathbf{y}$).
(2) $\mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m}$ (resp. $\mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m}$) denotes the coefficient matrix of the *Geo-Ind constraints* for $\mathbf{z}'_{\mathcal{N}_m}$ (resp. $\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})$);
(3) $\mathbf{A}^{\text{unit}}_{\mathcal{N}_m}$ (resp. $\mathbf{B}^{\text{unit}}_{\mathcal{N}_m}$) denotes the coefficient matrix of the *unit measure constraints* for $\mathbf{z}'_{\mathcal{N}_m}$ (resp. $\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})$);
(4) $\mathbf{b}^{\text{GeoI}}_{\mathcal{N}_m}$ and $\mathbf{b}^{\text{unit}}_{\mathcal{N}_m}$ are the right hand side coefficient vectors of the *Geo-Ind constraints* and the *unit measure constraints*, respectively.

As Fig. 5 shows, the constraint matrix of the reformulated problem has a *block ladder structure*, lending the problem well to *Benders decomposition (BD)* [25]. Due to the limit of space, we list the detailed formulations of $\mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m}$, $\mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m}$, $\mathbf{A}^{\text{unit}}_{\mathcal{N}_m}$, $\mathbf{B}^{\text{unit}}_{\mathcal{N}_m}$, and coefficient vectors $\mathbf{b}^{\text{GeoI}}_{\mathcal{N}_m}$, and $\mathbf{b}^{\text{unit}}_{\mathcal{N}_m}$ in **Section A.1** in Appendix.

4.4.2 High level idea of the algorithm. Benders' decomposition is an optimization method that breaks a complex problem into two simpler stages: a *master program* and a set of *subproblems.*
Stage 1: Master Program (MP) - The MP focuses on deriving the set of decision variables $\{y_1, ..., y_K\}$. It simplifies the problem by replacing the cost terms $\mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m}$ associated with data perturbation in each user m with a single decision variable w_m. The MP is formulated as a linear programming problem that minimizes the sum cost of the primary decision variables $\sum_{k=1}^{K}\alpha_k y_k$ and the substituted cost terms $\sum_{m=1}^{M} w_m$. The MP iteratively guesses the values of w_m and uses them in the subsequent stage to guide optimization.
Stage 2: Subproblems - Each subproblem sub$_m$ validates the guessed values of w_m generated by the MP considering its own

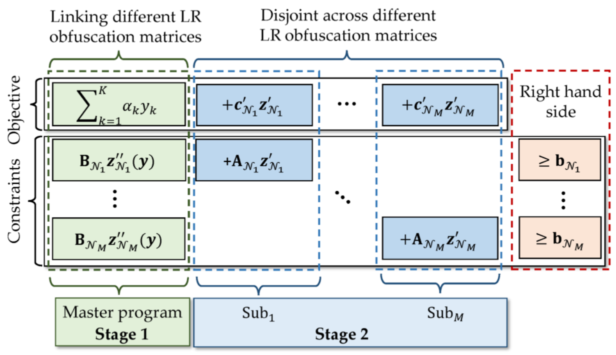

Linking different LR obfuscation matrices
Disjoint across different LR obfuscation matrices
Objective
Constraints
$\sum_{k=1}^{K}\alpha_k y_k$
$+\mathbf{c}'_{\mathcal{N}_1}\mathbf{z}'_{\mathcal{N}_1}$
$+\mathbf{c}'_{\mathcal{N}_M}\mathbf{z}'_{\mathcal{N}_M}$
Right hand side
$\mathbf{B}_{\mathcal{N}_1}\mathbf{z}''_{\mathcal{N}_1}(\mathbf{y})$
$+\mathbf{A}_{\mathcal{N}_1}\mathbf{z}'_{\mathcal{N}_1}$
$\geq \mathbf{b}_{\mathcal{N}_1}$
$\mathbf{B}_{\mathcal{N}_M}\mathbf{z}''_{\mathcal{N}_M}(\mathbf{y})$
$+\mathbf{A}_{\mathcal{N}_M}\mathbf{z}'_{\mathcal{N}_M}$
$\geq \mathbf{b}_{\mathcal{N}_M}$
Master program
Stage 1
Sub$_1$
Sub$_M$
Stage 2

Figure 5: Block ladder structure of the CLR-Geo problem.

constraint $\mathbf{A}_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} \geq \mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})$. If the guessed value is not optimal, the subproblem suggests a new constraint (or cut) to refine the MP's future guesses. As shown in Fig. 5, the constraint matrix $\mathbf{A}_{\mathcal{N}_m}$ of different subproblems are disjoint, allowing them to validate the guessed values of w_m in parallel. This iterative process continues, with each subproblem helping to progressively narrow the solution space until the optimal solution is found.

The method iteratively refines the solutions by collecting these cuts from subproblems, guiding the MP towards the optimal solution over time. This approach is particularly useful for problems where the main challenge is efficiently managing computational complexity by breaking it down into more manageable parts. Due to the limit of space, the detailed description of the algorithm is given in Section C in Appendix.

4.4.3 Time complexity of BD. The MP and the subproblems are both formulated as LP problems, with time complexity depending on the number of decision variables [26]. To facilitate analysis, we use a monotonically increasing function $O(T(n))$ to represent the time complexity of LP given the number of decision variables n.

In each iteration, the MP in Stage 1 has $O(T(|\mathcal{V}| + M))$ time complexity, including $|\mathcal{V}|$ decision variables to determine $y_1, \ldots, y_K$ and M decision variables $w_1, \ldots, w_M$ to "guess" the utility loss of all the users. Each subproblem m in Stage 2 (run in parallel) has $O(T(|\mathcal{V}||\mathcal{N}_m|^2))$ time complexity to validate the guessed value of w_m generated by the MP. Since Stage 2 is terminated only after all subproblems are completed, its time complexity is

$$O(\max_{l} T(|\mathcal{V}||\mathcal{N}_m|^2)) = O(T(|\mathcal{V}| \max_{m} |\mathcal{N}_m|^2)), \qquad (26)$$

assuming all the subproblems are run in parallel. Hence, the total computation time of each iteration is given by $O(T(|\mathcal{V}| + M) + T(|\mathcal{V}| \max_m |\mathcal{N}_m|^2))$. Considering that both $O(|\mathcal{V}| + M)$ and $O(|\mathcal{V}| \max_m |\mathcal{N}_m|^2)$ are much smaller than the number of decision variables in the original OMG, $O(|\mathcal{V}|^3)$, the time complexity of each iteration of BD is significantly lower than that of the original PMO. The remaining question is *how many iterations are needed to converge the solution to the optimal.*

Convergence of Benders decomposition.

Proposition 4.2. *(*Upper and lower bounds of CLR-Geo's optimal*) [25] (1) Because the MP relaxes the constraints of the original CLR-Geo (Equ. (20)–(22)), the optimal solution of the MP (Equ. (70) – (72)) offers a **lower bound** of the optimal solution of CLR-Geo.*

(2) If the solution of the subproblems (Equ. (75)-(76)) exists, then by combining the solutions of the subproblems and the solution of the MP, we obtain a feasible solution of CLR-Geo, which is an ***upper bound*** *of the optimal solution.*

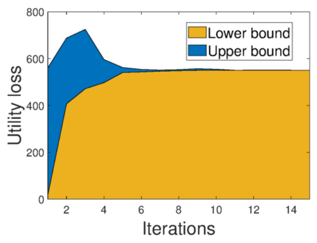


Figure 6: An example of Benders' convergence.

The optimal solution for the CLR-Geo necessarily resides within the interval demarcated by the upper and lower bounds as delineated in **Proposition 4.2**. The narrower this interval, the nearer the solution derived from the Benders' Decomposition is to the optimal solution. Fig. 6 illustrates the evolution of these bounds throughout the iterative process. Considering a prolonged convergence tail, we opt to conclude the algorithm once the discrepancy between the optimal upper and lower bounds diminishes to less than a specified margin, ξ (e.g., we set $\xi = 0.01$km in our experiment in Section 5).

4.5 Cost Matrix Estimation

In Sections 4.2 and 4.4, we assumed that each user m knows the cost matrix $\mathbf{C}_{\mathcal{N}_m,\mathcal{O}_m}$. We now relax this assumption and elucidate the methodology by which users, with the assistance of the server, can estimate $\mathbf{C}_{\mathcal{N}_m,\mathcal{O}_m}$.

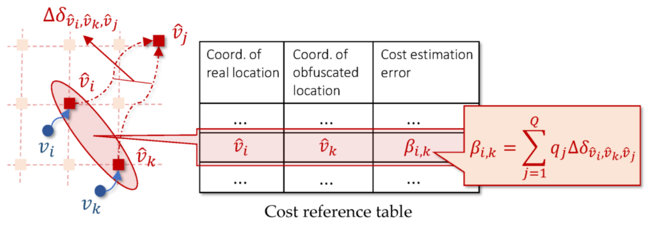
According to Equ. (5), the calculation of each c_{v_i,v_k} requires

- the coordinates of the locations in $\mathcal{N}_m$ and $\mathcal{O}_m$ (to derive cost error δ_{v_i,v_k,v_l}), which are **known by the user but unknown by the server**;
- the coordinates of the target locations (to derive cost error d_{v_i,v_k,v_l}) and the targets' prior distribution $\mathbf{q}$, which are **known by the server but unknown by the user**.

Since both the server and the user possess only partial information required to compute $\mathbf{c}_k$, a "cooperative" approach is employed to calculate $\mathbf{c}_k$ through the exchange of intermediate values between the two parties. *Throughout the calculation of* $\mathbf{c}_k$*, the server must remain unaware of* $\mathcal{N}_m$ *and* $\mathcal{O}_m$ *to protect privacy.* Since the server has the global information including the traveling cost between any pair of locations and the target distribution, we let the server generate a *cost reference table* to assist the user estimate $\mathbf{c}_k$.

