Hindawi
Security and Communication Networks
Volume 2020, Article ID 4948387, 13 pages
https://doi.org/10.1155/2020/4948387

WILEY | Hindawi

Research Article

Hail the Closest Driver on Roads: Privacy-Preserving Ride Matching in Online Ride Hailing Services

Haining Yu, Hongli Zhang, and Xiangzhan Yu

School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

Correspondence should be addressed to Haining Yu; yuhaining@hit.edu.cn

Received 19 September 2019; Revised 17 February 2020; Accepted 4 May 2020; Published 12 June 2020

Academic Editor: Prosanta Gope

Online ride hailing (ORH) services enable a rider to request a driver to take him wherever he wants through a smartphone app on short notice. To use ORH services, users have to submit their ride information to the ORH service provider to make ride matching, such as pick-up/drop-off location. However, the submission of ride information may lead to the leakages of users' privacy. In this paper, we focus on the issue of protecting the location information of both riders and drivers during ride matching and propose a privacy-preserving online ride matching scheme, called pRMatch. It enables an ORH service provider to find the closest available driver for an incoming rider over a city-scale road network, while protecting the location privacy of both riders and drivers against the ORH service provider and other unauthorized participants. In pRMatch, we compute the shortest road distance over encrypted data by using road network embedding and partially homomorphic encryption and further efficiently compare encrypted distances by using ciphertext packing and shuffling. The theoretical analysis and experimental results demonstrate that pRMatch is accurate and efficient, yet preserving users' location privacy.

1. Introduction

The widespread adoption of smartphones embedded with GPS, combined with the availability of digital road maps, provides the necessary enabling technologies for online ride hailing (ORH) services, such as Uber (http://www.uber.com), Lyft (http://www.lyft.com), and DiDi Chuxing (http://www.didiglobal.com). As reported in [1], 30% urban American adults had used ORH services, which have already far out-paced the growth of traditional carsharing services of the past. The general feature of ORH services is the ability for a rider to request a driver to take him exactly where he needs to go, via his ORH app. Upon receiving a rider's request, the ORH service provider makes ride matching between the rider and available drivers and forwards the request to the closest driver.

Unfortunately, along with the high convenience of ORH services come some important privacy concerns [2, 3]. To provide better services, ORH services require users to submit ride information to make ride matching, such as riders' pick-up and drop-off locations. This private information is usually sensitive and can be used to identify an individual or infer his/her daily activity. For example, if a driver Bob accepts a ride request: picking up a rider Alice at a particular location l at time t, it reveals that both Bob and Alice will be at location l at time t. Actually, an ORH service provider is not always fully trusted, it might collect users' ride information to profile them or infer additional sensitive information from harvested data for purpose of economic benefit. Furthermore, even if an ORH service provider is honest, it may be attacked by other adversaries. Nowadays, risks of data leakage happen more frequently than ever before. The leakages of users' ride information may pose threats on users' financial or personal safety. Therefore, it is a major technical challenge to enable ORH services while protecting users' privacy.

Many privacy-enhanced solutions for ORH services are proposed to protect users' privacy during ride matching. These solutions are based on either nonencryption or encryption. Nonencryption solutions usually employ spatial cloaking [4], geographic masking [5], mix-zone [6], and differential privacy [7] to trade-off between the level of

privacy preserving and the quality of ORH service. These solutions are of high-efficiency, but provide limited privacy preservation and suffer from distortion of location information. To improve quality of services, encryption solutions are proposed, which provide strong privacy preservation at the cost of heavy computation and communication cost. These encryption-based solutions integrate well-established cryptographic tools, such as private information retrieval (PIR) [8], secure multiparty computation (SMC) [9], private set intersection (PSI) [10], somewhat homomorphic encryption (SHE) [11], and partially homomorphic encryption (PHE) [12] to accomplish private distance measurements for ride matching. For efficiency, most existing encryption-based solutions use Euclidean distance as the travel cost metric, such as PrivateRide [4], ORide [13], and TRACE [14]. However, Euclidean distance may lead to false hits, because drivers always travel along road network. As evaluated in our experiments, roughly 15% false hits exist when using Euclidean distance to make ride matching. Actually, road distance should be used to evaluate the travel cost between riders and drivers, which can achieve a higher accuracy. As studied in [15, 16], it is a complex problem to compute the shortest road distance under city-scale road networks in the encrypted form. Existing secure shortest road distance approaches are not efficient enough to support online ride matching, which requires heavy cost on shortest road distance computation in real time. It is desired that there exists an encryption-based scheme to efficiently make ride matching for ORH services by using road distance.

In this paper, we focus on the issue of the leakage of user's location privacy during ride matching and propose a privacy-preserving online ride matching scheme, called pRMatch. It enables an ORH service provider to find the closest available driver for an incoming rider over a city-scale road network, while protecting the location privacy of both riders and drivers against the ORH service provider and other unauthorized participants. We summarize the following main contributions:

(i) We propose a privacy-preserving ride matching scheme for ORH services (pRMatch), which enables an ORH service provider to select the closest driver for an incoming rider by using approximate road distance, while preventing users' location privacy from disclosing to the ORH service provider and other curious participants.

(ii) We propose an efficient shortest road distance computation approach, which computes the shortest road distance over encrypted data by using road network embedding and partially homomorphic encryption.

(iii) We design a secure comparison protocol, which efficiently compares encrypted distances by using ciphertext packing and shuffling.

(iv) We implement pRMatch and perform extensive experiments to validate its accuracy and efficiency. Experimental results demonstrate that pRMatch is secure without hampering the functionality of ORH services.

The remainder of this paper is structured as follows: in Section 2, we describe the system model and give the problem definition; in Section 3, we list necessary preliminaries; in Section 4, we describe pRMatch scheme in details; in Section 5, we discuss theoretical analysis of pRMatch; in Section 6, we present the evaluation; in Section 7, we review the related literature. Finally, we summarize this paper.

2. Models and Problem Definition

2.1. System Model. Over a city-scale road network, pRMatch provides strong privacy guarantees to both drivers and riders during ride matching, without sacrificing the usability of the ORH services. As shown in Figure 1, pRMatch involves four participants:

(i) *SP*. The ORH service provider (SP) handles incoming ride hailing requests and matches riders with available drivers, based primarily on their encrypted locations.

(ii) *Proxy*. The proxy is a third party that offers efficient key management. With the proxy, heavy cryptographic operations can be relieved from users.

(iii) *Rider*. A rider u is the user who submits his encrypted pick-up location l_u to the SP to request a nearby available driver.

(iv) *Driver*. A driver d is the user who updates his encrypted current location l_d to the SP and waits for a new rider to serve.

Here, we assume a ride/driver has a smartphone embedded with GPS to obtain own location at anytime and anywhere, and he also installs and registers to an ORH App offered by the SP.

In our system model, we consider single-rider multi-driver ride matching, which prefers instant feedback to the users without batch processing requests at a fixed time interval. The batch processing may achieve better system optimization but sacrifice the user experience, e.g., a rider may wait for a while to retrieve the matching result. This single-rider multidriver ride matching is widely practised in user-centered services, such as [4, 13, 14, 17], which do not try to violate user experience to improve the overall system optimization.

2.2. Threat Model. The threat model of pRMatch is assumed as follows:

(i) The SP and the proxy are *honest-but-curious*, which try to learn additional information from received message and intermediate information during the scheme execution.

(ii) Users (including riders and drivers) are also *honest-but-curious*. That is, riders submit valid request and drivers update correct locations, but they try to harvest other unmatched users' location information.

(iii) There is *no collude* among the participants, including the SP and the proxy, the SP and users, the

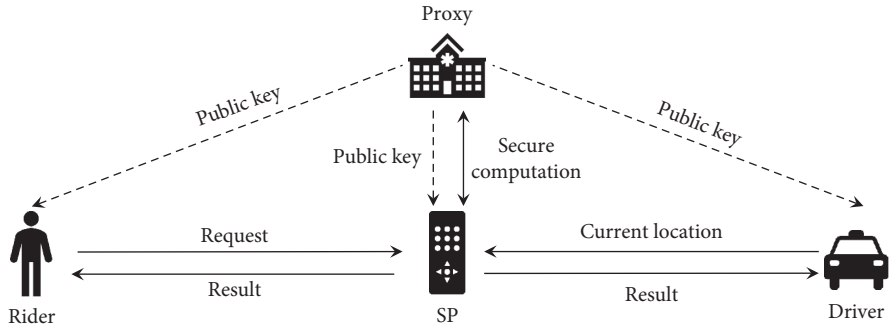

Figure 1: The system model of pRMatch.

proxy and users. In practice, the SP and the proxy are usually a large service provider (e.g., Uber and Google) which understand the importance of reputation. Active attacks like collusion are easy to detect and will seriously damage their reputation once caught.

