This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



# An Efficient and Privacy-Preserving Route Matching Scheme for Carpooling Services

Qi Xu, Hui Zhu, *Senior Member, IEEE,* Yandong Zheng, Jiaqi Zhao, Rongxing Lu, *Fellow, IEEE,* and Hui Li, *Member, IEEE*

***Abstract*—With the popularity of intelligent terminals and the advances of mobile Internet, carpooling service, which reduces the travel cost of each user by allowing multiple users to share one car, has received considerable attention and make our life more convenient. However, the vigorous development of carpooling services still faces severe challenges in users' location or route privacy. In this paper, we propose an efficient and privacy-preserving route matching scheme called TAROT for carpooling services. With TAROT, users can enjoy high-quality carpooling services while without revealing sensitive location and route information. Specifically, based on a Goldwasser-Micali based Equality Determination algorithm, we design an Accurate Similarity Computation algorithm, which allows users to get accurate carpooling results over ciphertexts. Meanwhile, the Reverse Minhash method is also designed to construct a Dissimilar Route Filter algorithm, which can filter out dissimilar routes in advance and reduce computational costs and communication overheads. Security analysis shows that TAROT can protect users' location privacy. In addition, TAROT is also evaluated with many random maps, and the simulation results demonstrate that TAROT is highly-efficient.**

***Index Terms*—Carpooling service, location based service (LBS), privacy preservation, route matching.**

## I. Introduction

It has been witnessed for the last decade that shared services have great impact on our daily lives and flourished around the world. As one of the typical shared services, carpooling service has become much more popular [1], [2]. A study has showed that by 2026, 13 percent of people will choose carpooling [3]. Meanwhile, more and more professional carpooling applications are extensively applied worldwide, e.g., BlaBlaCar covers 22 countries with 90 million members [4], and Waze has 140 million users worldwide [5]. Compared with traditional taxiing, carpooling is more economical and environmentally friendly. A conceptual architecture of carpooling service is shown in Fig. 1, which shows that users can find suitable carpooling partners to travel together without reducing the quality of the trip to save expenses.

In carpooling services, each user needs to submit a carpooling query to the service provider, which contains sensitive location information such as starting point, destination, travel route, and others. Obviously, once the location privacy is leaked, the life and property of the user would be threatened [6], [7]. For instance, Singaporean ride-hailing giant Grab had just been fined for its fourth data privacy leakage since 2018 [8]. Ride-hailing app Didi has been banned off from app stores because it is found to have severely violated the laws by illegally collecting and using personal information [9]. In addition, the GDPR became enforceable beginning May 25, 2018, which enables individuals to control their personal data [10]. Furthermore, the California Privacy Rights Act of 2020 shall become operative on January 1, 2023, which limits businesses' usage of sensitive personal information, e.g., precise geolocation [11]. Therefore, significant challenges have been brought to the popularization and promotion of carpooling services.

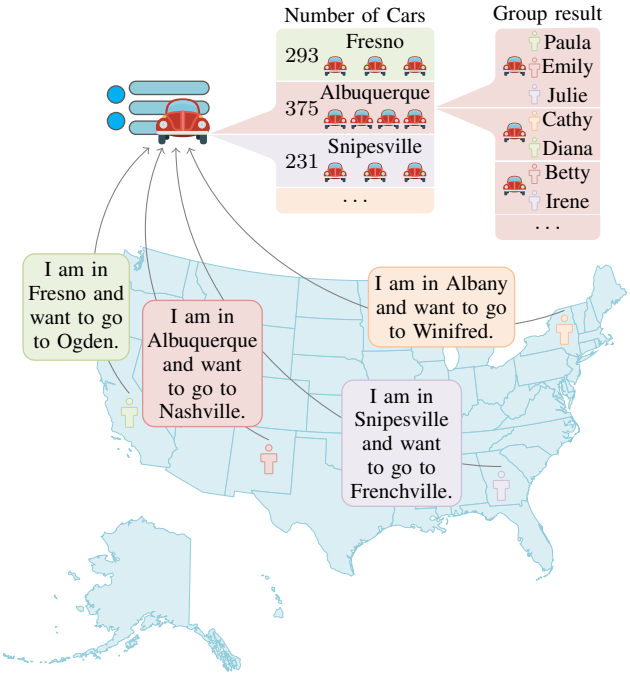



Fig. 1. Conceptual architecture of carpooling service.