4.5.1 *Cost reference table.*

The server constructs a discrete set of locations $\hat{\mathcal{V}}$, which is sufficiently dense to ensure that for any given location pair (v_j, v_k) from the sets $\mathcal{N}_m$ and $\mathcal{O}_m$, respectively, users can identify a corresponding pair $(\hat{v}_j, \hat{v}_k)$ within $\hat{\mathcal{V}}$ that is closely approximated to (v_j, v_k). This approximation is then used to estimate cost coefficients. To establish $\hat{\mathcal{V}}$, the server employs a grid map to discretize the location field, such that locations within the same grid cell are indistinguishable. For specific applications involving constrained user mobility, such as vehicles' mobility, the server might alternatively divide the road network into segments, treating each as a distinct location [11]. Given the superior computational resources available to servers compared to those of users, $\hat{\mathcal{V}}$ can


$\Delta\delta_{\hat{v}_i,\hat{v}_k,\hat{v}_j}$
$\hat{v}_j$
$\hat{v}_i$
v_i
$\hat{v}_k$
v_k
Coord. of real location
Coord. of obfuscated location
Cost estimation error
...
$\hat{v}_i$
$\hat{v}_k$
$\beta_{i,k}$
$\beta_{i,k} = \sum_{j=1}^{Q} q_j \Delta\delta_{\hat{v}_i,\hat{v}_k,\hat{v}_j}$
Cost reference table
Figure 7: Cost reference table.

afford to feature a finer granularity of location discretization than that of $\mathcal{N}_m$ and $\mathcal{O}_m$.

Table format. As Fig. 7 shows, each row of the cost reference table includes the coordinates of an "approximated" real location $\hat{v}_i$, an "approximated" obfuscated location $\hat{v}_k$, and the expected cost $\beta_{i,k}$ given the real and the obfuscated locations $\hat{v}_i$ and $\hat{v}_k$, respectively. The expectation of the cost $\beta_{i,k}$ is to take over all possible target locations $\beta_{i,k} = \sum_{j=1}^{Q} q_j \delta_{\hat{v}_i,\hat{v}_k,\hat{v}_j}$, where q_j is the probability that the target's nearest location in $\hat{\mathcal{V}}$ is $\hat{v}_j$.

In what follows, we use $\hat{\mathbf{C}}_{\mathcal{N}_m,\mathcal{O}_m} = \left\{\hat{c}_{v_i,v_k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ to denote the cost matrix estimated by the cost reference table. When a user estimates each $\hat{c}_{v_i,v_k}$ in $\hat{\mathbf{C}}_{\mathcal{N}_m,\mathcal{O}_m}$, the user first finds the nearest locations of v_i and v_k in $\hat{\mathcal{V}}$, denoted by $\hat{v}_i$ and $\hat{v}_k$ respectively, and then calculates the cost estimation error $\hat{c}_{v_i,v_k}$ by

$$\hat{c}_{v_i,v_k} = p_i \left(\beta_{i,k} + d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k}\right), \tag{27}$$

which gives an upper bound of the real c_{v_i,v_k} (see **Lemma B.1** in Appendix).

Time complexity of $\hat{\mathbf{C}}_{\mathcal{N}_m,\mathcal{O}_m}$'s estimation. The cost estimation traverses the $|\mathcal{N}_m||\mathcal{O}_m|$ pairs of locations in $\mathcal{N}_m \times \mathcal{O}_m$, and for each location pair, it needs to find their closest locations in $\hat{\mathcal{V}}$, taking $O(|\hat{\mathcal{V}}|)$ comparisons. Therefore, the time complexity of cost estimation is $O(|\mathcal{N}_m||\mathcal{O}_m||\hat{\mathcal{V}}|)$.

4.5.2 *Cost reference table generation.*

To generate a cost reference table, the server first creates a *weighted directed graph* $\hat{\mathcal{G}} = \left(\hat{\mathcal{V}}, \hat{\mathcal{E}}\right)$ to describe the traveling cost between locations in the discrete location set $\hat{\mathcal{V}}$, where each pair of *adjacent* locations $\hat{v}_i$ and $\hat{v}_j$ are connected by an edge $\hat{e}_{i,j} \in \hat{\mathcal{E}}$. Each $\hat{e}_{i,j} \in \hat{\mathcal{E}}$ is assigned a weight $d_{\hat{v}_i,\hat{v}_j}$, reflecting the traveling cost from $\hat{v}_i$ to $\hat{v}_j$. The server then builds the SP tree rooted at each target location $\hat{v}_j \in \hat{\mathcal{V}}$ in $\hat{\mathcal{G}}$, based on which it calculates the shortest path distance $D_{\hat{v}_i,\hat{v}_j}$ ($D_{\hat{v}_i,\hat{v}_k}$ resp.) from each location $\hat{v}_i$ ($\hat{v}_k$ resp.) to $\hat{v}_j$, and derives $\delta_{\hat{v}_i,\hat{v}_k,\hat{v}_j}$ using Equ. (3). Finally, the server calculates $\beta_{i,k}$ for each $(\hat{v}_i, \hat{v}_k)$ using their cost estimation errors $\delta_{\hat{v}_i,\hat{v}_k,\hat{v}_j}$

Time complexity of cost reference table generation. The construction of each SP tree can be achieved by Dijkstra's algorithm, which has a time complexity of $O(|\hat{\mathcal{V}}|^2)$ [24]. For each designated target location, the server is required to generate an SP tree, culminating in a collective computational effort of $O(|\hat{\mathcal{V}}|^3)$ operations. Furthermore, the computation of $\beta_{i,k}$ is called $|\hat{\mathcal{V}}|^2$ times, with each instance necessitating $O(|\hat{\mathcal{V}}|)$ operations, thereby rendering its time complexity to be $O(|\hat{\mathcal{V}}|^3)$. Consequently, the overall time complexity associated with the creation of the requisite table is determined to be $O(|\hat{\mathcal{V}}|^3 + |\hat{\mathcal{V}}|^3) = O(|\hat{\mathcal{V}}|^3)$, indicating significant computational demand for these operations.

4.5.3 Cost reference table size reduction. We can to further reduce the location set $\hat{\mathcal{V}}$ while guaranteeing the accuracy of the cost coefficient $\hat{\mathbf{c}}_k$ estimation. To achieve this, we define $\hat{\mathcal{V}}$ in a circular region, referred to as C_{cr}, so that $\hat{\mathcal{V}}$ consists of locations within this circle. Consequently, the accuracy of $\hat{\mathbf{c}}_k$ estimation can be guaranteed if C_{cr} encompasses both $\mathcal{N}_m$ and $\mathcal{O}_m$.

Note that if C_{cr} covers $\mathcal{N}_m$ and $\mathcal{O}_m$ only at a minimum level, the range of $\mathcal{N}_m$ and $\mathcal{O}_m$ can be possibly disclosed to the server. Therefore, instead of only covering $\mathcal{N}_m$ and $\mathcal{O}_m$, we allow the user to request a larger C_{cr}. Initially, the user randomly selects a location v_a from the LR location set $\mathcal{N}_m$ by following a uniform distribution. Subsequently, the user reports a circle $C(v_a, \max 2\Gamma, \Gamma + r_{\mathrm{obf}})$ to the server as the *requested range of cost reference table*, with v_a serving as the center.

Proposition 4.3. *The circle $C(v_a, \max\{2\Gamma, \Gamma + r_{\mathrm{obf}}\})$ covers all the locations in both $\mathcal{N}_m$ and $\mathcal{O}_m$.*

Moreover, according to the requested range $C(v_a, \max\{2\Gamma, \Gamma + r_{\mathrm{obf}}\})$ and how the user selects the location v_a, the server can only infer that the user's real location is in the circle $C(v_a, \Gamma)$, where $\Gamma > \gamma$, indicating that the user's location is well hidden from the server.

4.6 Performance Analysis

In this section, we provide the theoretical analysis of the performance for the LR-Geo solution (Equ. (20)–(22)), including the *privacy guarantee* in **Theorem 4.4**, the lower bound and the upper bound of *expected cost* in **Theorem 4.7** and **Theorem 4.6**, respectively. The detailed proofs of these theorems can be found in **Section B in Appendix**.

4.6.1 Privacy guarantee. We first prove that the chosen obfuscated locations adhering to the exponential distribution constraints (Equ. (19)) meet Geo-Ind constraints across users even though their geo-obfuscation is calculated in a relatively independent manner.

Theorem 4.4. *(Privacy guarantee) Given two locations, v_i from user n's LR location set $\mathcal{N}_n$ and v_j from user m's LR location set $\mathcal{N}_m$, if both locations satisfy the condition $q_{i,k} = q_{j,k} = 1$ (indicating that $z^{(n)}_{i,k}$ and $z^{(m)}_{j,k}$ satisfy the exponential distribution constraints), then their obfuscation distributions still satisfy the (ϵ, γ)-Geo-Ind constraints,*

$$z^{(n)}_{i,k} - e^{\epsilon d_{v_i,v_j}} z^{(m)}_{j,k} \leq 0, \ \forall v_k \in \mathcal{V}. \tag{28}$$

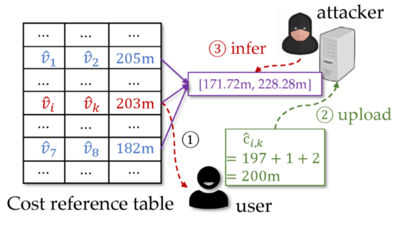
Proposition 4.5. *For any LR location v_i of a User n, i.e., $v_i \in \mathcal{N}_n$, let $\mathcal{A}_i$ denote the set of obfuscated locations of v_i that adhere to the exponential distribution constraints. Then:*

(1) *For each pair of Users n and m, the Geo-Ind constraint violation ratio is upper bounded by:*

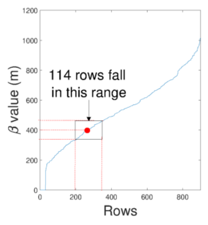
$$1 - \frac{2\sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} |\mathcal{A}_i \cap \mathcal{A}_j| + (|\mathcal{N}_n|^2 + |\mathcal{N}_m|^2 - |\mathcal{N}_n| - |\mathcal{N}_m|)|\mathcal{V}|}{(|\mathcal{N}_n| + |\mathcal{N}_m|)(|\mathcal{N}_n| + |\mathcal{N}_m| - 1)|\mathcal{V}|}. \tag{29}$$

(2) *For all users, the Geo-Ind constraint violation ratio is upper bounded by:*

$$1 - \frac{2\sum_{n=1}^{M}\sum_{m=n+1}^{M}\sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} |\mathcal{A}_i \cap \mathcal{A}_j| + \sum_{n=1}^{M}\left(|\mathcal{N}_n|^2 - |\mathcal{N}_n|\right)|\mathcal{V}|}{\sum_{n=1}^{M}|\mathcal{N}_n|\left(\sum_{n=1}^{M}|\mathcal{N}_n| - 1\right)|\mathcal{V}|}. \tag{30}$$



(a) A potential inference attack carried out by an attacker using $\hat{\mathrm{C}}_{\mathcal{N}_m,\mathcal{O}_m}$



(b) Example in experiment.