(iv) Drivers are independent contractors rather than the SP's employees; thus, the SP has no authority to access to the location information they hold.

(v) No external adversary can truncate or tamper the communications among the participants.

pRMatch focuses on the following representative attacks inspired from [2, 13]:

(i) *A*1 (*SP/Proxy* ⟶ *D/R*, *Daily Routines Tracking Attack*). The SP or the proxy might attempt to precisely track riders or drivers during their daily routines online or offline.

(ii) *A*2 (*SP/Proxy* ⟶ *D/R*, *Large-Scale Inference Attack*). The SP or the proxy might attempt to collect rides' or drivers' rides deliberately and perform a large-scale inference attack to learn additional users' privacy, e.g., their home/work addresses, behaviors, and interests.

(iii) *A*3 (*D/R* ⟶ *R/D*, *Unauthorized Ride Harvesting Attack*). A driver/rider might attempt to harvest additional unauthorized ride information of other unmatched users.

2.3. Problem Definition. The problem that we focus on is defined as follows: given a set of drivers D that travel along the road network and an incoming rider u, to find the driver with minimum road distance to serve the rider, while preserving the location privacy of both the rider and the drivers. The problem is represented as

$$d^* = \text{argmin}_{d_k \in D} \text{dist}_R\left(l_u, l_{d_k}\right), \tag{1}$$

where l_u is the pick-up location, l_{d_k} is the current location of driver d_k, and $\text{dist}_R(\cdot,\cdot)$ represents the shortest road distance between two locations.

Our design goals contain the following three-fold:

(i) *Accuracy*. pRMatch should provide accurate ride matching results. That is, the matched driver is the closest one to serve the given request with a high probability.

(ii) *Efficiency*. pRMatch should be efficient to support large-scale ORH services. That is, the computation and communication cost should be low enough to make ride matching over large-scale users.

(iii) *Privacy Preservation*. In pRMatch, the location privacy of both riders and drivers should be secret to the SP, the proxy, and other users beyond authorization.

3. Preliminaries

3.1. Paillier Cryptosystem. The Paillier cryptosystem [12] is a widely used PHE that supports additive homomorphic encryption. We brief it to help illustrate and understand our scheme as follows:

(i) *Key Generation* $((pk, sk) \leftarrow \text{KeyGen}(1^\lambda))$. Choose two primes p, q and compute $N = p \times q$ and $\lambda = \text{lcm}(p-1, q-1)$. Then, select a random $g \in \mathbb{Z}^*_{N^2}$ such that $\gcd(L(g^\lambda \bmod N^2), N) = 1$, where $L(x) = (x-1)/N$. The public key and private key are $pk = (N, g)$ and $sk = \lambda$, respectively.

(ii) *Encryption* $(c \leftarrow E(m, pk))$. Let $m \in \mathbb{Z}_N$ be a plaintext and $r \in \mathbb{Z}_N$ be a random number. The ciphertext is given by $c = E(m \bmod N; r \bmod N) = g^m r^N \bmod N^2$.

(iii) *Decryption* $(m \leftarrow D(c, sk))$. Given a ciphertext $c \in \mathbb{Z}_{N^2}$, the corresponding plaintext can be derived as

$$m = \frac{L(c^\lambda \bmod N^2)}{L(g^\lambda \bmod N^2)} \bmod N. \tag{2}$$

The Paillier cryptosystem has homomorphism properties: for any $m_1, m_2, r_1, r_2 \in \mathbb{Z}_N$, we have

$$\begin{aligned} E(m_1, r_1) \cdot E(m_2, r_2) &= E(m_1 + m_2, r_1 r_2) \bmod N^2, \\ E^{m_2}(m_1, r_1) &= E(m_1 m_2, r_1^{m_2}) \bmod N^2. \end{aligned} \tag{3}$$

3.2. Road Network Embedding. The road network embedding (RNE) technique [18] is used to convert the road network into a high-dimensional space, where complex shortest road distance computation can be converted to simple computation supported by existing cryptographic primitives.

A road network can be modeled as a weighted graph $\mathcal{G} = (V, E, W)$. Let $|V|$ be the number of vertices in $\mathcal{G}$ (i.e., road intersections) and $|E|$ be the number of edges in $\mathcal{G}$ (i.e., road segments). Define $\mathbf{R}$ as a set of $O(\log^2|V|)$ subsets of V, i.e.,

$$\mathbf{R} = \{V_{1,1}, \ldots, V_{1,\alpha}, \ldots, V_{\beta,1}, \ldots, V_{\alpha,\beta}\}, \tag{4}$$

where $\alpha = \beta = O(\log|V|)$ and $V_{i,j}$ is a subset composed of 2^i random vertices of V. Given a vertex v and a subset $V_{i,j}$, the shortest road distance between them is defined as

$$\mathrm{dist}_R(v, V_{i,j}) = \min_{v' \in V_{i,j}} \mathrm{dist}_R(v, v'), \tag{5}$$

where $\mathbf{R}$ defines a high-dimensional embedding space, where the coordinate of a vertex $v \in V$ is a $O(\log^2|V|)$-dimensional vector:

$$\begin{aligned}\mathbf{c}_v = \langle &\mathrm{dist}_R(v, V_{1,1}), \ldots, \mathrm{dist}_R(v, V_{1,\alpha}), \ldots, \\ &\mathrm{dist}_R(v, V_{\beta,1}), \ldots, \mathrm{dist}_R(v, V_{\alpha,\beta})\rangle.\end{aligned} \tag{6}$$

The embedded road network of $\mathcal{G}$ is represented as $\Omega = \{\mathbf{c}_v \mid v \in V\}$.

For a location l on the road segment $(v_s, v_d) \in E$, its coordinate can be represented as

$$\begin{aligned}\mathbf{c}_l = \langle &\mathrm{dist}_R(l, V_{1,1}), \ldots, \mathrm{dist}_R(l, V_{1,\alpha}), \ldots, \\ &\mathrm{dist}_R(l, V_{\beta,1}), \ldots, \mathrm{dist}_R(l, V_{\alpha,\beta})\rangle,\end{aligned} \tag{7}$$

where $\mathrm{dist}_R(l, V_{i,j}) = \min\{\mathrm{dist}_R(l, v_s) + \mathrm{dist}_R(v_s, V_{i,j}), \mathrm{dist}_R(l, v_d) + \mathrm{dist}_R(v_d, V_{i,j})\}$.

Without loss of generality, we assume the dimension of the embedded network is ω, s.t., $\omega \le \lceil \log^2|V| \rceil$. Given two locations l_s and l_d, the shortest distance between them can be approximated through computing the chessboard distance between $\mathbf{c}_{l_s}$ and $\mathbf{c}_{l_d}$, denoted by

$$\begin{aligned}\mathrm{dist}_R(l_s, l_d) &\approx \mathrm{dist}_C(c_{l_s}, c_{l_d}) \\ &= \max_{V_{i,j} \in R} \left|\mathrm{dist}_R(l_s, V_{i,j}) - \mathrm{dist}_R(l_d, V_{i,j})\right| \\ &= \max_{1 \le i \le \omega} \left|c_{l_s}[i] - c_{l_d}[i]\right|,\end{aligned} \tag{8}$$

where $\mathrm{dist}_C(\cdot, \cdot)$ represents the chessboard distance between two coordinates.

4. pRMatch Scheme

In this section, we present our pRMatch scheme. We begin with an overview of the scheme and then detail its operations.

4.1. Overview. As shown in Figure 1, the proxy generates keys and broadcasts the public key to other participants. Drivers periodically report to the SP the geographical zones where they are located. These zones are defined by the SP to prefilter drivers quickly. When a rider wants to hail a driver, he generates the encrypted request and sends it to the SP. Then, the SP prefilters all drivers on the basis of zone information and obtains candidate drivers. All candidate drivers send their encrypted locations to the SP. Upon received above ciphertexts, the SP computes all shortest road distances from the candidate drivers to the rider in encrypted form. The SP and the proxy perform secure comparison protocol to select the driver with the minimum road distance to serve the rider. Finally, the ride matching result is sent to the matched rider and driver.

