Preserving Location Privacy in Ride-Hailing Service

Youssef Khazbak, Jingyao Fan, Sencun Zhu, and Guohong Cao
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, Pennsylvania 16802
Email:{ymk111, jxf376, szhu, gcao}@cse.psu.edu

***Abstract*—Ride-hailing service has become part of our daily life due to its convenience and low cost. However, it also raises location privacy concerns for riders, because the service provider can observe the full mobility traces of riders while they hail rides. To address this problem, we first present a baseline privacy preserving solution. Although the baseline solution can provide personalized rider location privacy, we identify potential location inference attacks against it. To overcome these attacks, we propose an enhanced privacy preserving solution that exploits novel obfuscation techniques to enable matching ride requests to drivers without breaching riders' location privacy and with limited loss of matching accuracy. We use real dataset of taxi-cabs to show that our solution, compared to previous work, provides much better ride matching, i.e., ride matching closer to the optimal solution, while preserving personalized riders' location privacy.**

I. Introduction

Ride-hailing services such as Uber, Lyft and Gett, enable drivers to offer rides using their own vehicles, and enables riders to request and hail rides. Millions of people enjoy such service due to its convenience and low cost. However, the current ride-hailing service significantly threatens riders' location privacy. Since rider mobility, including the pick-up and drop-off location information, is tracked, a Service Provider (SP) can infer sensitive information about the riders such as where they live and work [1]. Many real world incidents have been reported about the misuse of such data. For example, in November 2014, Uber investigated one employee who was reported to have tracked riders [2]. If this location privacy issue is not addressed, many users may not be willing to use such service despite its popularity.

Location privacy has been well studied in the literature, and researchers have proposed many location obfuscation mechanisms, such as location perturbation [3], spatial cloaking [4] [5], dummy location generation [6] [7], etc. In location perturbation, noise is added to locations to generate obfuscated locations. In spatial cloaking, a user reduces the granularity of his location so that his location can be hidden inside a cloaked region. The dummy location generation technique generates $k-1$ properly selected dummy locations to hide the user's actual location. These techniques increase users' location privacy as it would be more difficult for an adversary to know the actual locations of the users. However, these techniques cannot be directly applied to the current ride-hailing systems without affecting the system usability.

Recently, researchers started to look into privacy issues in ride-hailing services. PrivateRide [8] is the first system that aims to enhance location privacy for riders. It relies on spatial cloaking algorithms to obfuscate locations of riders and drivers by replacing their actual locations with cloaked regions. Then, in order to match a rider to a driver, the SP measures the distances between the rider's pick-up location and the drivers' locations. Based on the calculated distances, the rider is assigned to the driver who is the closest to his cloaked region. Since the ride matching is based on the cloaked regions, instead of the actual locations, the chosen driver may not be the optimal one and has to drive extra distance for rider pickup, and the rider may wait extra time to get the ride.

In this paper, we present a location privacy preserving solution that efficiently matches riders and drivers while preserving riders' location privacy. We first propose a baseline solution that allows a rider to select the driver who is the closest to his pick-up location without revealing his location to the SP. However, with side information such as knowledge of ride matching and temporal cloaking techniques deployed in the baseline solution, the SP can launch location inference attacks. To overcome these attacks, we propose an enhanced solution that allows a rider to specify his privacy preference. In this solution, the ride matching algorithm selects a set of drivers that are as close to the rider's location as possible, and meanwhile located within an area that meets rider's privacy preference. Then, the rider selects a driver among the set of drivers. The pick-up and drop-off times are further obfuscated in a way to prevent the SP from using the temporal information to improve the inference of the rider's location.

In summary, our main contributions are:

1) We present a baseline privacy preserving solution that protects locations of riders in ride-hailing services, and show potential inference attacks against it.
2) To overcome the inference attacks against the baseline solution, we propose an enhanced privacy preserving solution which provides personalized riders' location privacy. It relies on novel obfuscation techniques that satisfy riders' privacy requirements without affecting the convenience of the service, and with limited loss in ride matching accuracy.
3) We analyze a real dataset that contains 60000 rides. The experiment results show that our enhanced solution outperforms PrivateRide by achieving much better matching accuracy with negligible computational overhead.

The rest of the paper is organized as follows. Section II

presents the preliminaries. We present the baseline privacy preserving solution in Section III, and the enhanced solution in Section IV. We present evaluation results in Section V, and present related work in Section VI. Section VII discusses some practical issues, and Section VIII concludes the paper.

II. Preliminaries

In this section, we introduce ride-hailing services, the security model, the design goals, and the basic concept of Voronoi diagram which is employed in our solutions.

A. Ride-Hailing Services

The ride-hailing service involves three parties: riders, drivers and a service provider (SP). Riders are typically smartphone users who need to hail a ride. Drivers are car owners who are willing to offer rides. The SP receives ride requests from riders and matches the requests with available drivers.

Ride matching is primarily based on the locations of the rider and drivers. It is initiated when a rider sends a ride request to the SP that includes his pick-up and optionally drop-off locations. Then the SP selects among available drivers the closest driver to the rider's pickup location. To do so, drivers have to continuously report their locations to the SP.

After matching a driver to a ride request, the SP allows the driver and rider to coordinate the ride by sending each party the information of the other, i.e., name, reputation and phone number. If both parties accept the ride, the SP continuously shares the driver's location with the rider. Once the driver picks up the rider, he notifies the SP with the start of the ride. During the ride, the driver continuously updates the SP with his actual location. At the end of the ride, the driver notifies the SP and becomes available again to offer a new ride.

B. Security Model

We assume the SP is an honest but curious adversary who has the incentive to track riders' precise locations. The SP can use collected location traces of the riders to profile and infer sensitive information about them, improve its own service, or to sell the collected data to other third parties (e.g., advertisement agencies). However, the SP has no incentive to attack the riders' and drivers' mobile devices by providing them with malicious application. Because such malicious behavior can be detected via reverse engineering the application, and hence harm its reputation and lead to business loss over its competitors.