Figure 8: Potential inference attack using estimated cost matrix $\hat{\mathrm{C}}_{\mathcal{N}_m,\mathcal{O}_m}$ and the cost reference table.

Remark According to Proposition 4.5(1), the Geo-Ind violation ratio for each user i also depends on the proportion of obfuscated locations from other users that follow the exponential mechanism. For example, there is no Geo-Ind guarantee for a user when no other user applies exponential mechanism. Hence, our system encourages users to cooperate to achieve low Geo-Ind violation ratio for both sides. While (ϵ, γ)-Geo-Ind is not guaranteed between $z^{(n)}_{i,k}$ and $z^{(m)}_{j,k}$ if one of them doesn't follow the exponential mechanism, our experimental findings in Fig. 11 demonstrate that unselected obfuscated locations still possess a high probability (99.81% on average) of meeting Geo-Ind constraints.

4.6.2 Lower bound and upper bound of the expected cost. Given the LR location set $\mathcal{N}_m$, we formulate the following relaxed LR-Geo problem:

$$\min \quad \sum_{m=1}^{M} \mathcal{L}\left(\mathrm{Z}_{\mathcal{N}_m}\right) \tag{31}$$

$$\text{s.t.} \quad \text{Equ. (16)(17) are satisfied for each } \mathcal{N}_m \tag{32}$$

Theorem 4.6. *(Upper bound of the minimum expected cost) Using the estimated cost coefficient $\hat{c}_{v_i,v_k}$ in Equ. (27), the solution of the CLR-Geo problem in Equ. (20)–(22) offers an upper bound of the minimum expected cost.*

Next, we define another cost estimation $\tilde{c}_{v_i,v_k}$ by

$$\tilde{c}_{v_i,v_k} = p_i\left(\beta_{i,k} - d_{v_i,\hat{v}_i} - d_{v_k,\hat{v}_k}\right), \tag{33}$$

which gives a lower bound of the real c_{v_i,v_k} (see **Lemma B.2 in Appendix**).

Theorem 4.7. *Using the estimated cost in Equ. (33), the solution of the relaxed LR-Geo problem in Equ. (31)–(32) offers a lower bound of the minimum expected cost.*

4.7 Discussion of Potential Inference Using Estimated Cost Matrix

In this part, we illustrate that it is hard to infer the locations in $\mathcal{N}_m$ and $\mathcal{O}_m$ using the estimated cost matrix $\hat{\mathrm{C}}_{\mathcal{N}_m,\mathcal{O}_m}$.

Fig. 8(a) gives an example, where the user calculates the cost coefficient $\hat{c}_{v_i,v_k} = \beta_{i,k} + d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k} = 197 + 1 + 2 = 200\text{m}$ (①), and uploads $\hat{c}_{v_i,v_k}$ to the server (②). After receiving $\hat{c}_{v_i,v_k}$, a potential attack by the attacker is to find the corresponding β value in the cost reference table, of which the estimated cost coefficient by a user can be possibly 200m according to Equ. (27).

Note that both $d_{v_i,\hat{v}_i}$ and $d_{v_k,\hat{v}_k}$ in Equ. (27) are unknown by the server, while the server can drive the maximum possible value of

$d_{v_i,\hat{v}_i}$ and $d_{v_k,\hat{v}_k}$, denoted by $\delta_{\max}$ (e.g. $\delta_{\max} = 28.28$ in Fig. 8(a)), based on the distribution of $\hat{\mathcal{V}}$. In this case, the server can derive that the matched β value is in the interval $[\hat{c}_{v_i,v_k} - \delta_{\max}, \hat{c}_{v_i,v_k} + \delta_{\max}] = [171.72, 228.28]$ (③), which might cover other β values in the cost reference table, like 205m and 182m in Fig. 8(a). In this case, the attacker cannot identify which β is the true $\beta_{i,k}$ in the interval, and the more β values fall in the interval, the more difficult for the attacker to find the true $\beta_{i,k}$.

Fig. 8(b) gives another example of how many β values can possibly match an estimated cost coefficient using the real world map information (more details can be found in our experiment in Section 5). In this example, the server creates a cost reference table covering 900 locations in $\hat{\mathcal{V}}$ by a grid map with each cell size equal to 100m. The maximum distance from a location in $\mathcal{N}_m$ and $\mathcal{O}_m$ to its nearest location in $\hat{\mathcal{V}}$ is 70.7m. Given an estimated cost coefficient $\hat{c}_A$ = 400m, its corresponding β_A is in the interval [400m−70.7m, 400m+70.7m], where 114 β values fall in this interval. On average, each cost coefficient is matched by 102.98 rows of the cost reference table. The more comprehensive experimental results can be found in Fig. 13 in **Section 5**.

5 Performance Evaluation

In this section, we conduct a simulation using real-world map information to evaluate the performance of LR-Geo in terms of computation efficiency, privacy, and cost, with the comparison of several benchmarks [3, 11, 17]. Specifically, we focus on the application of vehicular spatial crowdsourcing [11], such as Uber like platform [27], where vehicles need to physically travel to a disignated location to complete the task[1].

We first introduce the settings of the experiment in **Section 5.1**, and then evaluate the performance of different geo-obfuscation methods in **Section 5.2 and Section 5.3**.

5.1 Settings

5.1.1 Dataset. We selected the city "*Rome, Italy*" as the target region (the bounding area with coordinate ($lat = 41.66, lon = 12.24$) as the south-west corner, and coordinate ($lat = 42.10, lon = 12.81$) as the north-east corner). Similar to existing works [1, 6, 9], we approximate the location field by partitioning the entire target region into a 40 × 40 grid. Each grid cell represents a discrete location within the location set, and the distances between cells are calculated based on the travel distance between the centers of the cells. To calculate the travel distances, we retrieve the road map information of Rome, including both the node set and edge set, using OpenStreetMap [23]. We compute the shortest path distances between cell centers on the road map using Dijkstra's algorithm [24]. Additionally, we assume a uniform distribution of targets.

5.1.2 Benchmarks. We compare LR-Geo with the following benchmarks, which are all based on Geo-Ind:

(1) *LP-based geo-obfuscation (labeled as "LP")* [11]: LP considers the network-constrained mobility features of the vehicles and employs LP formulated in Equ. (6)(7) to minimize the expected cost.
(2) *Laplacian noise (labeled as "Laplace")* [3]: Laplace adds a polar Laplacian noise ϕ to the real location, i.e., $v_i + \phi$ and approximate it by the closest location $v_k = \arg\min_{v \in \mathcal{V}} d_{v,v_i+\phi}$.
(3) *Exponential mechanism (labeled as "*ExpMech*")* [3]: In ExpMech, the probability distribution of the obfuscated location of each real location v_i follows a polar Laplace distribution $z_{i,k} \propto e^{-\epsilon c_{v_i,v_k}/2}$.
(4) *"*ConstOPTMech*" or "*ConstOPT*"* [17]: Like our approach, ConstOPT applies the exponential distribution constraint for a subset of the obfuscation probabilities and uses LP for the optimization of the remaining obfuscation probabilities, to balance the utility and scalability of the data perturbation method.
(5) *"*LR-Geo-F*"*: In addition to the four benchmarks mentioned above, we also consider LR-Geo with the neighbor threshold γ set to an infinite value. In this case, the Geo-Ind Graph is fully connected, which aligns with existing works such as [6, 9, 11], which do not require Geo-Ind to be satisfied only between locations that are within a distance smaller than γ. We use "LR-Geo-F" to label LR-Geo with $\gamma = \infty$, where "F" stands for "fully connected Geo-Ind Graph".

5.1.3 Metrics. We measure the following metrics to evaluate the performance of our method and the benchmarks:

(i) *Computation time*, which is defined as the amount of time to calculate an obfuscation matrix. The experiments are performed by a desktop with 13th Gen Intel Core i7 processor, 16 cores. We used the Matlab LP toolbox `linprog`, with the algorithm "`dual-simplex`" [28] to solve LP.
(ii) *Expected cost* $\mathcal{L}(\mathbf{Z})$: $\mathcal{L}(\mathbf{Z})$ is defined in Equ. (4), meaning the expected estimation error of traveling cost caused by $\mathbf{Z}$.
(iii) *Geo-Ind violation (GV) ratio*, which is defined as the ratio:

$$\frac{\#\text{ of }(z_{i,k}, z_{j,k})\text{ violating Geo-Ind in Equ. (1)}}{\#\text{ of }(z_{i,k}, z_{j,k})\text{ that should satisfy Geo-Ind in Equ. (1)}}. \tag{34}$$

The GV ratio reflects how the derived obfuscation matrix can achieve Geo-Ind. In the following experiment, by default, we set ϵ by 10.0km^{-1}, the cell size of the cost reference table by 0.1km, the LR distance threshold Γ by 20km.

5.2 Computation Efficiency

5.2.1 Comparison with the benchmarks. Table 1 compares the computational times for LR-Geo against four benchmark methods, where the number of locations K equals 100, 200, 300, and 400, respectively. The table reveals that **while LR-Geo has higher computational time compared to Laplace and ExpMech, it significantly outperforms both LP and ConstOPT in terms of efficiency**.

Specifically, at $K = 200$, LR-Geo demonstrates a remarkable reduction in computation time, showing a decrease of 99.51% and 98.12% compared to LP and ConstOPT, respectively. For both LP and ConstOPT, computation times exceed the 1800-second threshold when $K \geq 300$. This enhanced efficiency of LR-Geo is due to its strategic approach of confining the set of locations under consideration to LR locations only. Conversely, the alternative LP-based methods evaluate every location within the targeted area, resulting in substantial computational overhead.

Additionally, the computation time of LR-Geo is 87.71% lower than that of LR-Geo-F, as LR-Geo-F imposes Geo-Ind constraints for all pairs of locations, not just those with a distance smaller than γ. This leads to a higher number of linear constraints in the LP formulation and, consequently, greater computational overhead. Nevertheless, the computation time of LR-Geo-F is still 99.26% and 97.18% lower than that of LP and ConstOPT at $K = 200$, respectively.

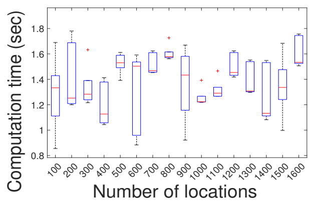
[1]The MATLAB source code of LR-Geo is available at: https://github.com/chenxiunt/LocalRelevant_Geo-Obfuscation

Table 1: Computation time (seconds) of different methods. Mean±1.96× standard deviation.