4.2. Ride Prerequisites. Given the road network $\mathcal{G} = (V, E, W)$ and the dimension ω of the embedded road network, several offline initialization works need to be executed:

(i) The proxy generates a pair of keys (pk, sk). The public key pk is open to other participants, and the secret key sk is kept by itself.

(ii) The SP computes the coordinate of each vertex in $\mathcal{G}$ to generate the embedded road network $\Omega = \{\mathbf{c}_v \mid v \in V\}$ by using RNE technique introduced in Section 3.2, where each coordinate is represented as a ω-dimensional vector. To support drivers prefiltering, the SP partitions the road network $\mathcal{G}$ into the set of zones $G = \{z_i\}_{1 \le i \le n}$. Note that Ω and G are public to all users.

(iii) A driver d_k periodically reports to the SP the geographical zone z_{d_k} where he is located.

4.3. Ride Matching. When a rider hails a driver using the ORH service, the operations performed by the rider, the drivers, the SP, and the proxy are as follows (Figure 2):

(1) The rider u first computes the coordinate of his pick-up location l_u in the embedded road network Ω according to equation (7), denoted as $\mathbf{c}_u = (\mathbf{c}_u[1], \ldots, \mathbf{c}_u[\omega])$, and the zone where he is located, denoted as z_u. Then, the rider encrypts each element in $\mathbf{c}_u$ with the public key pk using Paillier cryptosystem, denoted as

$$[\![\mathbf{c}_u]\!] = (E(\mathbf{c}_u[1]), \ldots, E(\mathbf{c}_u[\omega])). \tag{9}$$

Finally, the rider u sends his request $R_u = ([\![\mathbf{c}_u]\!], z_u)$ to the SP.

(2) Upon receiving the request R_u, the SP selects the candidate drivers D^* that is located in the zone z_u and the adjacent zones of z_u. Then, the SP asks the candidate drivers to update their encrypted current locations.

(3) For $d_k \in D^*$, the driver d_k computes the coordinate of his current location l_{d_k} in the embedded road network Ω according to equation (7), denoted as

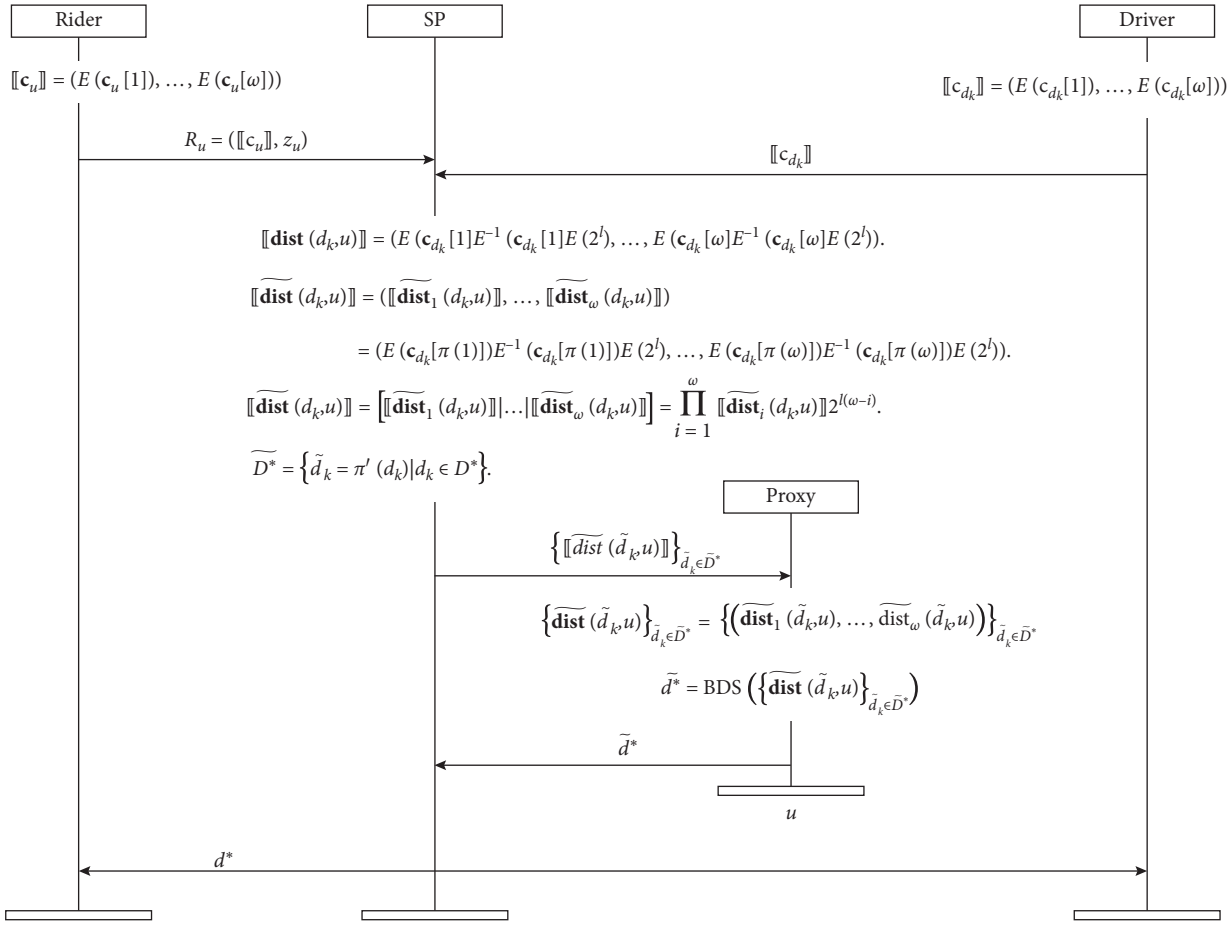

Figure 2: The framework of pRMatch.

$\mathbf{c}_{d_k} = (\mathbf{c}_{d_k}[1], \cdots, \mathbf{c}_{d_k}[\omega])$. Then, the driver encrypts each element in $\mathbf{c}_{d_k}$ with the public key pk using Paillier cryptosystem, denoted as

$$[\![\mathbf{c}_{d_k}]\!] = \left(E\left(\mathbf{c}_{d_k}[1]\right), \ldots, E\left(\mathbf{c}_{d_k}[\omega]\right)\right). \tag{10}$$

Finally, the driver d_k sends his encrypted current location $[\![\mathbf{c}_{d_k}]\!]$ to the SP.

(4) Upon receiving $\{[\![\mathbf{c}_{d_k}]\!]\}_{d_k \in D^*}$, the SP computes the shortest road distances from all candidate drivers D^* to the rider u in ciphertext domain. Assume the upper bound of road distance in the road network is a ℓ-bit integer. Given $d_k \in D^*$, the SP computes the shortest road distance between l_{d_k} and l_u according to equation (8). Concretely, the SP computes the following vector $[\![\mathbf{dist}(d_k, u)]\!]$ over ciphertexts $[\![\mathbf{c}_{d_k}]\!]$ and $[\![\mathbf{c}_u]\!]$ based on homomorphism properties of the Paillier cryptosystem:

$$\begin{aligned}[\![\mathbf{dist}(d_k,u)]\!] &= \left(E\left(\mathbf{c}_{d_k}[1] - \mathbf{c}_u[1] + 2^\ell\right), \ldots,\right.\\ &\quad \left.E\left(\mathbf{c}_{d_k}[\omega] - \mathbf{c}_u[\omega] + 2^\ell\right)\right)\\ &= \left(E\left(\mathbf{c}_{d_k}[1]\right)E^{-1}\left(\mathbf{c}_{d_k}[1]\right)E\left(2^\ell\right), \ldots,\right.\\ &\quad \left.E\left(\mathbf{c}_{d_k}[\omega]\right)E^{-1}\left(\mathbf{c}_{d_k}[\omega]\right)E\left(2^\ell\right)\right).\end{aligned} \tag{11}$$