In our model, drivers continuously update the SP with their actual locations, because they are workers who provide ride offers to riders and in return receive money for such a service. We also assume that drivers do not disclose riders' locations to the SP, because drivers are normally independent workers who do not have the incentive to collude with the SP.

C. Design Goals

Our design goals for a ride-hailing service are as follows:

1) **Preserving rider's personalized location privacy:** The goal is to provide ride-hailing service without disclosing the riders' actual locations to the SP. Moreover, the service has to satisfy the riders' needs for personalized location privacy throughout the operation of the service.
2) **Maximizing the accuracy of ride matching:** In ride matching, the optimal is achieved when a rider is matched with the closest driver, because it minimizes the riders' waiting time, and the drivers' driving distance. Hence, the objective is to select a driver that is as close as possible to the rider's pick-up location while preserving his personalized location privacy.

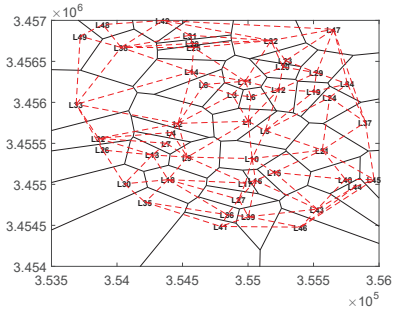

Fig. 1. Voronoi diagram of a set of drivers' locations. The dashed lines represent the dual graph for the Voronoi diagram of a set of drivers' locations.

D. Voronoi Diagram

The Voronoi diagram [9], [10] is a data structure in the field of computational geometry that represents the proximity information about a set of nodes. It partitions the space with a set of nodes into polygons such that each polygon contains exactly one node. Every point inside a given polygon is closer to the node in this polygon than to any other nodes. Figure 1 shows a Voronoi diagram and its dual graph represented by the dashed lines. The dual graph of a plane graph G has a vertex for each face of G and an edge for every two faces of G that are separated by an edge. Each vertex lies inside one Voronoi polygon, and each edge connects two vertices in two Voronoi polygons.

In this paper, we will show a potential location inference attack that exploits Voronoi diagram. Figure 1 shows the Voronoi diagram of a set of drivers' locations. Each driver location is enclosed by a Voronoi polygon. Locations inside one polygon are closer to the driver inside this polygon than any other drivers. Then based on the attack, we propose an enhanced privacy preserving ride matching algorithm that protects riders' location privacy by constructing a Voronoi diagram and its dual graph.

III. Baseline Privacy Preserving Ride-Hailing

In this section, we introduce the baseline privacy preserving ride-hailing solution, and describe location inference attacks against it.

A. System Overview

As shown in Figure 2, our system consists of three parties: riders, drivers, and SP. The SP handles the incoming ride

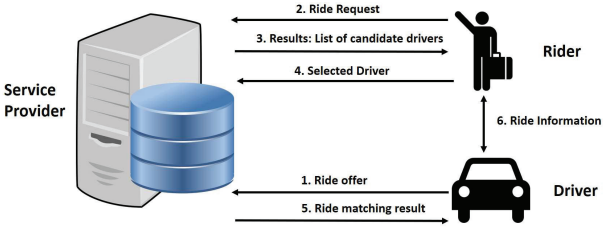

Fig. 2. System Model.

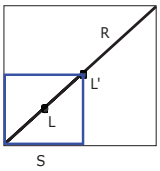

Fig. 3. Construction of geographical region Q.

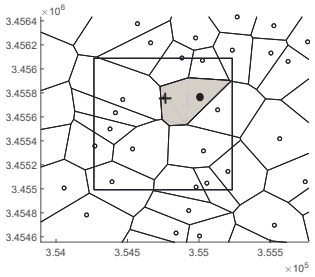

Fig. 4. Location Inference Attack with knowledge of ride matching.

requests from riders. A rider sends a ride request includes geographical region, and optionally obfuscated drop-off location. Then based on the provided geographical region, the SP constructs a list of drivers' locations for drivers that are close to the rider. The rider receives the list, picks a driver, and notifies the SP with the selected driver. The SP allows the driver and rider to coordinate the ride through direct communication. Through the established communication channel, they can share sensitive information such as actual pick-up location, drop-off location, and obfuscated locations. In addition, the driver continuously shares his actual location updates while heading to the pick-up location. On reaching the cloaked region where the rider is located, he updates the SP with the same cloaked region of the rider. Then the driver picks up the rider, and notifies the SP with an obfuscated pick-up time of the ride. During the ride, the driver continuously sends the SP the cloaked region until he moves out of it. Afterwards, the driver continuously sends his actual location to the SP. After the end of the ride, the driver notifies the SP with an obfuscated drop-off time of the ride and becomes available again to offer a new ride. Next, we describe the details of the baseline privacy preserving ride-hailing solution, which includes three components: ride initiation, ride matching, and temporal cloaking.

1) Ride Initiation: A rider initiates a ride request to the SP as follows. He sends a geographical region, denoted as Q, which contains his actual pick-up location and its cloaking region. Figure 3 shows the construction of Q. To construct Q, the rider specifies his privacy preference in the form of a cloaking region [11], where his actual location will be indistinguishable from any other location inside it. Specifically, the rider specifies the cloaking region as a square of an area equals to S^2 centered at the actual location L. Then another location, denotes L', is chosen uniformly at random from points inside the cloaking region. Finally, Q is generated centered at L' with a diagonal $2R$. To guarantee that the cloaking region will lie inside Q, R should be of length greater than or equal to the length of the diagonal of the cloaking region, i.e., $R \geq \sqrt{2}S$. Using the same construction method, the rider can generate an obfuscated drop-off location. Afterwards, the rider sends the ride request with the geographical area Q to the SP. Once the SP receives the ride request, it uses Q to filter out all drivers that are irrelevant to the rider's pick-up location and sends to the rider a list of drivers' locations in Q.