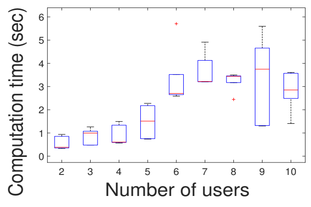
	Problem size			
Methods	$K = 100$	$K = 200$	$K = 300$	$K = 400$
LR-Geo	1.28±1.35	1.42± 0.63	1.34±0.64	1.20±0.85
LR-Geo-F	1.63±1.00	2.14± 0.71	2.42±0.85	3.56±1.07
LP	27.31±10.31	287.50±48.28	≥ 1800	≥ 1800
ConstOPT	3.69±1.92	75.88±10.81	≥ 1800	≥ 1800
Laplace	≤0.005	≤0.005	≤0.005	≤0.005
ExpMech	≤0.005	≤0.005	≤0.005	≤0.005

Table 2: Cost (kilometers) of different methods. Mean±1.96× standard deviation.

	Problem size			
Methods	$K = 100$	$K = 200$	$K = 300$	$K = 400$
LR-Geo	0.36±0.04	0.36±0.07	0.35±0.03	0.37±0.05
LR-Geo-F	0.38±0.03	0.36±0.05	0.38±0.06	0.40±0.05
ConstOPT	0.35±0.03	0.34±0.04	——	——
LP	0.33±0.02	0.33±0.03	——	——
Laplace	0.81±0.02	0.78±0.02	0.80±0.06	0.79±0.01
ExpMech	0.67±0.04	0.64±0.07	0.68±0.12	0.70±0.05
Lower bound	0.29±0.08	0.30±0.06	0.31±0.08	0.30±0.05

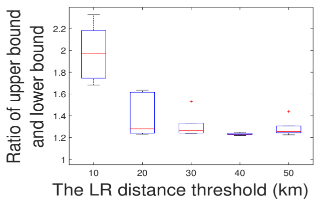


(a) Different location set sizes

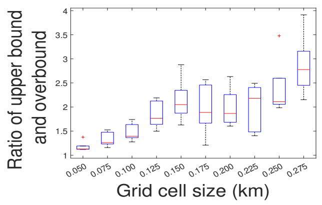


(b) Different numbers of users

Figure 9: Scalability.



(a) Different Γ



(b) Difference cell size

Figure 10: Ratio of upper bound and lower bound.

In addition, both Laplace and ExpMech can attain lower computation times compared to LR-Geo. This efficiency stems from their methodology of selecting obfuscated locations based on predefined probability distributions - the Laplacian and exponential distributions, respectively - bypassing the need for LP, which in turn reduces the computation overhead. However, a notable drawback of these two methods is their inability to accurately estimate the cost caused by geo-obfuscation. This oversight results in an increased cost associated with geo-obfuscation, as the chosen obfuscated locations may lead to high traveling distances to the designated locations.

5.2.2 *Scalability.* Table 1 illustrates that the computation time for all algorithms escalates as the size of the location set K increases. Notably, **even when K reaches 400, the average computation time for LR-Leo remains at a comparatively low figure, approximately 0.8–1.8 seconds**.

We expanded our examination of K across a wider spectrum, from 100 to 1,600, and charted the computation times of LR-Geo in Fig. 9(a). This figure reveals that the computation time for LR-Geo is maintained at the same level with an increase in K, reaching up to 1.8 seconds. Moreover, Fig. 9(b) presents the computation times for LR-Geo as the number of users varies from 2 to 10. As expected, there is a noticeable rise in computation time corresponding to an increase in the number of users. This trend is attributed to the framework of Benders' decomposition (introduced in Section 4.4), where the server is tasked with generating a subproblem for each user. The increase in the number of subproblems heightens the probability of encountering at least one subproblem that fails to achieve optimal convergence swiftly, thereby prolonging the convergence time.

5.3 Cost Measurement

5.3.1 *Comparison with the benchmarks.* Table 2 compares the expected costs incurred by various algorithms for $K = 100, 200, 300, 400$. It is observed that LR-Geo significantly reduces the expected cost compared to Laplace and ExpMech. Specifically, LR-Geo's expected cost is, on average, 54.70% and 46.64% lower than that of Laplace and ExpMech, respectively. This efficiency is attributed to Laplace and ExpMech's reliance on Laplace/Exponential distributions for selecting obfuscated locations, which fails to accurately reflect the mobility constraints of vehicles within the road network, thereby elevating the cost. Furthermore, LR-Geo's cost performance is nearly on par with ConstOPT's for $K = 100, 200, 300$, yet it surpasses LP in cost at $K = 100, 200, 300$. Although LP is designed to achieve the global minimum cost by evaluating all potential locations within the target area, this approach is negated by its extensive computational requirements. As indicated in Table 1, LP struggles to compute obfuscation matrices within the 1800-second limit, highlighting a critical trade-off between cost efficiency and computational feasibility. Finally, the cost of LR-Geo-F is 5.56% higher than that of LR-Geo because LR-Geo-F imposes Geo-Ind constraints on all pairs of LR locations, resulting in a smaller feasible region for the obfuscation matrix and, consequently, higher utility loss.

5.3.2 *Comparison with the theoretical bounds.* To assess how close LR-Geo can achieve the optimal, we calculate a lower bound for the expected cost by solving the relaxed version of LR-Geo in Equ. (31)–(32), with the findings presented in Table 1. Here, we introduce the *approximation ratio*, defined as the quotient of the expected cost derived from LR-Geo over the calculated lower bound. A smaller approximation ratio indicates a closer proximity of LR-Geo's solution to the optimal. The results in the table indicate that, on average, **the approximation ratio for the expected cost of LR-Geo stands at 1.24, 1.2, 1.13, and 1.23 for $K = 100, 200, 300, 400$, respectively**.

It's important to recognize that LR-Geo does not attain the optimal solution since it operates with a constrained set of locations (LR locations) rather than the entire location set. Furthermore, LR-Geo does not utilize exact cost coefficients; instead, it estimates these coefficients using a cost reference table. Thus, it is interesting to test how the LR-Geo's approximation ratio is impacted by

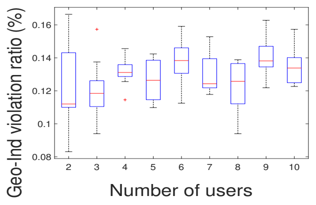


Figure 11: Geo-Ind violation ratio.

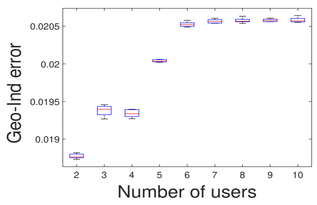


Figure 12: Geo-Ind violation error.

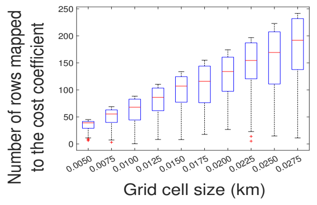


Figure 13: Threat using cost coefficient matrices.

(i) the selection of the LR locations, determined by the parameter Γ, i.e., the LR distance threshold, and
(ii) the accuracy of the cost coefficient estimation, determined by the **cell size** of the grid map of the cost reference table.

Fig. 10(a) shows the variation in the approximation ratio of LR-Geo as Γ increases from 10km to 50km. As defined in Equ. (11), Γ influences the size of the LR location set $\mathcal{N}_m$, with a higher Γ resulting in a larger $\mathcal{N}_m$. The figure indicates that the approximation ratio experiences a more pronounced decrease (averaging 34.72%) as Γ is increased from 10km to 20km. However, the decrease becomes marginal (only 0.79%) when Γ is further expanded from 20km to 50km. This observation suggests that enhancing Γ contributes to the optimality of the obfuscation matrices, yet beyond a certain threshold (20km in this instance), additional increases in Γ yield negligible improvements.

Fig. 10(b) shows the approximation ratio of LR-Geo as the cell size increases from 0.05km to 0.275km. As expected, the approximation ratio escalates with the increase in cell size, indicating that finer granularity in the location's representation within the cost reference table allows LR-Geo to more closely approximate the optimal solution. Specifically, the approximation ratio remains relatively stable and low for cell sizes up to 0.20km. Beyond this point, particularly when the cell size surpasses 0.20km, the ratio sees a marked increase. This trend underscores the importance of maintaining a cell size at or below 0.20km to optimize cost efficiency.

5.4 Privacy Measure

In LR-Geo, the computation of obfuscation matrices for each user is performed independently. While the obfuscation probabilities that conform to the constraints of the exponential distribution (in Equ. (19)) meet the Geo-Ind privacy criterion, as substantiated by **Theorem 4.4**, the remaining obfuscation probabilities do not guarantee Geo-Ind privacy. In this part, we examine the GV ratio as defined in Equ. (34). Fig. 11 shows the GV ratios for varying numbers of users. The figure reveals that the GV ratio remains exceptionally low, with a maximum of only 0.13%, demonstrating that, in practice, the Geo-Ind constraints are exceedingly likely to be met across the various obfuscation matrices tailored for different users.

Another question is the extent to which obfuscation probabilities violate the Geo-Ind constraint during these violations. If a pair of obfuscation probabilities, $z_{i,k}$ and $z_{j,k}$, fails to satisfy the Geo-Ind constraint—i.e., $z_{i,k} > e^{\epsilon d_{v_i,v_j}} z_{j,k}$—we define their 'Geo-Ind error (GVE)' as:

$$\mathrm{GVE}(z_{i,k}, z_{j,k}) = z_{i,k} - e^{\epsilon d_{v_i,v_j}} z_{j,k}. \quad (35)$$

A smaller $\mathrm{GVE}(z_{i,k}, z_{j,k})$ indicates that the Geo-Ind constraint between $z_{i,k}$ and $z_{j,k}$ is violated to a lesser extent.

Using this definition, we measured the Geo-Ind error for all Geo-Ind violations as the number of users increased from 2 to 10. Figure 12 illustrates the distribution of these errors. The results show that the Geo-Ind error reaches up to 0.021 across all experiments, and this error tends to increase as the number of users grows. This is because Geo-Ind violations occur between users, and as the user count rises, the number of potential Geo-Ind violations increases, resulting in a greater likelihood of larger Geo-Ind errors.