To protect users' sensitive location information in location based service (LBS), a variety of privacy-preserving technologies have been proposed, which are respectively designed based on the anonymization strategy and cryptographic primitives. The anonymization strategy based ones include spatial cloaking, location obfuscation, and dummy. Spatial cloaking

This work was supported by National Key R&D Program of China (2021YFB3101300), National Natural Science Foundation of China (61972304 and 61932015), Shaanxi Provincial Key Research and Development Program (2020ZDLGY08-04), and OPPO Research Fund. (Corresponding author: Hui Zhu.)

Qi Xu, Hui Zhu, Jiaqi Zhao, and Hui Li are with the School of Cyber Engineering, Xidian University, Xi'an, Shaanxi, China (e-mail: qixu@stu.xidian.edu.cn, zhuhui@xidian.edu.cn, jq_zhao@stu.xidian.edu.cn, and lihui@mail.xidian.edu.cn). Yandong Zheng and Rongxing Lu are with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: yzheng8@unb.ca and rlu1@unb.ca).

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



technology [12]–[14] constructs a cloaking area where the user is located, and there are at least $k-1$ other users in the cloaking area. It is difficult for the service provider to distinguish the location and identity of the user from the cloaking area. Location obfuscation technology [15]–[19] reduces the accuracy of location by adding noise to protect the privacy of location. Dummy technology [20]–[23] protects location privacy by generating false location data along with the true location data to obfuscate an adversary. However, these technologies based on generalization and obfuscation make it impossible for service providers to obtain the accurate location. Thus, users cannot enjoy accurate LBS. The cryptographic primitives based ones include homomorphic encryption [24]–[30], secure multiparty computing [26], [31], [32], private set intersection [27], [33], and others, which can provide accurate LBS for users. However, cryptography-based techniques usually involve a large number of complex operations. Therefore, they are not applicable for ciphertext computation among a large number of data items. Thus, existing solutions suffer from either accuracy or efficiency issues and are not applicable to implement carpooling services.

Aiming at addressing these issues, in this paper, we propose an efficient and privacy-preserving route matching scheme for carpooling services, named TAROT. Specifically, our contributions are threefold.

- First, TAROT achieves accurate route matching in carpooling services without leaking users' location information. Specifically, we design a Goldwasser-Micali based Equality Determination algorithm (GMEDA), and based on GMEDA, an accurate similarity computation algorithm (ASCA) is proposed to compute the similarity in ciphertext accurately. Therefore, TAROT can protect the privacy of users' sensitive locations.
- Second, TAROT improves the carpooling matching efficiency significantly. Specifically, we design a Reverse Minhash (RM) method and an RM based Jaccard similarity. Based on them, we construct a dissimilar route filter algorithm (DRFA) to efficiently filter dissimilar routes, which can greatly reduce the number of route matching.
- Third, TAROT is effective and efficient in carpooling dataset. We implement TAROT by Python, and deploy it in carpooling dataset with different parameter selection to evaluate its performance. The results show TAROT is effective and efficient in terms of computational costs and communication overheads.

The remainder of this paper is organized as follows. First of all, we formalize the system model and security model in Section II. Second, some preliminaries used in our work are introduced in Section III. Then, we give some building blocks and the details of TAROT in Section IV and Section V, respectively. After that, security analysis is described in Section VI. Next, parameter selection is described in Section VII. Later, experimental evaluation is described in Section VIII. Finally, we review some related works in Section IX and draw our conclusion in Section X.

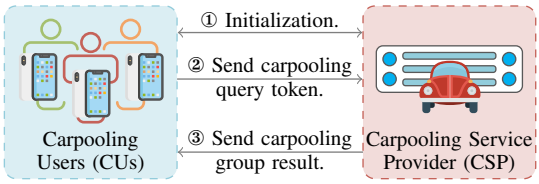



Fig. 2. System model under consideration.

## II. System Model and Security Model

In this section, we formalize our system model and security model.

### A. System Model

In the system model, we mainly focus on how to provide an accurate and efficient privacy-preserving carpooling service for carpooling users. As shown in Fig. 2, our system model consists of two kinds of entities: carpooling users (CUs) and carpooling service provider (CSP).