2) Ride Matching: Following the ride initiation process, the SP sends a set of candidate drivers' locations which are within the geographical region Q provided by the rider. Then the rider selects the closest driver to his actual location, and notifies the SP with the selected driver.

3) Temporal Cloaking: To obfuscate the pick-up and drop-off times, the SP discretizes the time of the day into time intervals. Then a driver reports to the SP the time interval where an event occurs. For example, if the time interval is 4 minutes, an event such as rider pick-up occurring between $0{:}00$ and $0{:}04$ would be reported to the SP at $0{:}04$.

B. Location Inference Attacks

In this subsection, we first show how the SP can launch a location inference attack with knowledge of ride matching. Then, we show another location inference attack which exploits knowledge of temporal cloaking along with information about driver's car speed to improve the inference of riders' locations.

1) Location inference Attack with knowledge of ride matching : Since a rider selects the driver closest to his location, the SP can improve the inference of the rider's location based on the selected driver. The SP has access to all drivers' locations, and thus the selected driver indicates the rider's location is closest to that driver's location than any other driver's location. Therefore, the SP can perform the attack by constructing a Voronoi diagram based on a set of drivers' locations. Every driver's location is enclosed by a Voronoi polygon which consists of all locations closer to that location than any other driver's location. By using the Voronoi diagram, the SP can infer the rider's location is within the Voronoi polygon of the selected driver, denoted by P_{D^*}. Therefore, when the cloaking region is larger than P_{D^*}, the SP can reduce rider's cloaking region into a smaller region represented by P_{D^*}.

Figure 4 describes the location inference attack. The '$\circ$' symbol represents the drivers locations, and the constructed Voronoi diagram partitions the region based on the distance to the drivers' locations. The square represents the cloaking region of size $A_R = S^2$. The gray polygon represents the Voronoi polygon P_D^* of the closest driver. The '$\bullet$' symbol represents the location of the driver closest to the rider's pick-up location. The pick-up location is represented by a '**+**' symbol. Considering the rider chooses the closest driver to his

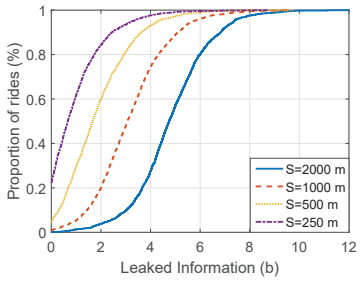

Fig. 5. *Leaked Information* for the location inference attack.

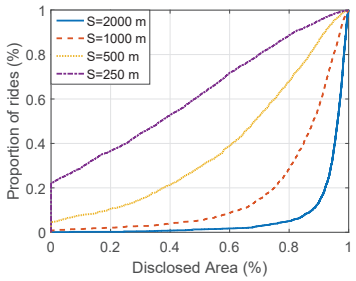

Fig. 6. *Disclosed Area* for the location inference attack.

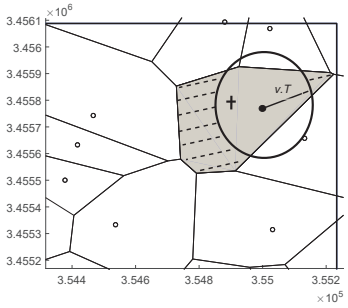

Fig. 7. Location inference attack with knowledge of temporal cloaking.

location, then the SP deduces that the rider's actual location lies within P_{D^*}.

To quantify the effect of the attack on location privacy, we introduce the following notations. Let O denote the cloaking region and divide it into a grid of K cells. Each location is represented by the cell it falls into. The probability of being in a cell i is denoted by p_i. Then, we first quantify the attack using a metric called *Leaked Information*, defined as follows:

$$\textit{Leaked Information} = H(O) - H(O|P_{D^*}), \tag{1}$$

where $H(O) = \sum_{i=1}^{R} p_i \ . \ \log \frac{1}{p_i}$.

$H(.)$ denotes the entropy of a random variable [12]. The $H(O)$ evaluates the SP's prior knowledge about rider's location and it is the entropy of the prior probability. We assume that before the attack, the SP does not have any side information and the best he can do is to reveal O. Hence the actual rider's location can be within any cell inside O with probability equal to $\frac{1}{K}$. Here, the prior probability is represented by a uniform distribution which provides the maximum entropy before running the attack, and hence $H(O) = \log K$. The $H(O|P_{D^*})$ measures the SP's posterior knowledge after running the attack and it is the entropy of the posterior probability. Hence, the privacy leakage is measured in terms of the change in SP's knowledge. Next, we quantify the attack using another metric called *Disclosed Area*, defined as follows.

$$\textit{Disclosed Area} = \frac{A_R - A_p}{A_R}. \tag{2}$$

The *Disclosed Area* metric measures the proportion of the area of the cloaking region disclosed after running the attack. A_R denotes the size of the cloaking region and A_p denotes the size of P_{D^*}.

Using these two metrics, we quantify the effect of the attack on a real dataset consisting of mobility traces of taxi cabs collected in Shanghai, China [13]. The dataset is described in more details in Section V.A. Figure 5 and Figure 6 show the cumulative distribution functions of the *Leaked Information* and *Disclosed Area*, respectively. It can be observed as S increases, both *Leaked Information* and *Disclosed Area* increase. This is because when A_R is larger, more drivers can be located inside O. With many drivers inside O, many constructed voronoi polygons lie inside O, and the SP can infer that the rider's location is only inside one of them which is P_{D^*}. As shown in Figure 6, even with small O, a square with $S = 250$ m, only around 20% cases do not disclose any information about O. For cloaking regions with larger A_R, the privacy breach is nearly inevitable. For example, when $S = 1000$ m, in 80% cases, the portion of the area disclosed is larger than 75%.