Finally, we investigate the potential risk associated with the upload of cost matrices, a concern discussed in **Section 4.7**. We simulate a scenario where a user uploads 100 cost matrices. We analyze, for each cost coefficient, the number of rows in the cost reference table that can be mapped to that coefficient. Intuitively, a greater number of rows mapped to a specific uploaded coefficient suggests a broader range of potential real and obfuscated location pairs, thereby diminishing the risk of LR location set disclosure (noting that the real location is within the LR location set). Fig. 13 displays the number of rows mapped to the uploaded cost coefficients for various grid cell sizes. As anticipated, the quantity of rows corresponding to a given coefficient increases with the increase of the cell size, indicating an increase in ambiguity and a reduced risk of location inference. The figure also underscores the difficulty of deducing the real location from the uploaded cost coefficient, as, on average, each coefficient is matched by 102.98 rows, providing a significant degree of location privacy.

6 Related Works

The study of location privacy began nearly two decades ago with Gruteser and Grunwald's pioneering work [33], where they introduced the concept of *location k-anonymity*. This idea has since evolved to include *l-diversity*, which ensures a user's location is indistinguishable from $l-1$ other locations [1]. However, the *l*-diversity model simplifies the threat landscape by assuming all alternative locations are equally probable as the user's actual location from an attacker's perspective. This assumption renders it susceptible to a range of sophisticated inference attacks [1, 3, 11].

In recent years, Andrés et al. [3] introduced a more applicable privacy criterion, *Geo-Ind*, grounded in the established concept of *differential privacy (DP)*. Following this work, a large body of location obfuscation strategies have been proposed, e.g., [1, 4–14]. Andrés et al., in their seminal work, not only proposed the Geo-Ind concept but also devised a method for achieving it by perturbing the actual location using a polar Laplacian distribution. Furthermore, as geo-obfuscation naturally introduces errors in the reported locations, thereby impacting the quality of LBS, a critical challenge addressed by several studies involves balancing the trade-off between service quality and privacy. For instance, within the constraints of Geo-Ind, Bordenabe et al. [9] first developed an optimization framework for geo-obfuscation aimed at minimizing individual user costs. Chatzikokolakis et al. [30] introduced "privacy mass"

Table 3: Comparison of major related works ("PN" means "privacy notion". "γ" means "whether γ is considered", "EIE" means "expected inference error", In the column "Location set size", "—" means "There is no limit to the size").

	Features						
Obfuscation methods	Lap.	Exp.	LP	γ	K	Privacy notion	Utility loss definition
CCS 2012 [18]			✓		30	EIE	Expected & max distance between real and reported locations
CCS 2013 [3]	✓			✓	—	Geo-Ind	Expected distance between real and reported locations
CCS 2014 [9]			✓		50	Geo-Ind	Expected distance between real and reported locations
PETS 2015 (Privacy game) [29]			✓		300	EIE & DP	Expected utility cost (depends on applications)
PETS 2015 (Elastic metric) [30]		✓			—	Geo-Ind (generalized)	Expected distance between real and reported locations
ICDM 2016 [14]			✓		57	DP	Expected residual standard error
WWW 2017 [6]			✓		16	Geo-Ind	Expected travel distance
NDSS 2017 [1]			✓		50	EIE & Geo-Ind	Expected distance between real and reported locations
CCS 2017 [31]			✓		25	EIE & Conditional entropy	Average & worst-case quality loss (depends on applications)
PETS 2017 [32]	✓		✓		—	Geo-Ind	Expected quality loss (depends on applications)
TMC 2020 [11]			✓		300–400	Geo-Ind	Expected difference between real and estimated travel distance
PETS 2020 [4]	✓				—	Geo-Ind	Expected distance between real and reported locations
SIGSPATIAL 2022 [12]					100	Geo-Ind	Expected difference between real and estimated travel distance
UAI 2022 [17]		✓	✓		400	Metric DP	Expected utility cost (depends on applications)
EDBT 2023 [19]			✓		70	Geo-Ind	Expected distance between real and reported locations
EDBT 2024 [20]			✓		300–400	Geo-Ind	Expected distance between real and reported locations
Our work		✓	✓	✓	**1,600**	Geo-Ind	Expected distance between real and reported locations

for points of interest, determining the Geo-Ind privacy budget ϵ for a location based on the local characteristics of each area. Wang et al. [6] addressed the collective cost incurred by users, proposing a privacy-preserving target assignment algorithm to reduce the total travel expense. Chatzikokolakis et al. [32] introduced a Bayesian remapping procedure to enhance the utility of geo-obfuscation, which can be applied to both infinite and finite location domains.

The majority of existing works in geo-obfuscation employ an LP framework, which generally necessitates $O(|\mathcal{V}|^2)$ decision variables and $O(|\mathcal{V}||\mathcal{E}|)$ linear constraints [9, 34], making the LP approach computationally intensive and challenging to implement on a large-scale LBS. Table 3 compares the related geo-obfuscation methods in different categories, including Laplacian noise ("Lap."), the exponential mechanism ("Exp."), and LP-based methods ("LP"). As the table indicates, the computational complexity of LP restricts most geo-obfuscation studies to handling at most 100 discrete locations. However, recent advancements [10, 11] have expanded the capability of processing secret datasets to approximately 300 records by leveraging Dantzig-Wolfe decomposition and column generation techniques. These studies primarily target LP models with Geo-Ind constraints applied across all pairs of secret records, facilitating the initialization process for column generation but are less applicable to broader geo-obfuscation challenges that only necessitate constraints for adjacent locations. Other innovative approaches, such as [17], combine LP with the exponential mechanism to improve scalability, though this may lead to compromises in solution optimality. Given the time-sensitive natures of many LBS applications, existing geo-obfuscation methodologies are constrained to either low spatial resolution over large areas (for instance, [1] focuses on city-scale regions, discretizing the location field into a grid where each cell measures 766m by 766m), or to high resolution within smaller areas (as in [11], which examines a small town with location points sampled every 500 square meters).

Compared to those existing works, LR-Geo introduced in this paper substantially lowers computational costs while maintaining a degree of optimality. This advancement facilitates the application of geo-obfuscation in large-scale LBS applications, enabling more accurate representations of locations.

7 Conclusions

We proposed to reduce the computation cost of the geo-obfuscation calculation by shrinking its range to a set of more relevant locations. Considering that the reduced geo-obfuscation range can possibly disclose the user's real location, we designed a remote computing strategy to migrate the geo-obfuscation calculation to the server without disclosing the location set covered by geo-obfuscation. The experimental results have demonstrated the superiority of our method in terms of privacy, service quality, and time efficiency, with the comparison of the selected benchmarks.

We envision several promising directions to continue this research. Firstly, this paper focuses on locations that are evenly distributed. In the next phase, we will consider cases where the locations are not necessarily evenly distributed. Since varying location densities lead to different distance matrices, which may inadvertently reveal information about a user's actual location, we will conduct a formal analysis of the potential information leakage caused by these distance matrices. Secondly, our current work considers a homogeneous mobility model, where a single cost reference table graph is sufficient to describe users' traveling costs. In reality, the users might be heterogeneous, e.g., a mixture of pedestrians and vehicles, and even a single user's mobility can possibly switch between different models. Then, how to model the mobility features of heterogeneous users using multiple cost reference table graphs is another problem to address. Finally, leveraging *reinforcement learning (RL)* could accelerate BD convergence by treating cut selection in Stage 2 as a parameterized stochastic policy. A trained RL model can identify an optimal sequence of cuts, eliminating the need for re-training with each new problem instance.

Acknowledgments

This research is partially supported by U.S. NSF grants CNS2136948 and CNS-2313866. The authors used ChatGPT 4o to revise the text in Sections 1–6 to correct typos, grammatical errors, and awkward phrasing. Work of Anna Squicciarini was supported by (while serving at) the National Science Foundation. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

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

[2] D.E. Bakken, R. Rarameswaran, D.M. Blough, A.A. Franz, and T.J. Palmer. Data obfuscation: anonymity and desensitization of usable data sets. *IEEE Security & Privacy*, 2(6):34–41, 2004.

[3] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *Proc. of ACM CCS*, pages 901–914, 2013.

[4] Ricardo Mendes, Mariana Cunha, and Joao Vilela. Impact of frequency of location reports on the privacy level of geo-indistinguishability. *PoPETS*, 2020:379–396, 04 2020.

[5] Simon Oya, Carmela Troncoso, and Fernando Pérez-González. Rethinking location privacy for unknown mobility behaviors. In *2019 IEEE EuroS&P*, pages 416–431, 2019.

[6] 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 ACM WWW*, pages 627–636, 2017.

[7] Reza Shokri, George Theodorakopoulos, and Carmela Troncoso. Privacy games along location traces: A game-theoretic framework for optimizing location privacy. *ACM Trans. Priv. Secur.*, 19(4), dec 2016.

[8] Yonghui Xiao and Li Xiong. Protecting locations with differential privacy under temporal correlations. In *Proc. of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, oct 2015.

[9] N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Optimal geo-indistinguishable mechanisms for location privacy. In *Proc. of ACM CCS*, pages 251–262, 2014.

[10] C. Qiu, A. C. Squicciarini, Z. Li, C. Pang, and L. Yan. Time-efficient geo-obfuscation to protect worker location privacy over road networks in spatial crowdsourcing. In *Proc. of ACM CIKM*, 2020.

[11] C. Qiu, A. C. Squicciarini, C. Pang, N. Wang, and B. Wu. Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability. *IEEE TMC*, pages 1–1, 2020.

[12] C. Qiu, L. Yan, A. Squicciarini, J. Zhao, C. Xu, and P. Pappachan. Trafficadaptor: An adaptive obfuscation strategy for vehicle location privacy against vehicle traffic flow aware attacks. In *Proc. of ACM SIGSPATIAL*, 2022.

[13] Raed Al-Dhubhani and Jonathan M. Cazalas. An adaptive geo-indistinguishability mechanism for continuous lbs queries. *Wirel. Netw.*, 24(8):3221–3239, nov 2018.

[14] Leye Wang, Daqing Zhang, Dingqi Yang, Brian Y. Lim, and Xiaojuan Ma. Differential location privacy for sparse mobile crowdsensing. In *2016 IEEE ICDM*, pages 1257–1262, 2016.

[15] Frederick S. Hillier. *Linear and Nonlinear Programming*. Stanford University, 2008.