- *CUs:* Each CU is equipped with an intelligent terminal with public parameters from CSP, through which CU can launch carpooling query requests to CSP. The query request is specified by a set of routes from the same starting point to the same destination, which means CU intends to ride from the starting position to the destination. Each route in query request consists of a few location points and the number of location points in a route depends on the starting point and destination. To reduce the cost, he/she prefers to use carpooling services. Meanwhile, considering the privacy of the starting point and destination, CUs demand to send the query request to CSP in an encrypted form.
- *CSP:* CSP is equipped with a workstation and is responsible for initializing the system by generating public parameters. After initialization, it will offer carpooling query services to CUs. In our scheme, we consider that all CUs in the current timestamp simultaneously launch carpooling query requests to CSP. On receiving these query requests in an encrypted form, CSP computes the similarity of the corresponding routes in different query requests on ciphertext by some cryptographic techniques. Then, CSP groups CUs based on these similarities and returns the carpooling group result to CUs. It is worth noting that CSP finishes the carpooling query requests processing with the assistance of CUs.

### B. Security Model

In the security model, we consider that both CSP and CUs are honest-but-curious [34]. Specifically, the CSP will honestly follow our scheme to offer carpooling query services to CUs but may be curious about the plaintext location and route information of CUs. For CUs, each of them also honestly follows our scheme to launch carpooling query requests to the CSP. However, since it assists the CSP to process the query requests and may have a chance to access some ciphertexts related to other CUs' location and route information, they may

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



attempt to obtain the plaintext location and route information of other CUs. In the realistic carpooling system, the location information cannot be protected between CUs who match successfully. Therefore, it is not practical to prevent CSP from colluding with CU to steal other CUs' private location information. Since our target is to protect sensitive data of CUs in carpooling, some attacks such as collusion attack and data pollution attack are currently out of scope of this paper and will be considered in future work.

## III. Preliminaries

In this section, we review some preliminaries used in TAROT, which serve as the basis of our TAROT scheme.

### A. Jaccard Similarity and Route Similarity

In this section, we introduce the definition of Jaccard similarity [35] and route similarity considered in our scheme.

**Definition 1** (Jaccard Similarity)**.** *Let $\mathcal{U}$ be a complete set, and $\mathcal{A}, \mathcal{B}$ be two subsets of $\mathcal{U}$. The Jaccard similarity between $\mathcal{A}$ and $\mathcal{B}$ is defined as the ratio of their intersection size to union size, i.e.,*

$$\phi(\mathcal{A}, \mathcal{B}) = \frac{|\mathcal{A} \cap \mathcal{B}|}{|\mathcal{A} \cup \mathcal{B}|}. \tag{1}$$

**Definition 2** (Route Similarity)**.** *In our scheme, each route is represented by a set with several locations, which also can be regarded as a vector. Meanwhile, the similarity between two routes is measured by Definition 1.*

### B. Goldwasser-Micali (GM) Algorithm

Goldwasser-Micali (GM) algorithm [36] is a probabilistic public key encryption algorithm that encrypts only one bit at a time and has a homomorphic property of exclusive OR operation. Specifically, the GM algorithm is defined as follows.

- $\mathrm{KeyGen}(\kappa)$*:* Given a security parameter $\kappa$, the key generation algorithm first selects two random prime numbers $p, q$ such that $|p| = |q| = \kappa$, and let $n = p \times q$. Then, it chooses an integer $g$ such that $g$ is quadratic nonresidue modulus $n$. Finally, it outputs the public key $pk = (n, g)$, the private key $sk = (p, q)$, and the message space $M = \{0, 1\}$.
- $\mathrm{Enc}(m, pk)$*:* A message $m \in M$ can be encrypted by $pk$ as $c = \mathrm{Enc}(m) = g^m \times r^2 \bmod n$, where $r \in \mathbb{Z}_n^*$ is a random number.
- $\mathrm{Dec}(sk, c)$*:* On input $sk$ and $c$, the decryption algorithm first computes the Jacobi symbol $(\frac{c}{n}) = (\frac{c}{p})(\frac{c}{q})$ to determine whether $c$ is a quadratic residue or not. Then, if $c$ is a quadratic residue, it means that $m = 0$. If $c$ is quadratic nonresidue, it means $m = 1$.

The GM algorithm satisfies the exclusive OR homomorphic property, i.e., $\mathrm{Enc}(m_1) \times \mathrm{Enc}(m_2) \rightarrow \mathrm{Enc}(m_1 \oplus m_2)$.