2) Location inference Attack with knowledge of temporal cloaking: The temporal cloaking technique does not prevent the SP from exploiting temporal information along with side information such as driver's car speed to infer more precise locations. The inference attack is described as follows. Let v denote the maximum driver speed, and T denote the time difference between the time of the last known driver's location and the obfuscated time for pick-up. Let the maximum distance between any two points inside the cloaking region P_{D^*} be d_{max}. Then the SP may prune part of O by computing the maximum distance a driver can travel when moving at speed v. The maximum travel distance equals to $v.T$. Then the attack is successful iff $d_{max} > 2vT$.

As shown in Figure 7, when d_{max} of P_{D^*}, i.e., the gray polygon, is larger than the maximum travel distance a driver can travel, it indicates that part of P_{D^*} is unreachable by the driver at the pick-up time, and hence the actual pick-up location cannot lie within the unreachable region, i.e., the region shown by dashed lines. Hence, the SP can further reduce the cloaking region P_{D^*}. Similar attack can be used to improve the inference of the drop-off location.

IV. Enhanced Privacy Preserving Ride-Hailing

Our enhanced privacy preserving ride-hailing is based on the baseline solution. By improving the techniques used in ride matching and temporal cloaking, our enhanced solution can defend against location inference attacks.

A. The Enhanced Ride Matching

The enhanced ride matching consists of two parts. The first part is a spatial cloaking algorithm that constructs a cloaking region based on drivers' locations and rider's privacy preference. The second part is a driver selection algorithm that picks a single driver among the set of drivers located

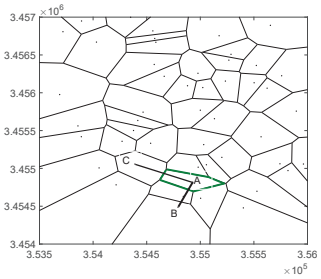

Fig. 8. The distance between a rider's location at node A and the location of its neighbor node C is larger than the distance between it and the location of its two-hop neighbor node B.

within the cloaking region. The driver is chosen according to a probabilistic mechanism.

In the spatial cloaking algorithm, the actual pick-up location of a rider, denoted as L_R^p, is hidden inside a cloaking region. The rider specifies his privacy preference in the form of an area of size A_R. The privacy preference corresponds to hiding the rider's actual location inside an area of size at least A_R. With such privacy preference, the cloaking algorithm generates the cloaking region as follows. First, let $L_D = \{L_{D_1}, .., L_{D_n}\}$ denote the set of n drivers' locations provided by the SP. The algorithm constructs the Voronoi diagram and its dual graph from L_D. The Voronoi diagram divides the area into a set of Voronoi polygons represented by $\{P_1, .., P_n\}$. The dual graph of the Voronoi diagram corresponds to constructing a Delaunay triangulation [14] on the set L_D. Each node L_{D_i} of the dual graph is labeled with two metrics: the Euclidean distance between L_{D_i} and the L_R^p, denoted as $D_{D_i} = dist(L_{D_i}, L_R^p)$, and the area of the Voronoi polygon P_i, denoted as A_{P_i}. The A_{P_i} is computed using the polygon triangulation method [15]. Then the problem can be formulated as follows.

Definition IV.1. Let the dual graph of the Voronoi diagram denote $G = (V, E)$, where $V = \{L_{D_1}, .., L_{D_n}\}$ and each location L_{D_i} lies inside a Voronoi polygon P_i. Find a subgraph $G_S = (V_S, E_S)$, $E_S \subset E$ that satisfies the following: 1) the drivers' locations in V_S are as close as possible to the rider's location L_R^p; that is, for any driver's location $L_{D_i} \in V_S$ and driver's location $L_{D_j} \notin V_S$, $D_{D_i} \leq D_{D_j}$, 2) the area covered by the Voronoi polygons of the drivers in V_S, denoted as $A_T = A_{P_1} + .. + A_{P_{|V_S|}}$, must be equal to at least A_R.

The problem can be solved using a greedy algorithm. The intuition of the algorithm is to construct a subgraph that includes a set of drivers' locations by sequentially picking a driver's location that is closest to the rider's location until the size of the area A_T covered by the selected driver's polygons exceeds the required threshold A_R. Hence, the algorithm progressively expands the covered area by aggregating drivers' polygons until the privacy preference $A_R \leq A_T$ is satisfied.

To construct the subgraph, the algorithm traverses the dual graph G. However, the traversal of G cannot be done using a straight forward Breadth-First search. Because it is not always the case that immediate neighbor nodes are closer than farther-away neighbors. For example, as shown in Figure 8, the distance between a rider's location at node A and the location of its neighbor node C is larger than the distance between it and the location of its two-hop neighbor node B. Based on this observation, the algorithm uses a priority queue Q_S in traversing G and the traversal works as follows. Initially, V_S contains the location of the driver closest to the rider, i.e., the root node. Next, the algorithm explores and adds the neighbor nodes of the root node to Q_S. Then it picks a node among the neighbors from Q_S that has the smallest distance, inserts it into V_S, and explores and adds its neighbors to Q_S. Hence at every step, the algorithm chooses, among all nodes in Q_S, a node that is a neighbor to any node in V_S without favoring root's neighbors over others. The choice from neighbors is to preserve the connectivity of the cloaking region constructed by the drivers' polygons of V_S.

The second part of the enhanced ride matching is the driver selection algorithm, which runs on a set S equivalent to V_S but ordered according to the distance from the root node. It divides S into two sets S_1 and S_2, such that S_1 contains locations that are closer to the rider's location when compared to locations in S_2. Then a single driver is selected among S_1 and S_2 with probability equal to $w * p_i$ and p_i, respectively. The parameter w denotes a weight factor which, when set to a value larger than one, allows the driver selection algorithm to favor closer drivers. The parameter p_i is equal to $\frac{1}{(w|S_1|+|S_2|}$).