Note that $E(2^\ell)$ is used to ensure all elements in $[\![\mathbf{dist}(d_k, u)]\!]$ are positive. To prevent the proxy from inferring the locations of u and d_k, the SP shuffles these encrypted values in $[\![\mathbf{dist}(d_k, u)]\!]$. For every element in the new vector, no one knows its position in the original vector. Consequently, the release of the index of the minimum in a shuffled vector is meaningless and does not reveal any private information. With the random permutation function π, the original vector $[\![\mathbf{dist}(d_k, u)]\!]$ is permuted into the shuffled vector:

$$\begin{aligned}[\![\widetilde{\mathbf{dist}}(d_k,u)]\!] &= \left([\![\widetilde{\mathbf{dist}}_1(d_k,u)]\!], \ldots, [\![\widetilde{\mathbf{dist}}_\omega(d_k,u)]\!]\right)\\ &= \left(E\left(\mathbf{c}_{d_k}[\pi(1)]\right)E^{-1}\left(\mathbf{c}_{d_k}[\pi(1)]\right)E\left(2^\ell\right), \ldots,\right.\\ &\quad \left.E\left(\mathbf{c}_{d_k}[\pi(\omega)]\right)E^{-1}\left(\mathbf{c}_{d_k}[\pi(\omega)]\right)E\left(2^\ell\right)\right).\end{aligned} \tag{12}$$

It is worth noting that this permutation prevents the proxy from knowing the original position of any element in $[\![\mathbf{dist}(d_k, u)]\!]$. To improve the efficiency, the SP uses ciphertext packing technique to pack multiple ciphertexts into one single ciphertext, because the plaintext space of the Paillier cryptosystem is much greater than that of the upper bound of road distance. In the Paillier cryptosystem, the number of slots in a

packed ciphertext is $p = [N/(\ell + 1)]$, that is, we can pack p ciphertexts into one single packed ciphertext. The basic idea of ciphertext packing is introduced as follows. Suppose $a_1, \cdots, a_l$ are ℓ-bit integers $(1 \le l \le p)$, their corresponding ciphertexts are $[\![a_1]\!], \cdots, [\![a_l]\!]$. We construct the packed ciphertext by

$$[[\![a_1]\!]|\cdots|[\![a_l]\!]] = \prod_{i=1}^{l} [\![a_i]\!]^{2^{\ell(l-i)}}. \tag{13}$$

We only need one decryption to obtain the packed plaintext

$$[a_1|\cdots|a_l] = \sum_{i=0}^{l} a_i 2^{l(l-i)}, \tag{14}$$

and then recover $a_1, \cdots, a_l$. Based on above ciphertexts packing, the SP packs all encrypted elements of $[\![\mathbf{dist}(d_k, u)]\!]$ into one single packed ciphertext:

$$\begin{aligned}[\![\widetilde{\mathrm{dist}}(d_k, u)]\!] &= \left[[\![\widetilde{\mathbf{dist}}_1(d_k, u)]\!]|\cdots|[\![\widetilde{\mathbf{dist}}_\omega(d_k, u)]\!]\right] \\ &= \prod_{i=1}^{\omega} [\![\widetilde{\mathbf{dist}}_i(d_k, u)]\!]^{2^{\ell(\omega-i)}},\end{aligned} \tag{15}$$

s.t., $p \ge \omega$. To protect the real identifiers of drivers against the proxy, the SP generates an one-time pseudonym for each driver $d_k \in D^*$, denoted as $\widetilde{d}_k = \pi'(d_k)$, and we have

$$\widetilde{D^*} = \left\{\widetilde{d}_k = \pi'(d_k) \,\middle|\, d_k \in D^*\right\}. \tag{16}$$

Finally, the SP sends a set of packed ciphertexts $\left\{[\![\widetilde{\mathrm{dist}}(\widetilde{d}_k, u)]\!]\right\}_{\widetilde{d}_k \in \widetilde{D^*}}$ to the proxy.

(5) Upon receiving these packed ciphertexts, the proxy decrypts each ciphertext in $\left\{[\![\widetilde{\mathrm{dist}}(\widetilde{d}_k, u)]\!]\right\}_{\widetilde{d}_k \in \widetilde{D^*}}$ with its secret key sk and then unpacks them to obtain

$$\left\{\widetilde{\mathrm{dist}}(\widetilde{d}_k, u)\right\}_{\widetilde{d}_k \in \widetilde{D^*}} = \left\{\left(\widetilde{\mathbf{dist}}_1(\widetilde{d}_k, u), \ldots, \widetilde{\mathbf{dist}}_\omega(\widetilde{d}_k, u)\right)\right\}_{\widetilde{d}_k \in \widetilde{D^*}}. \tag{17}$$

Then, the proxy runs Algorithm 1 to obtain the shortest road distances between the candidate drivers $\widetilde{D^*}$ and the rider u and then selects the driver with the minimum road distance as the matched driver, denoted as $\widetilde{d}^*$. Finally, the proxy sends $\widetilde{d}^*$ to the SP.

(6) When receiving $\widetilde{d}^*$, the SP can learn the real identifier of $\widetilde{d}^*$ is d^*. Then, the match result is sent to the driver d^* and the rider u.

5. Theoretical Analysis

5.1. Complexity Analysis. We analyze the online computation (comp.) cost and communication (comm.) cost for different participants in pRMatch. For the Paillier cryptosystem, we set N and g to 1024 bits and 160 bits, respectively. Under this assumption, one encryption (Enc) needs two 1024-bit exponentiations and one 2048-bit multiplication, and one decryption (Dec) costs essentially one 2048-bit multiplication, and a ciphertext is 2048 bits. Let mul and exp denote one 2048-bit multiplication and one 2048-bit exponentiation, respectively.

At the user side, a rider (resp. driver) needs to perform ω Enc to obtain the encrypted request (resp. current location). Accordingly, ω ciphertexts need to be transferred to the SP. Thus, the comm. cost between the SP and a user is roughly 2048ω bits.

At the SP side, the SP performs 2ω mul and ω exp to compute one shortest road distance. The total cost of the shortest road distance computation is $2\omega|D^*|$ mul and $\omega|D^*|$ exp. Besides, the SP packs $\omega|D^*|$ ciphertexts into $\lceil \omega|D^*|/p \rceil$ packed ciphertexts, where $p = \lfloor 1024/(\ell + 1) \rfloor$. The comp. cost of above packing is $\omega|D^*|$ mul and $\omega|D^*|$ exp. The SP sends $\lceil \omega|D^*|/p \rceil$ packed ciphertexts to the proxy, thus the comm. cost between the SP and the proxy is roughly $\lceil 2048\omega|D^*|/p \rceil$ bits.

At the proxy side, the proxy primarily performs $\lceil \omega|D^*|/p \rceil$ Dec to decrypt receiving ciphertexts. Afterwards, it returns one plaintext to the SP.

5.2. Security Analysis. The SP cannot harvest any rider's pick-up location $\mathbf{c}_u$ from $[\![\mathbf{c}_u]\!]$, and any driver's current location $\mathbf{c}_{d_k}$ from $[\![\mathbf{c}_{d_k}]\!]$, because they are encrypted by the Paillier cryptosystem that has semantic security. The SP only learns the zone information of users and the match result, rather than the exact locations of users.

The proxy also cannot learn any rider's pick-up location $\mathbf{c}_u$ and any driver's current location $\mathbf{c}_{d_k}$ from $\left\{\widetilde{\mathbf{dist}}(\widetilde{d}_k, u)\right\}_{\widetilde{d}_k \in \widetilde{D^*}}$, because each location coordinate in the road network embedded space has been shuffled and the real identifiers of drivers have been replaced by pseudonyms. Given $\widetilde{\mathbf{dist}}(\widetilde{d}_k, u)$, the proxy infers $\mathbf{dist}(\widetilde{d}_k, u)$ with the probability $(1/\omega!)$.

Next, we present an attack analysis to demonstrate that pRMatch effectively addresses the attacks described in Section 2.2.

(i) *Against A1.* To launch a *daily routines tracking attack* to a user, the SP/proxy needs to know his exact pick-up/drop-off location and time. In pRMatch, the drop-off event is not reported to the SP/proxy. As aforementioned, the SP cannot learn any rider's pick-up location or any driver's current location from the received ciphertext. Moreover, the SP only knows the rough pick-up time and the zone information, which is not enough to track the user. The proxy knows nothing about a ride, because each location coordinate have been shuffled. Above analysis indicates that pRMatch can tackle daily routines tracking attacks.

(ii) *Against A2.* To launch a *large-scale inference attack* to a user, the SP/proxy needs to learn his identity and at least his pick-up location and time. We have analyzed that the pick-up location of a rider and the current location of a driver are never exposed to the

Input: $\{\widetilde{\mathbf{dist}}(\widetilde{d}_k, u)\}_{\widetilde{d}_k \in \widetilde{D^*}} = \{(\widetilde{\mathbf{dist}}_1(\widetilde{d}_k, u), \cdot, \widetilde{\mathbf{dist}}_\omega(\widetilde{d}_k, u))\}_{\widetilde{d}_k \in \widetilde{D^*}}$.
Output: $\widetilde{d}^*$.