## IV. Building Blocks

In this section, we propose a GM based Equality Determination algorithm (GMEDA) and a Reverse Minhash (RM) based Jaccard similarity, which will serve as the building blocks of our proposed scheme. Some notations will be used throughout the scheme. These notations and their corresponding definitions are shown in TABLE I.

TABLE I
Common Notations in TAROT

| *Notation* | *Definition* |
|---|---|
| $\iota$ | The total number of location points on the map. |
| $k_i$ | The total number of $\mathrm{CU}_i$'s routes. |
| $P_i$ | The route vector set of $\mathrm{CU}_i$. |
| $p_{i,u}$ | The $u$th route vector in $P_i$. |
| $\phi$ | The Jaccard similarity. |
| $\psi$ | The RM based Jaccard similarity. |
| $\Psi$ | The matrix composed of $\psi$s. |
| $\Lambda$ | The RM function tuple. |
| $\lambda_t$ | The $t$th RM function in $\Lambda$. |
| $\tau$ | The number of RM functions in $\Lambda$. |
| $s_{i,u}$ | The RM vector of $p_{i,u}$. |
| $s_{i,u}^t$ | The $t$th RM value of $s_{i,u}$. |
| $S_i$ | The RM vector tuple of $P_i$. |

### A. GM Based Equality Determination Algorithm (GMEDA)

To determine whether two location points are identical, based on the GM algorithm, we design a privacy-preserving equality determination algorithm named GMEDA. The GMEDA is run between two users $\mathrm{CU}_i$ and $\mathrm{CU}_j$, who respectively hold two integers $A$ and $B$. They run the GMEDA to determine whether $A = B$ or not, as Alg. 1. In the GMEDA, to simplify the description, we use $[\![]\!]$ to represent $\mathrm{Enc}()$. Meanwhile, $[\![]\!]$ is also used to represent the ciphertext of an integer.

In the GMEDA, $\mathrm{CU}_i$ generates public and private keys $(pk_i, sk_i) \leftarrow \mathrm{KeyGen}(\kappa)$ and publishes $pk_i$. It expresses $A$ in binary as $A = (a_1, a_2, \ldots, a_l)$ and computes $[\![A]\!] = ([\![a_1]\!], [\![a_2]\!], \ldots, [\![a_l]\!])$ with $pk_i$. Then, it sends $[\![A]\!]$ to $\mathrm{CU}_j$. On receiving $[\![A]\!]$, $\mathrm{CU}_j$ expresses $B$ in binary as $B = (b_1, b_2, \ldots, b_l)$ and computes $[\![B]\!] = ([\![b_1]\!], [\![b_2]\!], \ldots, [\![b_l]\!])$ with $pk_i$. After that, it computes the Hadamard product of $[\![A]\!]$ and $[\![B]\!]$ and gets $[\![A]\!] \circ [\![B]\!] = ([\![a_1]\!] \times [\![b_1]\!], [\![a_2]\!] \times [\![b_2]\!], \ldots, [\![a_l]\!] \times [\![b_l]\!])$. Next, $\mathrm{CU}_j$ gets $[\![A]\!][\![B]\!]$ by randomly permutating the elements in $[\![A]\!] \circ [\![B]\!]$. This is the key to achieving privacy preservation. Later, $\mathrm{CU}_j$ sends $[\![A]\!][\![B]\!]$ to $\mathrm{CU}_i$. Finally, $\mathrm{CU}_i$ decrypts each element of $[\![A]\!][\![B]\!]$ with $sk_i$ in turn. Once a decryption result is 1, $\mathrm{CU}_i$ gets $A \neq B$. Otherwise, $\mathrm{CU}_i$ gets $A = B$. Finally, $\mathrm{CU}_i$ returns the result to $\mathrm{CU}_j$.

### B. RM Based Jaccard Similarity

In this section, we introduce an RM based Jaccard similarity $\psi$ to measure the similarity between two routes, which is larger than $\phi$ and will be used for dissimilar route filter in TAROT. Before introducing RM, we briefly describe MinHash [37]. MinHash can be directly used to compute route similarity efficiently. The hash values of routes obtained by MinHash are all points on the routes, which will lead to the data leakage of sensitive locations of CUs. Therefore, we design RM based on MinHash to solve the problem of privacy leakage in the process of efficient route similarity computation. In the following, we first introduce RM, and based on it, define RM based Jaccard similarity.