Spatial cloaking and driver selection are summarized in Algorithm 1. For the priority queue implementation, we consider using a regular binary heap. Then the complexity time of the algorithm is as follows. Lines $1 \sim 2$ constructs the Voronoi diagram and its dual graph. An algorithm such as the Fortune's algorithm [16] can be used to generate the Voronoi diagram and its dual, which runs in $O(n \log n)$. Lines $3 \sim 6$ iterate on all nodes in V to label them, which takes $O(n)$. The traversal of the dual graph in lines $7 \sim 14$ runs in time $O(|V_S| + |E_S|)$, where $|V_S| = O(n)$ and $|E_S| = O(n)$ as stated in the following lemma:

Lemma IV.1. *Any planar graph on $n \geq 3$ vertices has at most $3n - 6$ edges and at most $2n - 4$ faces [17].*

Hence the running time of the graph traversal is $O(n)$ and an iteration of it runs in $O(\log n)$ because each iteration consists of: 1) finding and removing the node with the smallest distance among all nodes in Q_S, which requires $O(\log n)$, and 2) adding to Q_S the node's children, which requires $O(\log n)$. Therefore, Lines $7 \sim 14$ requires $O(n \log n)$ time. And lines $15 \sim 17$ has the time complexity of $O(n \log n)$ for sorting V_S using merge sort algorithm. Hence, the time complexity of the greedy algorithm is $O(n \log n)$.

Figure 9 shows the constructed cloaking region, the selected driver by our algorithm, and one of the drivers that can be selected by PrivateRide. As can be observed, our algorithm works as follows. V_S initially contains the location of the closest driver, and A_T equals to the area of the closest driver's polygon, represented by the light gray polygon. Then the algorithm explores and adds the neighbor nodes of the root

node, until it constructs the cloaking region that satisfies the privacy preference A_R. The cloaking region is represented by the light and dark gray regions. Then a single driver is selected among the set of drivers within this region. The driver selected by our algorithm is represented by a '$\odot$' symbol. The driver selected by PrivateRide is picked among the set of drivers inside the square region, i.e., the cloaking region represented by the square. Because PrivateRide considers all drivers co-located with the rider at the same cloaking region have a distance equals to zero from that rider's location, it cannot distinguish between them. The driver represented by '$\oplus$' symbol, which is at the boundary of the cloaking region, can be selected by PrivateRide. The closest driver is represented by a '$\bullet$' symbol. As observed, our algorithm does not select the closest driver due to the privacy preference. However, the selected driver by our algorithm can be at a distance closer to the rider's location compared to the selected driver by PrivateRide. This is because PrivateRide can select a driver located in the boundary of the square region, while our algorithm uses nearby drivers to generate the cloaking region.

Algorithm 1: ENHANCED RIDE MATCHING

Input: rider's pickup location L_R^p and set of n drivers' locations $L_D = \{L_{D_1}, .., L_{D_n}\}$
Output: an optimal driver location that satisfies the privacy requirement A_R

/* Spatial cloaking */
1 construct Voronoi diagram from set L_D
2 construct the dual graph $G = (V, E)$ of the Voronoi diagram where $V = L_D$ and each location L_{D_i} lies inside a Voronoi polygon P_i.
3 **for** *each node* L_{D_i} **do**
4 compute the Voronoi polygon area A_{P_i}
5 compute the distance $D_{D_i} = dist(L_{D_i}, L_R^p)$
6 label the node with $\{A_{P_i}, D_{D_i}\}$
7 set V_S to the node u with the smallest distance D_{D_i} among all nodes in V
8 set A_T to the area A_{P_i} of the Voronoi polygon of u
9 initialize priority queue Q_S and push in Q_S the children of u
10 **while** $A_T < A_R$ **do**
11 find and remove a node $v \in Q_S$ with the smallest distance among all nodes in Q_S
12 add to A_T the area A_{P_i} of the Voronoi polygon of v
13 add v to V_S
14 push in Q_S the children of v

/* Driver selection */
15 set S to V_S and sort it in ascending order by distance
16 divide S into two sets S_1 and S_2
17 pick a driver location L_D^* from S_1 with probability $w * p_i$ or from S_2 with probability p_i
18 return L_D^*

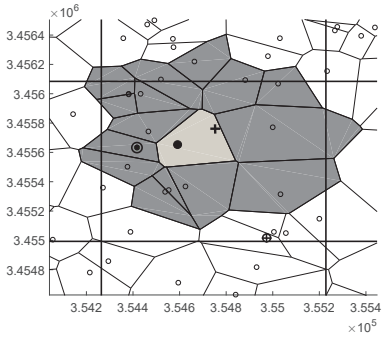

Fig. 9. Construction of the spatial cloaking region.

B. The Enhanced Temporal Cloaking

To overcome the inference attack against the baseline temporal cloaking, we need to obfuscate the time such that the possible travel distance of a driver within the given time covers the whole cloaking region. Otherwise, some part of the cloaking region can be pruned by the SP, because the driver cannot reach them at his maximum speed.

The enhanced temporal cloaking algorithm is described in Algorithm 2. In this algorithm, the cloaking region is represented by a set of points X, which represent the vertices of the Voronoi polygons lying inside the cloaking region. The SP using some statistics can determine the maximum speed v of a driver within this region. In addition, the SP knows the time of the last actual location reported by the driver before entering the rider's cloaking region, denoted by t_d.

In Line 1 $\sim$ 2, first we use the randomized incremental algorithm proposed in [18] to compute the smallest enclosing circle of the points X. As the smallest enclosing circle has at least two points of X on its boundary, the maximum traveling distance between any two points inside the circle, denoted by d_{max}, corresponds to the obtained circle diameter. Then in Line 3, t denotes the time delay which should be added to obfuscate the actual time and is computed according to d_{max}. Finally, we set the obfuscated time according to Line 4 in the algorithm. A random value r is added to the obfuscated time to prevent the SP from knowing d_{max} that reveals information about the cloaking region. Hence, the proposed cloaking algorithm runs in time of $O(n)$.