[16] Lorenzo Bracciale, Marco Bonola, Pierpaolo Loreti, Giuseppe Bianchi, Raul Amici, and Antonello Rabuffi. CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from https://crawdad.org/roma/taxi/20140717, July 2014.

[17] Jacob Imola, Shiva Kasiviswanathan, Stephen White, Abhinav Aggarwal, and Nathanael Teissier. Balancing utility and scalability in metric differential privacy. In *Proc. of UAI 2022*, 2022.

[18] R. Shokri, G. Theodorakopoulos, C. Troncoso, J. Hubaux, and J. L. Boudec. Protecting location privacy: Optimal strategy against localization attacks. In *Proc. of ACM CCS*, pages 617–627, 2012.

[19] P. Pappachan, C. Qiu, A. Squicciarini, and V. Manjunath. User customizable and robust geo-indistinguishability for location privacy. In *Proc. of International Conference on Extending Database Technology (EDBT)*, 2023.

[20] Chenxi Qiu, Sourabh Yadav, Yuede Ji, Anna Squicciarini, Ramanamurthy Dantu, Juanjuan Zhao, and Chengzhong Xu. Fine-grained geo-obfuscation to protect workers' location privacy in time-sensitive spatial crowdsourcing. In *Proceedings of 27th International Conference on Extending Database Technology (EDBT)*, 2024.

[21] Yelp. https://www.yelp.com/, 2020. Accessed: 2020-04-07.

[22] Waze. https://www.waze.com/, 2019. Accessed: 2019-07-22.

[23] openstreetmap. https://www.openstreetmap.org/, 2020. Accessed: 2020-04-07.

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

[25] Ragheb Rahmaniani, Teodor Gabriel Crainic, Michel Gendreau, and Walter Rei. The benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817, 2017.

[26] Michael B. Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 938–942, New York, NY, USA, 2019. Association for Computing Machinery.

[27] Uber. https://www.uber.com/, 2022. Accessed in October 2022.

[28] MATLAB. https://www.mathworks.com/products/matlab.html, 2019. Accessed: 2019-07-22.

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

[30] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. Constructing elastic distinguishability metrics for location privacy. *PoPETs*, 2015:156–170, 2015.

[31] Simon Oya, Carmela Troncoso, and Fernando Pérez-González. Back to the drawing board: Revisiting the design of optimal location privacy-preserving mechanisms. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 1959–1972, New York, NY, USA, 2017. Association for Computing Machinery.

[32] Konstantinos Chatzikokolakis, Ehab Elsalamouny, and Catuscia Palamidessi. Efficient utility improvement for location privacy. *Proceedings on Privacy Enhancing Technologies*, 2017.

[33] M. Gruteser and D. Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of ACM MobiSys*, 2003.

[34] Chenxi Qiu. Enhancing scalability of metric differential privacy via secret dataset partitioning and benders decomposition. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 1944–1952, 8 2024. Main Track.

Appendix

A Math Notations

Table 4: Main notations and their descriptions

Symbol	Description
$\mathcal{V}$	The discrete location set $\mathcal{V} = \{v_1, ..., v_k\}$
$\mathcal{G}$	The Geo-Ind graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ are the location set and the edge set of $\mathcal{G}$
d_{v_i,v_j}	The Haversine distance between v_i and v_j
$\mathbf{Z}$	The obfuscation matrix $\mathbf{Z}$
$z_{i,k}$	The probability of selecting v_k as the obfuscated location given the real location v_i
$\mathbf{z}_i$	The obfuscation vector of v_i, i.e., $\mathbf{z}_i = [z_{i,1}, ..., z_{i,K}]$
$C\,(v, r)$	The circle centered at v with radius r
$tc\,(v, u)$	The travel cost from location v to location u
c_{v_i,v_k}	The cost coefficient of $z_{i,k}$ in OMG
$\mathcal{N}_m$	The LR location set of v_i
$\mathcal{O}_m$	The obfuscated location set of $\mathcal{N}_m$
$\hat{\mathcal{V}}$	The discrete location set covered by cost reference table
$\tilde{\mathcal{G}}$	The cost reference table graph $\tilde{\mathcal{G}} = \left(\hat{\mathcal{V}}, \hat{\mathcal{E}}\right)$, where $\hat{\mathcal{V}}$ and $\hat{\mathcal{E}}$ are the location set and the edge set of $\tilde{\mathcal{G}}$
$\beta_{i,k}$	The expectation of the cost estimation error taken over all possible destination locations
ϵ	The privacy budget of Geo-Ind
γ	The neighbor threshold
Γ	The LR distance threshold

A.1 Detailed Notations in Benders' Decomposition

- The coefficient matrices $\left[\mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m}, \mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m}\right]$ includes the Geo-Ind constraints between the obfuscation vectors of the locations in $\mathcal{N}_m$:

$$\left[\mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m}, \mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m}\right] = \begin{bmatrix} \ddots & \cdots & \cdots & \cdots & \iddots \\ \cdots & 1 & \cdots & -e^{\epsilon d_{v_i,v_j}} & \cdots \\ \cdots & -e^{\epsilon d_{v_i,v_j}} & \cdots & 1 & \cdots \\ \iddots & \cdots & \cdots & \cdots & \ddots \end{bmatrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \begin{matrix} \forall v_i, v_j \in \mathcal{N}_m \\ \text{s.t. } d_{v_i,v_j} \le \gamma \end{matrix}$$

- $\left[\mathbf{A}^{\text{unit}}_{\mathcal{N}_m}, \mathbf{B}^{\text{unit}}_{\mathcal{N}_m}\right]$ includes $|\mathcal{N}_m|$ rows, where each row corresponds to the unit measure constraint of the obfuscation vector $\mathbf{z}_i$ of location $v_i \in \mathcal{N}_m$.
- $\mathbf{b}^{\text{GeoI}}_{\mathcal{N}_m}$ is an all-zeros vector, which corresponds to the right-hand side coefficients of the constraint matrix $\left[\mathbf{A}^{\text{GeoI}}_{\mathcal{N}_m}, \mathbf{B}^{\text{GeoI}}_{\mathcal{N}_m}\right]$ in the LP formulation.
- $\mathbf{b}^{\text{unit}}_{\mathcal{N}_m}$ is an all-ones vector, which corresponds to the right-hand side coefficients of the constraint matrix $\left[\mathbf{A}^{\text{unit}}_{\mathcal{N}_m}, \mathbf{B}^{\text{unit}}_{\mathcal{N}_m}\right]$ in the LP formulation.

B Omitted Proofs

B.1 Proof of Theorem 3.1

Proof. We let $\{v_i, v_{l_1}, v_{l_2}, ..., v_{l_{n-1}}, v_{l_n}, v_j\}$ represent the sequence of locations in the shortest path between v_i and v_j. Therefore,

$$D_{v_i,v_j} = d_{v_i,v_{l_1}} + \sum_{m=1}^{n-1} d_{v_{l_m},v_{l_{m+1}}} + d_{v_{l_n},v_j}.$$

Since each pair of adjacent locations is geo-indistinguishable, for each $v_k \in \mathcal{V}$, we have

$$\frac{z_{i,k}}{z_{l_1,k}} \le e^{\epsilon d_{v_i,v_{l_1}}}, \tag{36}$$

$$\frac{z_{l_m,k}}{z_{l_{m+1}},k} \le e^{\epsilon d_{v_{l_m},v_{l_{m+1}}}} \;\; (m = 1, ..., n-1), \tag{37}$$

$$\frac{z_{l_n,k}}{z_{j,k}} \le e^{\epsilon d_{v_{l_n},v_j}}, \tag{38}$$

from which we can derive that

$$\begin{aligned} \frac{z_{i,k}}{z_{j,k}} &= \frac{z_{i,k}}{z_{l_1,k}} \prod_{m=1}^{n-1} \frac{z_{l_m,k}}{z_{l_{m+1}},k} \frac{z_{l_n,k}}{z_{j,k}} && (39) \\ &\le e^{\epsilon d_{v_i,v_{l_1}}} \prod_{m=1}^{n-1} e^{\epsilon d_{v_{l_m},v_{l_{m+1}}}} e^{\epsilon d_{v_{l_n},v_j}} && (40) \\ &= e^{\epsilon\left(d_{v_i,v_{l_1}} + \sum_{m=1}^{n-1} d_{v_{l_m},v_{l_{m+1}}} + d_{v_{l_n},v_j}\right)} && (41) \\ &= e^{\epsilon D_{v_i,v_j}}. && (42) \end{aligned}$$

The proof is completed. □

B.2 Proof of Proposition 4.3

Proof. First, since the Haversine distance between v_m and v_j should be no larger than their path distance in the Geo-Ind graph, i.e.,

$$d_{v_m,v_j} \le D_{v_m,v_j}. \tag{43}$$

According to the definition of LR location set in Equ. (11), $\forall v_j \in \mathcal{N}_m$

$$D_{v_m,v_j} \le \Gamma. \tag{44}$$

Based on Equ. (43) and Equ. (44), we obtain that

$$d_{v_m,v_j} \le \Gamma, \forall v_j \in \mathcal{N}_m. \tag{45}$$

According to Equ. (12), we have

$$d_{v_m,v_j} \le r_{\text{obf}}, \forall v_j \in \mathcal{O}_m. \tag{46}$$

According to Equ. (45) and Equ. (46), we have

$$d_{v_m,v_j} \le \max\{\Gamma, r_{\text{obf}}\},\; \forall v_j \in \mathcal{N}_m \cup \mathcal{O}_m. \tag{47}$$

$d_{v_m,v_a} \le \Gamma$ because v_a is selected within the LR location set $\mathcal{N}_m$. Then, according to the triangle inequality,

$$\begin{aligned} d_{v_a,v_j} &\le d_{v_m,v_a} + d_{v_m,v_j} && (48) \\ &\le \Gamma + \max\{\Gamma, r_{\text{obf}}\} && (49) \\ &= \max\{2\Gamma, \Gamma + r_{\text{obf}}\}, && (50) \end{aligned}$$

for each $v_j \in \mathcal{N}_m \cup \mathcal{O}_m$, indicating that $C(v_a, \max\{2\Gamma, \Gamma + r_{\text{obf}}\})$ covers both $\mathcal{N}_m$ and $\mathcal{O}_m$. □