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



**procedure** ENC ▷ $CU_i$
  $(pk_i, sk_i) \leftarrow \mathrm{KeyGen}(\kappa)$ ▷ *public and private key*
  $A \leftarrow (a_1, a_2, \ldots, a_l)$ ▷ *express $A$ in binary*
  $[\![A]\!] \leftarrow ([\![a_1]\!], [\![a_2]\!], \ldots, [\![a_l]\!])$ ▷ *encrypt $A$ with $pk_i$*
  **send** $[\![A]\!]$ to $\mathrm{CU}_j$
**procedure** MULTIENC ▷ $CU_j$
  $B \leftarrow (b_1, b_2, \ldots, b_l)$ ▷ *express $B$ in binary*
  $[\![B]\!] \leftarrow ([\![b_1]\!], [\![b_2]\!], \ldots, [\![b_l]\!])$ ▷ *encrypt $B$ with $pk_i$*
  $[\![A]\!] \circ [\![B]\!] \leftarrow ([\![a_1]\!] \times [\![b_1]\!], [\![a_2]\!] \times [\![b_2]\!], \ldots, [\![a_l]\!] \times [\![b_l]\!])$
  $[\![A]\!][\![B]\!] \leftarrow$ randomly permutate $[\![A]\!] \circ [\![B]\!]$
  **send** $[\![A]\!][\![B]\!]$ to $\mathrm{CU}_i$
**procedure** GETEQUALITY ▷ $CU_i$
  **for** $e \in [\![A]\!][\![B]\!]$ **do**
    **if** $\mathrm{Dec}(sk_i, e) = 1$ **then**
      **send** $A \neq B$ to $\mathrm{CU}_j$
  **if** all $\mathrm{Dec}(sk_i, e) = 0$ **then**
    **send** $A = B$ to $\mathrm{CU}_j$

Alg. 1. GMEDA.

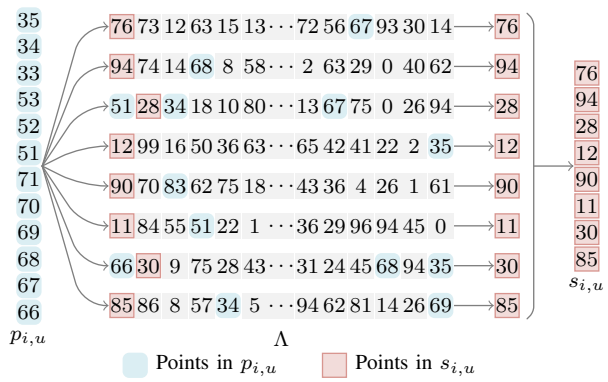



Fig. 3. Example of RM vector generation.

*1) RM:* Suppose that there are $\iota$ location points on the map, and each location has a serial number $\zeta_i$. Let $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_\tau)$ denote $\tau$ different RM functions, and each $\lambda_t$ is a random permutation of $(\zeta_1, \zeta_2, \ldots, \zeta_\iota)$. Let $p_{i,u}$ and $p_{j,v}$ denote two routes of $\mathrm{CU}_i$ and $\mathrm{CU}_j$ consisting of the location serial numbers.

Then, $\mathrm{CU}_i$ can represent the route $p_{i,u}$ to an RM vector $s_{i,u}$ according to $\Lambda$. The phase of generating one RM vector is clearly illustrated in the example shown in Fig. 3. Assume $\iota$ is 100, $\tau$ is 8, and $\zeta$s are represented by consecutive integers 0 to 99. $\Lambda$ is randomly generated. $\mathrm{CU}_i$ labels all numbers in $\Lambda$ belonging to $p_{i,u}$ and finds the first number among the remaining unlabeled numbers in each $\lambda$ as the RM value. These RM values will form RM vector $s_{i,u} = [s^1_{i,u}, s^2_{i,u}, \ldots, s^\tau_{i,u}]$. Similarly, $\mathrm{CU}_j$ can represent the route $p_{j,v}$ to an RM vector $s_{j,v}$ according to $\Lambda$.

*2) RM Based Jaccard Similarity:* Based on the RM vectors, we can formally define RM based Jaccard similarity. Then, we prove that the new similarity is larger than Jaccard similarity.