C. Privacy Analysis

We formalize our enhanced privacy preserving ride-hailing solution by comparing the probabilities of rider being located in different Voronoi polygons within cloaking region O, after observing the chosen driver's location L_D^*. Let P_i, and P_j be two Voronoi polygons that lie inside O, then the ideal case is that $\frac{\mathrm{P}(P_i|L_D^*)}{\mathrm{P}(P_j|L_D^*)} \leq w$. For $w = 1$, the SP may deduce that the rider has equal probability of being located in P_i or P_j . However, the SP may have side information on the prior probability of the rider being located in each polygon, represented by $\mathrm{P}(P_i)$, and $\mathrm{P}(P_j)$. For example, the probability that the rider being located in a shopping area may be higher

Algorithm 2: ENHANCED TEMPORAL CLOAKING

Input: cloaking region represented by a set of points X, maximum driver speed v, last known time where driver's actual location is reported t_d, a random value r.

Output: an obfuscated time t'

compute the smallest enclosing circle for the set of points X
set d_{max} to the circle diameter
set t to $\frac{d_{max}}{v}$
set t' equals $t_d + t + r$
return t'

than the probability of being located in a lake. Hence, we can formalize our privacy preserving solution as follows:

Theorem IV.2. *The enhanced privacy preserving ride-hailing solution guarantees strong privacy for rider's actual location inside a cloaking region O iff it satisfies the following for all prior probabilities and any chosen driver L_D^* from S.*

$$\frac{\mathrm{P}(P_i|L_D^*)}{\mathrm{P}(P_j|L_D^*)} \leq w.\frac{\mathrm{P}(P_i)}{\mathrm{P}(P_j)}, \qquad \forall P_i, P_j \in O. \tag{3}$$

Proof. The proof is by contradiction. Assume it is safe for a rider to release the cloaking region O to the SP. Then the SP knows the locations of drivers inside O, and it also knows that the algorithm uses the location information of nearby drivers to generate O. Therefore, it may deduce that the rider tends to be close to the center of the cloaking region. With such information, the SP can improve the inference of the rider's actual location as follows. For each driver's location L_{D_i}, it computes the distances between that location and all other drivers' locations, denoted by $D_i = \{dist_j(L_{D_i}, L_{D_j})\}_{\forall j \neq i}$. Then it obtains the variance of the computed distances, represented by $\mathrm{Var}(D_i) = \mathrm{E}[(D_i - \mu)^2]$. Finally, the SP chooses the Voronoi polygon of the driver's location with the smallest variance, i.e., $\arg\min_i \mathrm{Var}(D_i)$. The chosen polygon contains the rider's actual location with high probability, and thus the inference of the rider's location can be improved.

However, in our solution, a rider releases the chosen driver's location L_D^* only to the SP, and no more information is released about the cloaking region. We obtain a contradiction and thus, the described inference attack is not possible without knowing the cloaking region O. Hence, the observed L_D^* has limited effect to the probabilities deduced by the SP. □

V. PERFORMANCE EVALUATIONS

In this section, we evaluate the performance of the enhanced privacy preserving solution and compare it to PrivateRide. We first introduce our evaluation setup and then show the evaluation results.

A. Evaluation Setup

The evaluations are based on a real dataset consisting of mobility traces of taxi cabs collected during a single day in Shanghai, China [13]. The dataset includes time-stamped location traces collected using GPS devices. In addition, each record includes an *OCCUPIED* variable that indicates whether the taxi is vacant or occupied. It is equal to 1 when the taxi is occupied and 0 otherwise. Hence, the ride records are a sequence of consecutive records where *OCCUPIED* equals to 1.

A ride is defined by the start time, end time, pick-up location, drop-off location, ride time, and ride distance. The start time and the pick-up location are obtained from the first record of the ride records. The end time and the drop-off location are obtained from the last record of the ride records. The ride distance is calculated using the haversine formula [19], as the distance between the pick-up and drop-off location. The total number of rides in the dataset is 60000. We filter out rides that have a total distance of zero or last for less than 5 minutes, and we only consider rides inside an area of $\approx$ (44 km $*$ 44 km). Thus, for our experiments we use 25000 rides.

We compare our enhanced solution with PrivateRide [8] by evaluating the accuracy of their ride matching. In the evaluation results, we call our enhanced ride matching *Our algorithm*. The performance of both solutions is evaluated based on a metric called *Relative Extra Distance*, which measures the extra distance a driver has to drive compared to the distance covered by the closest driver to the rider's location. We set $R = 4000$ m if not mentioned otherwise.

TABLE I
COMPUTATIONAL TIME OF OUR ALGORITHM

Setting	Algorithm Computational Time
$R = 500$ m	(1.12 ± 0.21) ms
$R = 1000$ m	(1.41 ± 0.36) ms
$R = 2000$ m	(2.64 ± 0.82) ms
$R = 5000$ m	(8.66 ± 2.20) ms

B. Evaluation Results

In this section, we present the experimental evaluation results. For comparison purpose, we implemented the cloaking algorithm in PrivateRide, and we set the area of the cloaking region to A_R. For instance, if A_R = 62500 m^2, the region is divided into square cells of $250 * 250$ m^2.

1) Effect of the privacy preference A_R and the weight factor w on the ride matching accuracy: In Figure 10, we show the effect of A_R and w on ride matching accuracy measured by the *Relative Extra Distance* metric. The results show the cumulative distribution function of the *Relative Extra Distance*, which reflects the extra cost for both drivers and riders. It can be observed that by imposing higher privacy preference by requiring larger A_R, the matching accuracy decreases. In addition, we can observe that the increase in w contributes to an increase in the matching accuracy, and hence a decrease in the extra distance.