(1) for $1 \le k \le |\widetilde{D^*}|$ do
(2) for $1 \le i \le \omega$ do
(3) if $\widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u) < 2^\ell$ then
(4) $\widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u) \leftarrow 2^\ell - \widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u)$;
(5) else
(6) $\widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u) \leftarrow \widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u) - 2^\ell$;
(7) end if
(8) if $i = 1$ or $\mathrm{dist} < \widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u)$ do
(9) $\mathrm{dist} \leftarrow \widetilde{\mathbf{dist}}_i(\widetilde{d}_k, u)$;
(10) end if
(11) end for
(12) if $k = 1$ or $\mathrm{dist}^* > \mathrm{dist}$ do
(13) $\mathrm{dist}^* \leftarrow \mathrm{dist}$;
(14) $\widetilde{d}^* \leftarrow d_k$;
(15) end if
(16) end for
(17) **return** $\widetilde{d}^*$;

Algorithm 1: Best driver selection (BDS).

SP or the proxy. The proxy does not even know the user's identity, because it is replaced by pseudonyms. Note that the zone size should be well-defined, because unreasonable zone size cannot protect users' location privacy against known background knowledge attack. For instance, if there is only one semantic location in a zone, then the SP will learn riders in the zone are likely to be picked up at this semantic location. From the above analysis, it is clear that it is difficult for the SP and the proxy to profile riders' and drivers' activities.

(iii) *Against A3*. A driver can learn a rider's ride information if and only if he is matched to the rider; otherwise, the driver harvests nothing. Likewise, a rider only receives the information of his matched driver. Obviously, unauthorized ride harvesting attacks are hard to be launched.

6. Experimental Evaluation

Our experiments are performed on the real road network of California (http://www.cs.utah.edu/lifeifei/SpatialDataset.htm) (Figure 3(a)), which contains 21048 vertices and 21693 edges. We generate riders and drivers on the edges in a random fashion (Figure 3(b)) and set the bit length of the upper bound of road distance to 16, i.e., $\ell = 16$. We implement a proof-of-concept pRMatch in C++, by relying on the Paillier cryptosystem library (http://acsc.cs.utexas.edu/libpaillier). All our experiments are conducted and executed on a PC running Ubuntu 18.04 LTS, with an Intel i7 processor at 3.4 GHz and 16 GB RAM.

pRMatch is compared with the state-of-the-art privacy-preserving schemes ORide [13] and pRide [17] under the same road network. ORide utilizes the Euclidean distance metric for ride matching, and pRide is the first privacy-preserving ride matching scheme using road distance metric. We build ORide on FV scheme [11] inspired from NFLlib (github.com/quarkslab/NFLlib) and FV-NFLlib (github.com/CryptoExperts/FV-NFLlib) and implement pRide on the same Paillier cryptosystem library and Obliv-C [19]. For pRide, the length of a wire in Yao's garbled circuits [9] is set to 80 bits, and the security parameter for blindness is set to 32 bits to achieve statistical security roughly 2^{-15}.

We evaluate the accuracy and the efficiency of pRMatch by varying the dimension of the embedded road network or driver scale, where the dimension ω is the element number of a coordinate in the embedded road network, as defined in equation (8), and driver scale $|D|$ is the total number of available drivers on road.

Accuracy: we use the matching success rate (MSR) to evaluate the accuracy. A matching is successful if the matched driver is the actual driver with the minimum road distance. We define $\mathrm{MSR} = N_{\mathrm{suc}}/N_{\mathrm{all}}$, where N_{all} is the total number of matchings and N_{suc} is the number of success matchings.

Figure 4(a) depicts the MSRs of pRMatch, pRide, and ORide by varying the dimension ω from 4 to 32. We can see the MSRs of pRMatch and pRide raise steadily as the dimension of the embedded road network increases, which is more than 95% if the dimension is higher than 24. That is because higher dimension means higher accuracy of shortest road distance computation. In contrast, Euclidean distance computation in ORide is irrelevant to the dimension, thus the MSR of ORide always stays around 85%. When the dimension is higher than 8, pRMatch and pRide outperform ORide in the terms of accuracy. This is of no surprise because both pRMatch and pRide choose road distance to make ride matching, which is more accurate than Euclidean distance. Figure 4(b) depicts

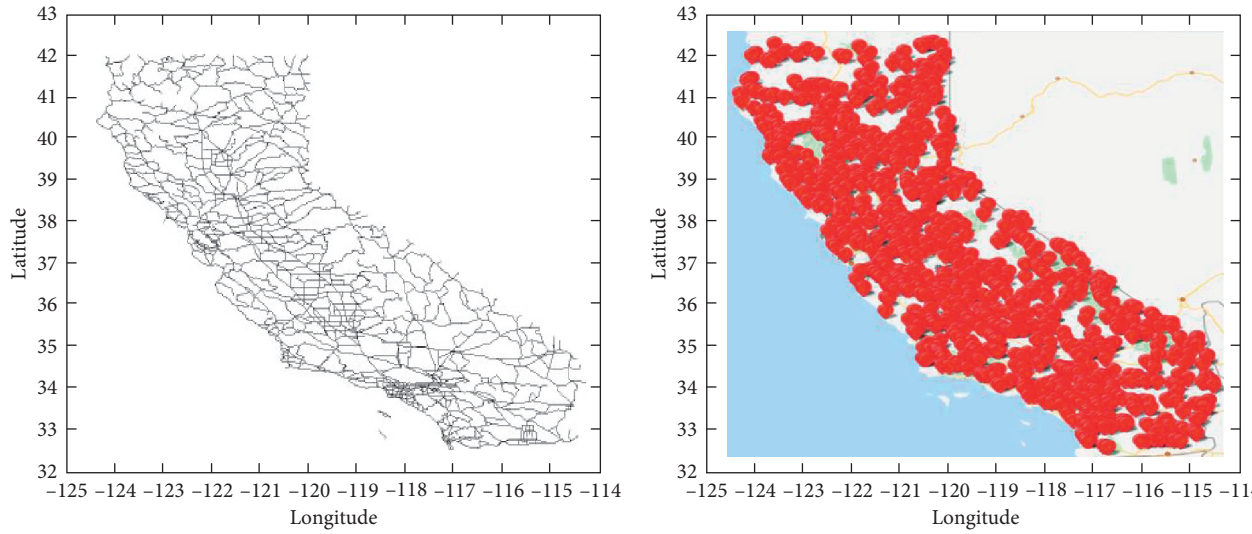

Figure 3: Dataset for experiments. (a) The road network of California (b) The user generation over the road network.

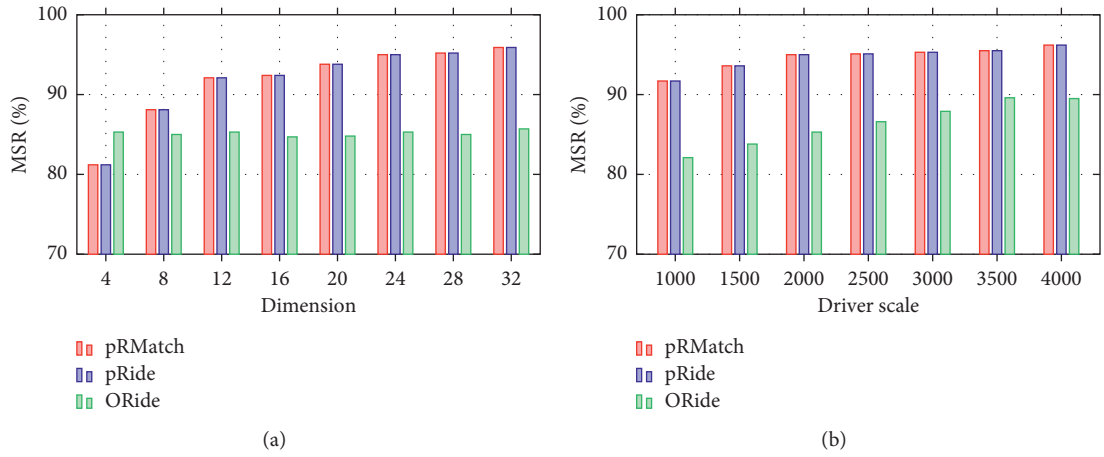

Figure 4: The accuracy of pRMatch, ORide, and pRide, when the number of zones is 25. (a) The MSR under a different dimension of the embedded road network, when $|D| = 2000$. (b) The MSR under a different driver scale, when $\omega = 24$.

the MSRs of pRMatch, pRide, and ORide by varying the driver scale from 1000 to 4000. As shown in Figure 4(b), the MSRs of pRMatch and pRide are always high under any driver scale. This demonstrates the accuracy of pRMatch is not affected by big-scale drivers. The MSR of ORide gradually rises as the driver scale increases, because larger driver scale indicates there may exist closer drivers located around riders. However, the MSR of ORide is still less than 90%. Above experimental results demonstrate that pRMatch can reach a higher accuracy due to the choice of road distance.