B.3 Proof of Theorem 4.4

Proof. We prove it by considering the following three cases:
Case 1: v_k is within the obfuscation range of both v_i and v_j, i.e., $v_k \in O_i \cap O_j$. Then, $z_{i,k}^{(n)}$ and $z_{j,k}^{(m)}$ satisfy the constraint Equ. (22):

$$z_{i,k}^{(n)} = y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}}, \tag{51}$$

$$z_{j,k}^{(m)} = y_k e^{-\epsilon\frac{d_{v_j,v_k}}{2}},\ \forall v_k \tag{52}$$

implying that

$$\begin{aligned} & z_{i,k}^{(n)} - z_{j,k}^{(m)} e^{\epsilon d_{v_i,v_j}} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon d_{v_j,v_k}}{2}} e^{\epsilon d_{v_i,v_j}} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon (d_{v_j,v_k} - d_{v_i,v_j})}{2}} e^{\frac{\epsilon d_{v_i,v_j}}{2}} \\ \leq \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} e^{\frac{\epsilon d_{v_i,v_j}}{2}} \text{ (triangle inequality)} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} (1 - e^{\frac{\epsilon d_{v_i,v_j}}{2}}) \\ \leq \ & 0 \end{aligned} \tag{53}$$

Case 2: v_k is outside of the obfuscation range of either v_i or v_j. Without loss of generality, we consider the case $v_k \in O_i$ and $v_k \notin O_j$ (meaning $r_{\text{obf}} < d_{v_j,v_k}$), indicating that $z_{i,k}^{(n)} = y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}}$ and $z_{j,k}^{(m)} = y_k e^{-\frac{\epsilon r_{\text{obf}}}{2}}$. Therefore,

$$\begin{aligned} & z_{i,k}^{(n)} - z_{j,k}^{(m)} e^{\epsilon d_{v_i,v_j}} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon r_{\text{obf}}}{2}} e^{\epsilon d_{v_i,v_j}} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon (r_{\text{obf}} - d_{v_i,v_j})}{2}} e^{\frac{\epsilon d_{v_i,v_j}}{2}} \\ < \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon (d_{v_j,v_k} - d_{v_i,v_j})}{2}} e^{\frac{\epsilon d_{v_i,v_j}}{2}} \text{ (since } r_{\text{obf}} < d_{v_j,v_k}) \\ \leq \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} - y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} e^{\frac{\epsilon d_{v_i,v_j}}{2}} \text{ (triangle inequality)} \\ = \ & y_k e^{-\frac{\epsilon d_{v_i,v_k}}{2}} (1 - e^{\frac{\epsilon d_{v_i,v_j}}{2}}) \\ \leq \ & 0 \end{aligned} \tag{54}$$

Case 3: v_k is outside of the obfuscation range of both v_i and v_j, i.e., $v_k \notin O_i \cup O_j$. In this case, $z_{i,k}^{(n)} = z_{j,k}^{(m)} = y_k e^{-\frac{\epsilon r_{\text{obf}}}{2}}$, and it is trivial to prove that $z_{i,k}^{(n)} - e^{\epsilon d_{v_i,v_j}} z_{j,k}^{(m)} \leq 0$, since $e^{\epsilon d_{v_i,v_j}} \geq 1$.

The proof is completed. □

B.4 Proof of Proposition 4.5

Proof. (1) First, for each pair of LR locations v_i and v_j, their (ϵ, γ)-Geo-Ind constraints are

$$z_{i,k} - e^{\epsilon d_{v_i,v_j}} z_{j,k} \leq 0,\ \forall v_k \in \mathcal{V}, \tag{55}$$

$$z_{j,k} - e^{\epsilon d_{v_i,v_j}} z_{i,k} \leq 0,\ \forall v_k \in \mathcal{V}, \tag{56}$$

including $2|\mathcal{V}|$ linear constraints.

For any two Users n and m, there are totally $(|\mathcal{N}_n| + |\mathcal{N}_m|)$ LR locations, including $\binom{|\mathcal{N}_n| + |\mathcal{N}_m|}{2}$ location pairs. Hence, the total number of Geo-Ind constraints for User n and User m is

$$2|\mathcal{V}| \times \binom{|\mathcal{N}_n| + |\mathcal{N}_m|}{2} = (|\mathcal{N}_n| + |\mathcal{N}_m|)(|\mathcal{N}_n| + |\mathcal{N}_m| - 1)|\mathcal{V}|.$$

Note that within $\mathcal{N}_n$ (or $\mathcal{N}_m$), the Geo-Ind constraints are satisfied between any peer of relevant locations due to the linear constraints Eq. (). Therefore, the total number of Geo-Ind constraints satisfied within $\mathcal{N}_n$ and $\mathcal{N}_m$ is

$$\begin{aligned} & 2|\mathcal{V}| \times \binom{|\mathcal{N}_n|}{2} + 2|\mathcal{V}| \times \binom{|\mathcal{N}_m|}{2} & (57) \\ = \ & (|\mathcal{N}_n|^2 + |\mathcal{N}_m|^2 - |\mathcal{N}_n| - |\mathcal{N}_m|)|\mathcal{V}| & (58) \end{aligned}$$

Now, we consider the Geo-Ind constraints across $\mathcal{N}_n$ and $\mathcal{N}_m$. For each pair of locations $v_i \in \mathcal{N}_n$ and $v_j \in \mathcal{N}_m$, the set of obfuscated locations following the exponential distributions for both v_i and v_j is $\mathcal{A}_i \cap \mathcal{A}_j$. For any $v_k \in \mathcal{A}_i \cap \mathcal{A}_j$, the constraints in Equ. (55) are guaranteed (according to Theorem 4.4), including totally $2|\mathcal{A}_i \cap \mathcal{A}_j|$ linear constraints. The total number of constraints following exponential distributions for all pairs $(v_i, v_j) \in \mathcal{N}_n \times \mathcal{N}_m$ is $\sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} 2|\mathcal{A}_i \cap \mathcal{A}_j|$.

Therefore, we can conclude that for each pair User n and User m, the Geo-Ind constraint violation ratio is upper bounded by

$$1 - \frac{2\sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} |\mathcal{A}_i \cap \mathcal{A}_j| + (|\mathcal{N}_n|^2 + |\mathcal{N}_m|^2 - |\mathcal{N}_n| - |\mathcal{N}_m|)|\mathcal{V}|}{(|\mathcal{N}_n| + |\mathcal{N}_m|)(|\mathcal{N}_n| + |\mathcal{N}_m| - 1)|\mathcal{V}|}. \tag{59}$$

The proof is completed.

(2) The total number of Geo-Ind constraints for all the users is

$$2|\mathcal{V}| \times \binom{\sum_{n=1}^{M} |\mathcal{N}_n|}{2} = \sum_{n=1}^{M} |\mathcal{N}_n| \left(\sum_{n=1}^{M} |\mathcal{N}_n| - 1\right) |\mathcal{V}|.$$

We can obtain the total number of Geo-Ind constraints within each of $\mathcal{N}_1, ..., \mathcal{N}_M$ is

$$2|\mathcal{V}| \times \sum_{n=1}^{M} \binom{|\mathcal{N}_n|}{2} = |\mathcal{V}| \sum_{n=1}^{M} \left(|\mathcal{N}_n|^2 - |\mathcal{N}_n|\right)$$

We can also obtain that the total number of obfuscated locations achieve Geo-Ind constraints across $\mathcal{N}_1, ..., \mathcal{N}_M$ (since they follow exponential distribution) is

$$2\sum_{n=1}^{M} \sum_{m=n+1}^{M} \sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} |\mathcal{A}_i \cap \mathcal{A}_j|$$

Finally, we can conclude that Geo-Ind constraint violation ratio for all the users is upper bounded by

$$1 - \frac{2\sum_{n=1}^{M} \sum_{m=n+1}^{M} \sum_{(v_i,v_j)\in\mathcal{N}_n\times\mathcal{N}_m} |\mathcal{A}_i \cap \mathcal{A}_j| + \sum_{n=1}^{M} \left(|\mathcal{N}_n|^2 - |\mathcal{N}_n|\right) |\mathcal{V}|}{\sum_{n=1}^{M} |\mathcal{N}_n| \left(\sum_{n=1}^{M} |\mathcal{N}_n| - 1\right) |\mathcal{V}|}. \tag{60}$$

The proof is completed. □

B.5 Proof of Theorem 4.6

Proof. Before proving Theorem 4.6, we first introduce the following lemma:

Lemma B.1. *The actual cost c_{v_i,v_k} between location v_i and v_k is upper bounded by the estimated cost $\hat{c}_{v_i,v_k}$. The detailed proof of this lemma can be found in Section B.6.*

Let $\hat{\mathbf{Z}}_{\mathcal{N}_m} = \left\{\hat{z}_{i,k}^{(m)}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times O_m}$ denote the optimal solution of the CLR-Geo problem in Equ. (20)–(22) using the estimated cost matrix $\hat{\mathbf{C}}_{\mathcal{N}_m,\mathcal{O}_m}$ ($m = 1, ..., M$). Then, for each user m, the minimum expected cost calculated by the CLR-Geo problem is given by

$$\begin{aligned}
&\mathcal{L}\left(\hat{\mathbf{Z}}_{\mathcal{N}_m}\right) && (61)\\
=& \sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} \hat{c}_{v_i,v_k}\hat{z}^{(m)}_{i,k} && (62)\\
\geq& \sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} c_{v_i,v_k}\hat{z}^{(m)}_{i,k} \text{ (Lemma B.1)} && (63)\\
\geq& \underbrace{\sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} c_{v_i,v_k} z^{(m)*}_{i,k}}_{\text{the minimum expected cost}} && (64)
\end{aligned}$$

where $\mathbf{Z}^*_{\mathcal{N}_m} = \left\{z^{(m)*}_{i,k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ denote user m's optimal obfuscation matrix that achieves the minimum cost. The proof is completed. □