Compared to PrivateRide, our algorithm, under the same A_R, achieves higher matching accuracy. As A_R increases, the

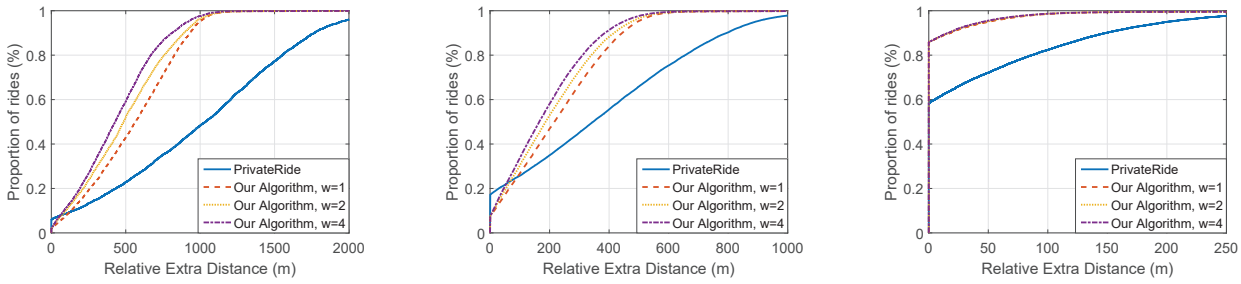

Fig. 10. Effect of A_R on ride matching accuracy (*Relative Extra Distance*). (a) $A_R = 2000^2$ m^2, (b) $A_R = 1000^2$ m^2, and (c) $A_R = 250^2$ m^2

accuracy gap between our algorithm and PrivateRide increases, which shows that our algorithm outperforms PrivateRide. As shown in Figure 10 (a), when $A_R = 2000^2$ m^2, in 80% of the rides, the *Relative Extra Distance* of our algorithm under different w's is up to: 831 m for $w = 1$, 760 m for $w = 2$, and 675 m for $w = 4$, while it is up to 1560 m in PrivateRide. In this case, comparing our algorithm with $w = 4$ to PrivateRide, the savings in the extra cost can be calculated as follows: $\frac{1560-675}{1560} = 56.7\%$. While as shown in Figure 10 (b), when $A_R = 1000^2$ m^2, in 80% of the rides, the *Relative Extra Distance* of our algorithm with $w = 1$ is up to 380 m, but up to 666 m in PrivateRide.

2) Computational overhead: We evaluate the computational time of the algorithm when $A_R = 62500$ m^2. We study the effect of R on the computational time. As shown in Table I, for a rider, the computational overhead introduced by our algorithm varies depending on R. It can be observed that the time overhead in our algorithm is small, especially when R ranges from 500 m to 2000 m, it is only ≈ 1.12 ms to 2.64 ms. As A_R and R increase, the time overhead increases. However, even with $A_R = 2500^2$ m^2, R can be equal to 5000 m, and in this case the time overhead is still small to be noticed by a rider. In PrivateRide, riders receive a single driver selected by the SP, hence there is no computational overhead.

3) Effect of the privacy preference A_R on the time delay t due to temporal cloaking: We evaluate the time delay t generated by the enhanced temporal cloaking algorithm. We study the effect of A_R on the introduced time delay. As shown in Figure 11, the time delay introduced by the temporal cloaking algorithm varies with A_R. As observed, the time delay varies and it is small, especially when A_R ranges from 250^2 m^2 to 500^2 m^2. In $\approx 90\%$ of the cases, the time delay is less than 2 minutes. In PrivateRide, the temporal cloaking introduces fixed time delay. However, as shown in Section III.B, the SP can improve the inference of riders' locations with the knowledge of such static temporal cloaking.

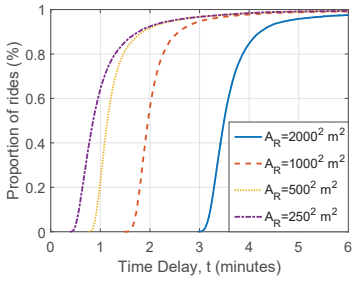

Fig. 11. Time delay introduced by the enhanced temporal cloaking algorithm for different A_R.

VI. Related Work

In the line of privacy preserving solutions for ride-hailing services, previous works have addressed privacy issues in ride-sharing services [20]–[23]. For example, Friginal *et. al* [20] proposed a solution for protecting location privacy in dynamic ride-sharing services. They consider a distributed architecture which allows users' location information to be scattered around the network, hence preventing the SP from collecting and storing sensitive location information from ride-sharing users. Ni *et. al* [21] proposed a protocol to solve the contradiction between safety and privacy preservation of riders and drivers. However, ride-sharing is different from ride-hailing. Ride-sharing services enable multiple individuals with similar trip schedules to share a single car along their route. While ride-hailing services allow users to use their own cars as taxies, so they can pick up and drop off riders at their specified pick-up and drop-off locations.

Recently, researchers started to look into privacy issues in ride-hailing services. PrivateRide [8] is the first system that aims to enhance location privacy for riders. It relies on spatial cloaking algorithms to obfuscate locations of riders and drivers by replacing their actual locations with cloaked regions. Then, in order to match a rider to a driver, the SP measures the distances between the rider's pick-up location and the driver's locations. Based on the calculated distances, the rider is assigned to the driver who is the closest to his cloaked region. Since the ride matching is based on the cloaked regions, instead of the actual locations, the chosen driver may not be the optimal one and has to drive extra distance for rider pickup, and the rider may wait extra time to get the ride. As another solution, ORide [24] relies on homomorphic encryption to encrypt riders' and drivers' locations before sending them to the SP. Then the SP computes the encrypted

distances based on the encrypted locations and returns them to the rider. The rider decrypts the distance and selects the closest driver. Although ORide achieves better matching results compared to PrivateRide, its cryptographic solution incurs much higher computation overhead. Moreover, this solution encrypts drivers' locations; as a result, the SP does not know the locations of the drivers. Hence, this model is not compatible with the current ride-hailing services and not incrementally deployable.