Efficiency: we use average online comp. cost and comm. cost for per ride matching to evaluate the efficiency.

Figures 5(a)–5(d) depict the comp. cost of pRMatch and pRide for per-matching varying the dimension of the embedded road network or driver scale. At the user (including riders and drivers) side, a user needs to encrypt its location coordinate in the embedded road network. As shown in Figure 5(a), the comp. cost of the rider (and the driver) in pRMatch raises with the dimension increases. The reason is that higher dimension requires more encryption operations over a location coordinate. Figure 5(b) shows the comp. cost of the rider (and the driver) stays steady under any driver scale. In addition, pRMatch costs more execution time at the user side, because it cannot pack encrypted coordinate into one single ciphertext. Nevertheless, pRMatch is still practical at the user side for resource-constrained devices. For instance, a user requires roughly 0.1 second to encrypt a 24-dimensional location coordinate. As shown in Figure 5(c) and 5(d), the comp. cost at the

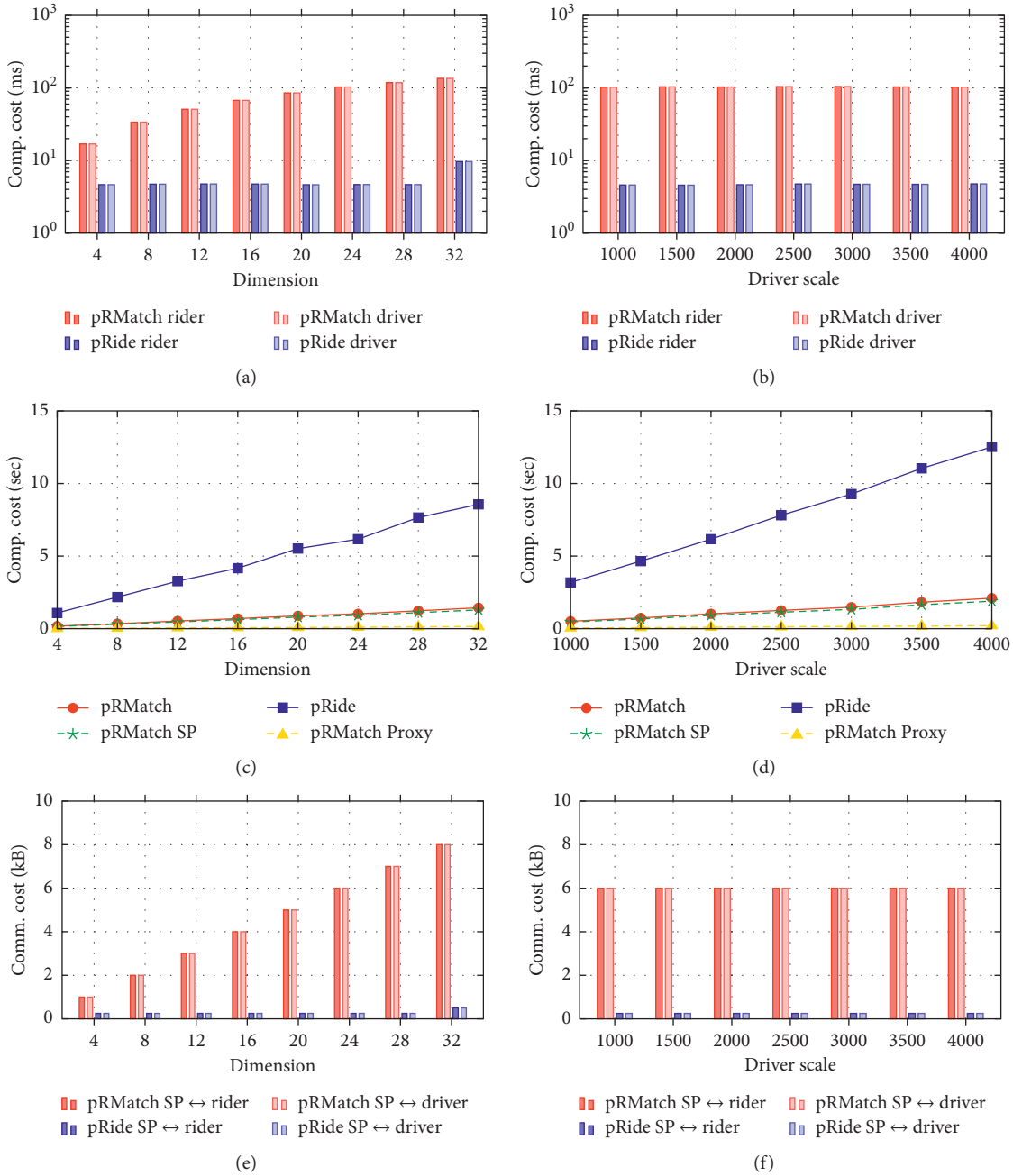

Figure 5: Continued.

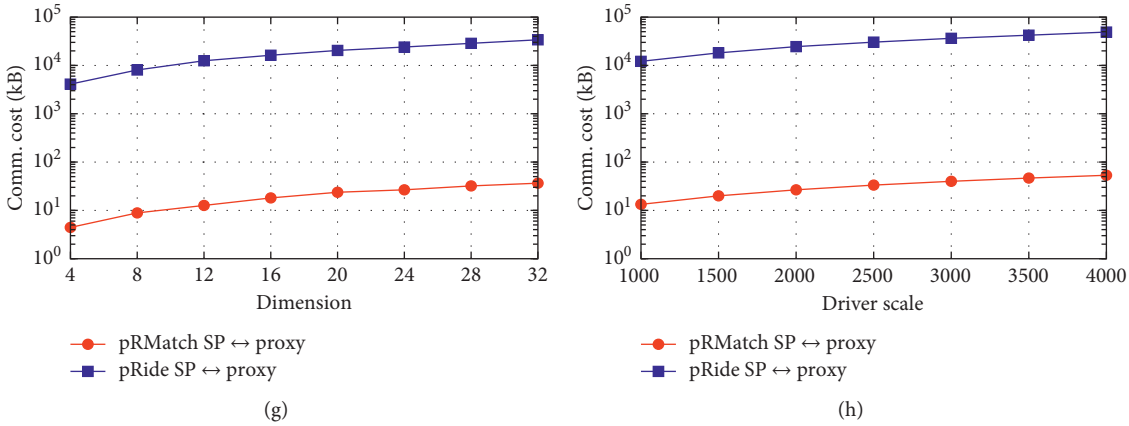

Figure 5: The efficiency of pRMatch and pRide, when the number of zones is 25. (a) The comp. cost of a user (including the rider and candicate drivers) under different dimension of the embedded road network, when $|D| = 2000$. (b) The comp. cost of a user (including the rider and candicate drivers) under a different driver scale, when $\omega = 24$. (c) The comp. cost of the servers under a different dimension of the embedded road network, when $|D| = 2000$. (d) The comp. cost of the servers under a different driver scale, when $\omega = 24$. (e) The comm. cost between the SP and a user (including the rider and candicate drivers) under a different dimension of the embedded road network, when $|D| = 2000$. (f) The comm. cost between the SP and a user (including the rider and candicate drivers) under a different driver scale, when $\omega = 24$. (g) The comm. cost between the servers under a different dimension of the embedded road network, when $|D| = 2000$. (h) The comm. cost of between servers under a different driver scale, when $\omega = 24$.

server (including the SP and the proxy) side in pRMatch and pRide increases almost linearly as the dimension or the driver scale increases. It is obvious that the comp. cost at the server side in pRMatch is much lower than that in pRide. For instance, pRMatch requires about 1 second for per-matching when the dimension is 24 and the driver scale is 2000, but pRide requires more than 6 seconds. Moreover, we can see that the SP undertakes most comp. cost at the server side, and the proxy always has low comp. cost with big-scale drivers (less than 0.15 second).