B.6 Proof of Lemma B.1

According to c_{v_i,v_k}'s definition (Equ. (5)),

$$\begin{aligned}
c_{v_i,v_k} =& p_i\sum_{j=1}^{Q} q_j\left|d_{v_i,v_j} - d_{v_k,v_j}\right|\\
=& p_i\sum_{v_j\in\mathcal{Q}'} q_j\left(d_{v_i,v_j} - d_{v_k,v_j}\right) + p_i\sum_{v_j\in\mathcal{Q}''} q_j\left(d_{v_k,v_j} - d_{v_i,v_j}\right)\\
\leq& p_i\sum_{v_j\in\mathcal{Q}'} q_j\left(\underbrace{\left(d_{\hat{v}_i,v_j} + d_{v_i,\hat{v}_i}\right)}_{\geq d_{v_i,v_j}\text{(triangle inequal.)}} - \underbrace{\left(d_{\hat{v}_k,v_j} - d_{v_k,\hat{v}_k}\right)}_{\leq d_{v_j,v_k}\text{(triangle inequal.)}}\right)\\
+& p_i\sum_{v_j\in\mathcal{Q}''} q_j\left(\underbrace{\left(d_{\hat{v}_k,v_j} + d_{v_k,\hat{v}_k}\right)}_{\geq d_{v_j,v_k}\text{(triangle inequal.)}} - \underbrace{\left(d_{\hat{v}_i,v_j} - d_{v_i,\hat{v}_i}\right)}_{\leq d_{v_i,v_j}\text{(triangle inequal.)}}\right)\\
=& p_i\sum_{j=1}^{Q} q_j\left|d_{\hat{v}_i,v_j} - d_{\hat{v}_k,v_j}\right| + p_i\sum_{j=1}^{Q} q_j\left(d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k}\right)\\
=& p_i\beta_{i,k} - p_i\left(d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k}\right)\\
=& \hat{c}_{v_i,v_k}. \qquad (65)
\end{aligned}$$

B.7 Proof of Theorem 4.7

Proof. Before proving Theorem 4.7, we first introduce the following lemma:

Lemma B.2. *The actual cost c_{v_i,v_k} between location v_i and v_k is lower bounded by the estimated cost $\tilde{c}_{v_i,v_k}$. The detailed proof of this lemma can be found in Section B.8.*

Let $\tilde{\mathbf{Z}}_{\mathcal{N}_m} = \left\{\tilde{z}^{(m)}_{i,k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ denote the optimal solution of the relaxed LR-Geo problem in Equ. (31)–(32) using the estimated cost matrix $\tilde{\mathbf{C}}_{\mathcal{N}_m,\mathcal{O}_m}$ ($m = 1, ..., M$). Then, for each user m, the minimum expected cost calculated by the relaxed LR-Geo problem is given by

$$\begin{aligned}
&\mathcal{L}\left(\tilde{\mathbf{Z}}_{\mathcal{N}_m}\right) && (66)\\
=& \sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} \tilde{c}_{v_i,v_k}\tilde{z}^{(m)}_{i,k} && (67)\\
\leq& \sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} \tilde{c}_{v_i,v_k} z^{(m)*}_{i,k} \text{ (as } \tilde{\mathbf{Z}}_{\mathcal{N}_m} \text{ is a relaxed solution of } \mathbf{Z}^*_{\mathcal{N}_m}) &&\\
\leq& \underbrace{\sum_{v_i\in\mathcal{N}_m}\sum_{v_k\in\mathcal{O}_m} c_{v_i,v_k} z^{(m)*}_{i,k}}_{\text{the minimum expected cost}} \quad \text{(Lemma B.1)} && (68)
\end{aligned}$$

where $\mathbf{Z}^*_{\mathcal{N}_m} = \left\{z^{(m)*}_{i,k}\right\}_{(v_i,v_k)\in\mathcal{N}_m\times\mathcal{O}_m}$ denote user m's optimal obfuscation matrix that achieves the minimum cost. The proof is completed.

B.8 Proof of Lemma B.2

According to c_{v_i,v_k}'s definition (Equ. (5)),

$$\begin{aligned}
c_{v_i,v_k} =& p_i\sum_{j=1}^{Q} q_j\left|d_{v_i,v_j} - d_{v_k,v_j}\right| \geq p_i\sum_{j=1}^{Q} q_j\left(d_{v_i,v_j} - d_{v_k,v_j}\right)\\
\geq& p_i\sum_{j=1}^{Q} q_j\left(\underbrace{\left(d_{\hat{v}_i,v_j} - d_{v_i,\hat{v}_i}\right)}_{\leq d_{v_i,v_j}\text{(triangle inequal.)}} - \underbrace{\left(d_{\hat{v}_k,v_j} + d_{v_k,\hat{v}_k}\right)}_{\geq d_{v_j,v_k}\text{(triangle inequal.)}}\right)\\
=& p_i\sum_{j=1}^{Q} q_j\left(d_{\hat{v}_i,v_j} - d_{\hat{v}_k,v_j}\right) - p_i\sum_{j=1}^{Q} q_j\left(d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k}\right)\\
=& p_i\beta_{i,k} - p_i\left(d_{v_i,\hat{v}_i} + d_{v_k,\hat{v}_k}\right)\\
=& \tilde{c}_{v_i,v_k}. \qquad (69)
\end{aligned}$$

□

C Detailed Description of Benders Decomposition

Benders' decomposition is composed of two stages,

- **Stage 1:** A **Master Program (MP)** to derive $\{y_1, ..., y_K\}$,
- **Stage 2:** A set of **subproblems** Sub_m ($m = 1, ..., M$), where each Sub_m aims to derive $\mathbf{z}'_{\mathcal{N}_m}$.

Stage 1: Master program. The MP derives $y_1, ..., y_K$ and replaces each cost $\mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m}$ by a single decision variable w_m, i.e., $w_m = \mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m}$. The MP is formulated as the following LP problem

$$\begin{aligned}
\min \quad & \textstyle\sum_{k=1}^{K}\alpha_k y_k + \sum_{m=1}^{M} w_m && (70)\\
\text{s.t.} \quad & \mathcal{H}: \text{Cut set of } y_1, ..., y_K, w_1, ..., w_M && (71)\\
& y_k \geq 0,\ k = 1, ..., K. && (72)
\end{aligned}$$

where each *cut* in $\mathcal{H}$ is a *linear inequality* of the decision variables $y_1, ..., y_K, w_1, ..., w_M$. According to the central LR-Geo formulated in Equ. (20)–(22), each w_m is given by

$$w_m = \min\left\{\mathcal{L}'\left(\mathbf{Z}_{\mathcal{N}_m}\right) \middle| \text{Equ. (24) for } \mathcal{N}_m \text{ is satisfied}\right\}. \qquad (73)$$

Since the MP doesn't know the optimal values of $\mathbf{Z}_{\mathcal{N}_m}$, instead of using Equ. (73), it "guesses" the value of w_m based the *cut set* $\mathcal{H}$. In the subsequent **Stage 2**, each Sub_m verifies whether the "guessed" value of w_m is feasible and achieves the minimum data cost as defined in Equ. (73); if not, Sub_m proposes the addition of a new cut to be included in $\mathcal{H}$, thereby guiding the MP to refine w_m during the next iteration.

In the following, we use $\left\{\overline{y}_1, ..., \overline{y}_K, \overline{w}_1, ..., \overline{w}_M\right\}$ to represent the optimal solution of the MP.

Stage 2: Subproblems. After the MP derives its optimal solution $\left\{\overline{y}_1, ..., \overline{y}_K, \overline{w}_1, ..., \overline{w}_M\right\}$ in **Stage 1**, each Sub_m validates whether $\overline{w}_m$ has achieved the minimum data cost,

$$w_m = \min\left\{\mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m}\left|\mathbf{A}_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} \geq \mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})\right.\right\}. \quad (74)$$

of which the *dual problem* can be formulated as the following LP problem:

$$\max \quad \left(\mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})\right)^\top \mathbf{u}_{\mathcal{N}_m} \quad (75)$$

$$\text{s.t.} \quad \mathbf{A}^\top_{\mathcal{N}_m}\mathbf{u}_{\mathcal{N}_m} \leq \mathbf{c}'_{\mathcal{N}_m},\ \mathbf{u}_{\mathcal{N}_m} \geq \mathbf{0}. \quad (76)$$

There are three cases of the dual problem:

Case 1: The optimal objective value is **unbounded**: By *weak duality* [15], $\overline{\mathbf{y}}$ does not satisfy $\mathbf{A}_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} \geq \mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})$ for any $\mathbf{z}'_{\mathcal{N}_m} \geq \mathbf{0}$. Since the dual problem is unbounded, there exists an *extreme ray* $\tilde{\mathbf{u}}_{\mathcal{N}_m}$ subject to $\mathbf{A}^\top_{\mathcal{N}_m}\tilde{\mathbf{u}}_{\mathcal{N}_m} \leq \mathbf{0}$ and $\left(\mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})\right)^\top \tilde{\mathbf{u}}_{\mathcal{N}_m} > 0$. To ensure that $\tilde{\mathbf{u}}_{\mathcal{N}_m}$ won't be an extreme ray in the next iteration, Sub_m suggests a *new cut* h (*feasibility cut*) to the MP:

$$h: \left(\mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})\right)^\top \tilde{\mathbf{u}}_{\mathcal{N}_m} \leq \mathbf{0}.$$

Case 2: The optimal objective value is **bounded** with the solution $\overline{\mathbf{u}}_{\mathcal{N}_m}$: By *weak duality*, the optimal value of the dual problem is equal to the optimal value of w_l constrained on the choice of $\overline{\mathbf{y}}$. In this case, Sub_m checks whether $\overline{w}_m < \left(\mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})\right)^\top \mathbf{u}_{\mathcal{N}_m}$. If yes, then $\overline{w}_m < \min\left\{\mathbf{c}'_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m}\left|\mathbf{A}_{\mathcal{N}_m}\mathbf{z}'_{\mathcal{N}_m} \geq \mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\overline{\mathbf{y}})\right.\right\}$, meaning that $\overline{w}_m$ derived by the MP is lower than the minimum cost. Therefore, Sub_m suggests a *new cut*

$$h: \ w_m \geq \left(\mathbf{b}_{\mathcal{N}_m} - \mathbf{B}_{\mathcal{N}_m}\mathbf{z}''_{\mathcal{N}_m}(\mathbf{y})\right)^\top \overline{\mathbf{u}}_{\mathcal{N}_m}$$

to the MP to improve w_m in the next iteration.

Case 3: There is **no feasible solution**: By *weak duality*, the primal problem either has no feasible/unbounded solution. The algorithm terminates.

After adding the new cuts (from all the subproblems) to the cut set $\mathcal{H}$, the BD moves to the next iteration by recalculating the MP and obtaining updated $\left\{\overline{y}_1, ..., \overline{y}_K, \overline{w}_1, ..., \overline{w}_M\right\}$. As **Stage 1** and **Stage 2** are repeated over iterations, the MP collects more cuts from the subproblems, converging the solution $\left\{\overline{y}_1, ..., \overline{y}_K, \overline{w}_1, ..., \overline{w}_M\right\}$ to the optimal.