In this paper, we focus on protecting riders' pick-up and drop-off locations which can be used to infer sensitive information such as where riders live and work. Protecting the whole riders' mobility trace is out of the scope of this work. There are existing works related to anonymizing users' mobility traces [25]. Despite the existence of these anaonymization solutions, the work in [26], [27] highlights the privacy vulnerability of anonymized traces by exploiting users' side information such as users' observed locations and co-locations.

VII. Discussions

One of the challenges in adapting our algorithm is how to preserve rider's location privacy at the drop-off location. After the ride ends, normally the driver continuously reports his actual location to the SP in order to receive new ride requests. However, by knowing the driver actual location after the drop-off, the SP can infer rider's drop-off location. To protect rider's drop-off location, one solution is to let the driver drive outside of the rider's cloaking region before reporting his actual location. Another solution is to wait for some time proportional to the cloaking region size before reporting the actual location. In both solutions, the driver would not be able to accept new rides for some time, which may adversely affect his business. To compensate for that, the riders who care for their location privacy should pay the driver an amount proportional to the time he has to wait or the extra distance he has to drive out of the cloaking region.

VIII. Conclusions

In this paper, we addressed the problem of preserving riders' location privacy in ride-hailing services. We presented two solutions, the baseline solution and the enhanced privacy preserving ride-hailing solution. Although the baseline solution can provide riders with personalized location privacy, we identified two inference attacks against it. Then, we presented an enhanced solution which relies on enhanced ride matching and temporal cloaking algorithms to deal with these inference attacks. The enhanced solution provides riders with personalized location privacy while limiting the matching accuracy loss. Experimental results showed that our solution outperforms previous work by providing more accurate matching results with negligible computational overhead.

Acknowledgments

This work was supported in part by the National Science Foundation (NSF) under grants CNS-1618684, CNS-1421578, and CNS-1526425.

References

[1] P. Golle and K. Partridge, "On the anonymity of home/work location pairs," *Pervasive computing*, 2009.

[2] M. Feeney and R. Companies Uber, "Is ridesharing safe?" *Cato Policy Analysis*, 2015.

[3] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in *IEEE Security and privacy*, 2011.

[4] M. L. Damiani, E. Bertino, C. Silvestri *et al.*, "The probe framework for the personalized cloaking of private locations." *Trans. Data Privacy*, 2010.

[5] M. Xue, P. Kalnis, and H. K. Pung, "Location diversity: Enhanced privacy protection in location based services," in *International Symposium on Location-and Context-Awareness*, 2009.

[6] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *IEEE INFOCOM*, 2014.

[7] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Enhancing privacy through caching in location-based services," in *IEEE INFOCOM*, 2015.

[8] A. Pham, I. Dacosta, B. Jacot-Guillarmod, K. Huguenin, T. Hajar, F. Tramèr, V. Gligor, and J.-P. Hubaux, "Privateride: A privacy-enhanced ride-hailing service," *Proceedings on Privacy Enhancing Technologies*, 2017.

[9] S. Fortune, "Voronoi diagrams and delaunay triangulations," *Computing in Euclidean geometry*, 1992.

[10] F. Aurenhammer, "Voronoi diagrams-a survey of a fundamental geometric data structure," *ACM Computing Surveys (CSUR)*, 1991.

[11] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar, "Preserving user location privacy in mobile data management infrastructures," *Lecture Notes in Computer Science*, 2006.

[12] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[13] S. Liu, Y. Liu, L. M. Ni, J. Fan, and M. Li, "Towards mobility-based clustering," in *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010.

[14] D.-T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, 1980.

[15] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *Visualization and mathematics III*, 2003.

[16] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, 1987.

[17] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*. Macmillan London, 1976.

[18] E. Welzl, "Smallest enclosing disks (balls and ellipsoids)," *New results and new trends in computer science*, 1991.

[19] C. Veness, "Movable type scripts," http://www.movable-type.co.uk/scripts/latlong.html, last accessed July 2017.

[20] J. Friginal, S. Gambs, J. Guiochet, and M.-O. Killijian, "Towards privacy-driven design of a dynamic carpooling system," *Pervasive and mobile computing*, 2014.

[21] J. Ni, K. Zhang, X. Lin, H. Yang, and X. S. Shen, "Ama: Anonymous mutual authentication with traceability in carpooling systems," in *IEEE ICC*, 2016.

[22] C. Stach and B. Mitschang, "Privacy management for mobile platforms–a review of concepts and approaches," in *IEEE MDM*, 2013.

[23] U. M. Aïvodji, S. Gambs, M.-J. Huguet, and M.-O. Killijian, "Meeting points in ridesharing: A privacy-preserving approach," *Transportation Research Part C: Emerging Technologies*, 2016.

[24] A. Pham, I. Dacosta, G. Endignoux, J. R. T. Pastoriza, K. Huguenin, and J.-P. Hubaux, "Oride: A privacy-preserving yet accountable ride-hailing service," in *26th USENIX Security Symposium*, 2017.

[25] C.-Y. Chow and M. F. Mokbel, "Trajectory privacy in location-based services and data publication," *ACM Sigkdd Explorations Newsletter*, 2011.

[26] Y. Khazbak and G. Cao, "Deanonymizing mobility traces with co-location information," in *IEEE CNS*, 2017.

[27] C. Y. Ma, D. K. Yau, N. K. Yip, and N. S. Rao, "Privacy vulnerability of published anonymous mobility traces," *IEEE/ACM Transactions on Networking*, 2013.