Figures 5(e)–5(h) depict the comm. cost of pRMatch and pRide for per-matching by varying the dimension of the embedded network or driver scale. As shown in Figures 5(e) and 5(f), pRMatch requires more comm. cost between the SP and the rider (and the driver) as the dimension increases, while the comm. cost is not affected by the driver scale. We can also see that the comm. cost between the SP and the rider (and the driver) in pRMatch is higher than that in pRide, but it is quite acceptable for resource-constrained devices. As shown in Figures 5(g) and 5(h), pRide requires very heavy comm. cost between the two servers, and it needs even more as the dimension or the driver scale increases. In contrast, the comm. cost between the SP and the proxy in pRMatch slowly grows with the increase of the dimension or driver scale, and it always stays low, about two orders of magnitude less than that in pRide. For instance, pRMatch requires less than 27 kBs comm. cost at the server side when the dimension is 24 and the driver scale is 2000, but pRide requires roughly 24 MBs.

Compared with ORide, pRMatch reaches higher accuracy. Compared with pRide, pRMatch achieves higher efficiency at the server side; especially, it brings an order-of-magnitude improvement in comm. cost. pRMatch requires more cost at the user side than pRide, but the cost is quite acceptable for resource-constrained devices.

7. Related Work

The rapid adoption of ORH services poses significant challenges for users' privacy disclosure, as there are a few studies that have emphasized the importance of privacy-preserving solutions in ORH services [2]. PrivateRide [4] is the first system to enhance location privacy for riders in ORH services. It uses spatial cloaking regions to replace their actual locations. But obfuscate locations, instead of the actual locations, affect the accuracy of ride matching; the matched driver may not be the optimal one and has to drive extra distance to pick up the rider; meanwhile, the rider may wait extra time to get the ride. Moreover, PrivateRide only provides limited privacy guarantees for drivers and offers less accountability features. Later, ORide [13] is proposed to address these limitations, which provides stronger privacy and accountability guarantees for both riders and drivers based SHE and anonymous credentials. In addition, ORide applies optimizations for ciphertext packing and transformed processing, hence enabling a notable boost in performance and a reduction in overhead. TRACE [14] supports privacy-preserving spatial query based on random masking technique and point-in-polygon strategy. CoRide [20] is a blockchain-assisted system model for several SPs to establish private collaborative-rides. To achieve efficiency, these schemes utilize the Euclidean distance metric for ride matching, because it is easy to compute over encrypted data. But Euclidean distance may lead to false hits, because drivers always travel along road network. As evaluated in our experiments, it roughly 15% false hits exist when using

Euclidean distance to make ride matching. pRide [17] uses PHE and Yao's garbled circuits to compute road distance for ride matching, but its comp. cost and comm. cost at server side are very heavy. lpRide [21] adopts modified PHE and ciphertext blinding to compute the shortest road distance for ride matching without two noncollusion servers assumption. For both accuracy and efficiency, our pRMatch uses road distance as the travel cost metric to make ride matching over encrypted data without any heavy cost.

Also relevant are the works in privacy-enhanced ridesharing services. PrivatePool [22] is a privacy-preserving protocol allowing users to check their feasible ride matches through PSI and SHE. PRIS [23] is a privacy-preserving ride matching scheme for selecting feasible ridesharing partners based on PHE and bilinear pairing. SRide [24] is a privacy-preserving ridesharing matching protocol based on users' spatial and temporal information, which employs SHE and SMC to compute feasible matches. Most of the previous studies have one limit that hampers their reuse in ORH service. That is, they require drivers offering ridesharing to riders along the route they had already planned to travel. It is impractical for ORH services, where riders can hail a driver to go wherever they want. To tackle this, PSRide [25] is proposed to make privacy-preserving ridesharing matching for ORH systems.

There is a vast literature on protecting location privacy against service providers in LBSs; we summarize them into four fundamentally different ways:

Location-Based k-Anonymity. It requires that the service provider cannot distinguish the request issuer from $k-1$ other users such that the probability of identifying him is $1/k$. To achieve this, one way is to enlarge the issuer's location to a cloaking region including $k-1$ other users by using a hierarchical spatial partitioning [26, 27] or space-filling curves [28]. Another way is to generate $k-1$ dummies (fake locations) together with the actual request to perform k requests to the service provider [29, 30]. Moreover, location l-diversity [31, 32] is also introduced to enhance k-anonymity. The main drawback of k-anonymity-based solutions is the difficulty of providing provable privacy which guarantees to be independent of background knowledge.

Location Obfuscation. It requires blurring or perturbing the location information contained in requests. As reviewed in [33], obfuscation techniques can be divided into three groups. Request enlargement techniques [34, 35] create obfuscation areas for requests to hide users' locations. Dummy-based techniques [36, 37] generate nonsensitive locations as the dummies without considering other potential users (unlike k-anonymity). Coordinate transformation techniques [38, 39] change the complete coordinate reference system using geometric transformations to confuse adversaries. Generally, obfuscation cannot provide provable privacy guarantees without background knowledge, and may impact the quality of service (QoS) of LBSs. Thus, background knowledge is necessary for location obfuscation, e.g., PPCS [40] generates dummy locations based on entropy while considering semantic location information that might be used by attackers.

Differential Privacy. Differential privacy [41] is proposed in statistical databases, which is independent of background knowledge. It can also be used for location privacy in LBSs, such as (D, ϵ)-location privacy [42] and ϵ-geo-indistinguishability [43]. The intuitive idea behind differential privacy is the following: the ratio between the probabilities of obtaining a certain output from two locations relies on their distance, that is, the privacy level increases with their distance. To achieve this, a fake location is probabilistically determined to replace the actual location in requests by running randomization functions. Differential privacy has to make a trade-off between privacy and QoS. Several studies focus on optimal trade-offs [44, 45], but it still has no clear answer.

Encryption-Based Solution. Many encryption-based solutions have been discussed above. Besides, cryptographic primitives have been explored in other LBSs, such as location-based kNN search [46, 47] and friend-finder services [48]. Encryption-based solutions strive for stronger, provable privacy guarantees. The essential challenge is to allow for efficient responding to the requests, despite the computational complexity of the cryptographic primitives.

8. Conclusion and Future Work

In this paper, we focus on the issue of the preservation of location privacy of both riders and drivers during ride matching in ORH services and proposed a privacy-preserving ride matching scheme, namely, pRMatch. It enables an ORH service provider to find the closest available driver for an incoming rider over a city-scale road network, without leaking users' location privacy to the ORH service provider and other unauthorized users. In pRMatch, we proposed an efficient shortest road distance computation approach over encrypted data by using road network embedding and PHE and further designed a secure comparison protocol over encrypted data by using ciphertexts packing and shuffling. We also implement pRMatch and perform extensive experiments to validate its accuracy and efficiency. Experimental results demonstrate that pRMatch achieves about 95% accuracy and guarantees efficiency.

For future work, we will consider the safety issues during ride execution in privacy-enhanced ORH services. When the urgent safety threat happens, e.g., hijacks, a privacy-preserving scheme cannot break the safety protection.

Data Availability

The road network data of California used to support the findings of this study have been deposited in http://www.cs.utah.edu/lifeifei/SpatialDataset.htm.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the Natural Science Foundation of China (Grant nos. 61601146 and 61732022) and National Key R&D Program of China (Grant no. 2016QY05X1000).

References

[1] R. R. Clewlow and G. S. Mishra, "Disruptive transportation: the adoption, utilization, and impacts of ride-hailing in the United States," Research Report–UCD-ITS-RR-17, University of California, Oakland, CA, USA, 2017.

[2] Q. Zhao, C. Zuo, G. Pellegrino, and L. Zhiqiang, "Geo-locating drivers: a study of sensitive data leakage in ride-hailing services," in *Proceedings of the Annual Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2019.

[3] V. Chang, Y. Shi, and X. Li, "Contemporary issues in the ethics of data analytics in ride-hailing service," *International Journal of Strategic Engineering*, vol. 2, no. 2, pp. 44–57, 2019.

[4] A. Pham, I. Dacosta, B. Jacot-Guillarmod et al., "PrivateRide: a privacy-enhanced ride-hailing service," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 2, pp. 38–56, 2017.

[5] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving *k*-anonymity in privacy-aware location-based services," in *Proceedings of the INFOCOM, 2014 Proceedings IEEE*, IEEE, Piscataway, NJ, USA, pp. 754–762, May 2014.

[6] B. Palanisamy and L. Liu, "Attack-resilient mix-zones over road networks: architecture and algorithms," *IEEE Transactions on Mobile Computing*, vol. 14, no. 3, pp. 495–508, 2015.

[7] W. Tong, J. Hua, and S. Zhong, "A jointly differentially private scheduling protocol for ridesharing services," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 10, pp. 2444–2456, 2017.

[8] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, ""Private information retrieval," in *Proceedings of the 36th Annual Symposium on Foundations of computer science*, pp. 41–50, Milwaukee, WI, USA, October 1995.

[9] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *Proceedings of the USENIX Security Symposium*, vol. 201, pp. 331–335, San Francisco, CA, USA, August 2011.

[10] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Interlaken, Switzerland, pp. 1–19, 2004.

[11] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *IACR Cryptology ePrint Archive*, vol. 144, 2012.

[12] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques*, Springer, Prague, Czech Republic, pp. 223–238, May 1999.

[13] T. V. A. Pham, I. I. Dacosta Petrocelli, G. F. M. Endignoux et al., "A privacy-preserving yet accountable ride-hailing service," in *Proceedings of the 26th USENIX Security Symposium*, Vancouver, Canada, August 2017.

[14] F. Wang, H. Zhu, X. Liu et al., "Efficient and privacy-preserving dynamic spatial query scheme for ride-hailing services," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 11084–11097, 2018.

[15] X. Meng, S. Kamara, K. Nissim, and G. Kollios, "Grecs: graph encryption for approximate shortest distance queries," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, ACM, Denver, CO, USA, pp. 504–517, October 2015.

[16] Q. Wang, K. Ren, M. Du, Q. Li, and A. Mohaisen, "SecGDB: Graph encryption for exact shortest distance queries with efficient updates," in *Proceedings of the International Conference on Financial Cryptography and Data Security*, Springer, Sliema, Malta, pp. 79–97, April 2017.

[17] Y. Luo, X. Jia, S. Fu, and M. Xu, "pRide: privacy-preserving ride matching over road networks for online ride-hailing service," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 7, pp. 1791–1802, 2019.

[18] C. Shahabi, M. R. Kolahdouzan, and M. Sharifzadeh, "A road network embedding technique for *k*-nearest neighbor search in moving object databases," *GeoInformatica*, vol. 7, no. 3, pp. 255–273, 2003.

[19] S. Zahur and D. Evans, "Obliv-c: a lightweight compiler for data-oblivious computation," Report 2015/1153, Cryptology ePrint Archive, UCSD, San Diego, CA, USA, 2015.

[20] M. Li, L. Zhu, and X. Lin, "Coride: a privacy-preserving collaborative-ride hailing service using blockchain-assisted vehicular fog computing," in *Proceedings of the International Conference on Security and Privacy in Communication Systems*, Springer, Orlando, FL, USA, pp. 408–422, October 2019.

[21] H. Yu, J. Shu, X. Jia, H. Zhang, and X. Yu, "lpRide: lightweight and privacy-preserving ride matching over road networks in online ride hailing systems," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 10418–10428, 2019.

[22] P. Hallgren, C. Orlandi, and A. Sabelfeld, "PrivatePool: privacy-preserving ridesharing," in *Procedings of the 2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, IEEE, Santa Barbara, AL, USA, pp. 276–291, August 2017.

[23] Y. He, J. Ni, X. Wang, B. Niu, F. Li, and X. S. Shen, "Privacy-preserving partner selection for ride-sharing services," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 5994–6005, 2018.

[24] U. M. Avodji, K. Huguenin, M.-J. Huguet, and M.-O. Killijian, "SRide: a privacy-preserving ridesharing system," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ACM, Stockholm, Sweden, pp. 40–50, June 2018.

[25] H. Yu, X. Jia, H. Zhang, X. Yu, and J. Shu, "PSRide: privacy-preserving shared ride matching for online ride hailing systems," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[26] B. Gedik and L. Liu, "Location privacy in mobile systems: a personalized anonymization model," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, IEEE, Columbus, OH, USA, pp. 620–629, June 2005.

[27] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd International Conference on Very Large Data Bases*, pp. 763–774, Seoul, Republic of Korea, Sepetember 2006.

[28] G. Ghinita, P. Kalnis, and S. Skiadopoulos, "Prive: anonymous location-based queries in distributed mobile systems,," in *Proceedings of the 16th International Conference on World Wide Web*, ACM, Banff, Canada, pp. 371–380, May 2007.

[29] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Proceedings of the 21st International Conference on Data*

Engineering Workshops (ICDEW'05), IEEE, Washington, DC, USA, 2005.

[30] P. Shankar, V. Ganapathy, and L. Iftode, "Privately querying location-based services with sybilquery," in *Proceedings of the 11th International Conference on Ubiquitous Computing*, ACM, Orlando, FL, USA, pp. 31–40, September 2009.

[31] B. Bamba, L. Liu, P. Pesti, and T. Wang, "Supporting anonymous location queries in mobile environments with privacygrid," in *Proceedings of the 17th International Conference on World Wide Web*, ACM, Beijing, China, pp. 237–246, April 2008.

[32] C. Bettini, S. Jajodia, and L. Pareschi, "Anonymity and diversity in lbs: a preliminary investigation," in *Proceedings of the Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, IEEE, White Plains, NY, USA, pp. 577–580, March 2007.

[33] C. S. Jensen, H. Lu, and M. L. Yiu, "Location privacy techniques in client-server architectures," in *Privacy in Location-Based Applications*, pp. 31–58, Springer, Berlin, Germany, 2009.

[34] C. A. Ardagna, M. Cremonini, S. D. C. di Vimercati, and P. Samarati, "An obfuscation-based approach for protecting location privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. 8, no. 1, pp. 13–27, 2011.

[35] M. L. Damiani, E. Bertino, C. Silvestri et al., "The probe framework for the personalized cloaking of private locations," *Trans-Atlantic Data Privacy*, vol. 3, no. 2, pp. 123–148, 2010.

[36] M. Duckham and L. Kulik, "A formal model of obfuscation and negotiation for location privacy," in *Proceedings of the International Conference on Pervasive Computing*, Springer, Kauai Island, HI, USA, pp. 152–170, March 2005.

[37] M. L. Yiu, C. S. Jensen, X. Huang, and H. Lu, "Spacetwist: managing the trade-offs among location privacy, query performance, and query accuracy in mobile services," in *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, IEEE, Cancun, Mexico, pp. 366–375, April 2008.

[38] A. Gutscher, "Coordinate transformation-a solution for the privacy problem of location based services?" in *Proceedings of the 20th IEEE International Parallel & Distributed Processing Symposium*, IEEE, Rhodes Island, Greece, April 2006.

[39] S. Mascetti, L. Bertolaja, and C. Bettini, "A practical location privacy attack in proximity services," in *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management*, vol. 1, IEEE, Milan, Italy, pp. 87–96, June 2013.

[40] G. Sun, S. Cai, H. Yu et al., "Location privacy preservation for mobile users in location-based services," *IEEE Access*, vol. 7, pp. 87425–87438, 2019.

[41] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*, pp. 338–340, Springer, Berlin, Germany, 2011.

[42] E. ElSalamouny and S. Gambs, "Differential privacy models for location-based services," *Transactions on Data Privacy*, vol. 9, no. 1, pp. 15–48, 2016.

[43] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, ACM, Berlin, Germany, pp. 901–914, November 2013.

[44] K. Chatzikokolakis, E. Elsalamouny, and C. Palamidessi, "Efficient utility improvement for location privacy," *Proceedings on Privacy Enhancing Technologies*, vol. 2017, no. 4, pp. 308–328, 2017.

[45] R. Shokri, "Privacy games: optimal user-centric data obfuscation," *Proceedings on Privacy Enhancing Technologies*, vol. 2015, no. 2, pp. 299–315, 2015.

[46] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ACM, Vancouver, Canada, pp. 121–132, June 2008.

[47] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan, "Practical approximate k nearest neighbor queries with location and query privacy," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1546–1559, 2016.

[48] S. Mascetti, D. Freni, C. Bettini, X. S. Wang, and S. Jajodia, "Privacy in geo-social networks: proximity notification with untrusted service providers and curious buddies," *The VLDB Journal*, vol. 20, no. 4, pp. 541–566, 2011.