**Definition 3** (RM Based Jaccard Similarity)**.** *Let $p_{i,u}$ and $p_{j,v}$ be two route vectors. $CU_i$ first generates the RM $s^t_{i,u}$ according to $\lambda_t$, and $CU_j$ generates $s^t_{j,v}$. The RM based Jaccard similarity of $p_{i,u}$ and $p_{j,v}$ is defined as the probability of $s^t_{i,u} = s^t_{j,v}$, i.e.,*

$$\psi(p_{i,u}, p_{j,v}) = \Pr[s^t_{i,u} = s^t_{j,v}]. \tag{2}$$

*Let $s_{i,u} = [s^1_{i,u}, s^2_{i,u}, \ldots, s^\tau_{i,u}]$ and $s_{j,v} = [s^1_{j,v}, s^2_{j,v}, \ldots, s^\tau_{j,v}]$ be the RM vectors of $p_{i,u}$ and $p_{j,v}$. According to Chebyshev's inequality, when $\tau \to \infty$, the frequency and probability are equal almost surely, i.e.,*

$$\lim_{\tau\to\infty} \frac{1}{\tau} \sum_{t=1}^{\tau} \left| s^t_{i,u} - s^t_{j,v} \right|^0 \overset{a.s.}{=\!=} \Pr[s^t_{i,u} = s^t_{j,v}].$$

*Therefore, with the appropriate $\tau$,*

$$\psi(p_{i,u}, p_{j,v}) = \frac{1}{\tau} \sum_{t=1}^{\tau} \left| s^t_{i,u} - s^t_{j,v} \right|^0.$$

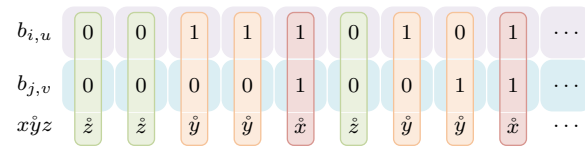



Fig. 4. Concrete example of the relationship between points in two routes.

**Theorem 1.** *$\psi$ is greater than $\phi$.*

*Proof.* For the convenience of description, we only consider $\lambda_t$ and two route vectors $p_{i,u}$ and $p_{j,v}$, which belong to $\mathrm{CU}_i$ and $\mathrm{CU}_j$. According to $p_{i,u}$ and $\lambda_t$, a new vector $b_{i,u}$ is constructed. The length of $b_{i,u}$ is the same as that of $\lambda_t$, both of which are $\iota$. Each element in $b_{i,u}$ is 0 or 1. When an element of $\lambda_t$ exists in $p_{i,u}$, the element of $b_{i,u}$ in the same position is 1. Otherwise, it is 0. Intuitively, $b_{i,u}$ reorders $p_{i,u}$ according to the relative positions of elements in $\lambda_t$. According to $p_{j,v}$ and $\lambda_t$, the vector $b_{j,v}$ can be obtained.

There may be three cases of elements at the same position of $b_{i,u}$ and $b_{j,v}$. Both are 1, one is 0, one is 1, and both are 0. These three cases are marked as $\mathring{x}$, $\mathring{y}$, and $\mathring{z}$. In addition, the number of occurrences of these three situations is denoted as $x$, $y$, and $z$. For the convenience of description, an example of actual values as shown in Fig. 4 is given. The corresponding location point where the first 0 element in $b_{i,u}$ or $b_{j,v}$ is located is $s^t_{i,u}$ or $s^t_{j,v}$. Therefore, event $s^t_{i,u} = s^t_{j,v}$ can also be expressed as, in vector $x\mathring{y}z$, there is no $\mathring{y}$ before $\mathring{z}$. It can be easily obtained that the probability of $s^t_{i,u} = s^t_{j,v}$ is $\Pr[s^t_{i,u} = s^t_{j,v}] = {}^{z}/_{z+y}$. According to (2), $\psi = {}^{z}/_{z+y}$. The size of the union is $x + y$, and the size of the intersection is $x$. According to (1), $\phi = {}^{x}/_{x+y}$. For any route, the elements it owns are only a small part of all the location points on the map, so $z$ must be greater than $x$. So far, we have $\phi = {}^{x}/_{x+y}$, $\psi = {}^{z}/_{z+y}$, and $z > x$. It is easy to get that

$$\psi - \phi = \frac{z}{z+y} - \frac{x}{x+y} = \frac{(z-x)y}{(z+y)(x+y)} > 0.$$

Therefore, $\psi$ is greater than $\phi$. □

## V. OUR PROPOSED SCHEME

Based on GMEDA and RM based Jaccard similarity, in this section, we propose an efficient and privacy-preserving

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



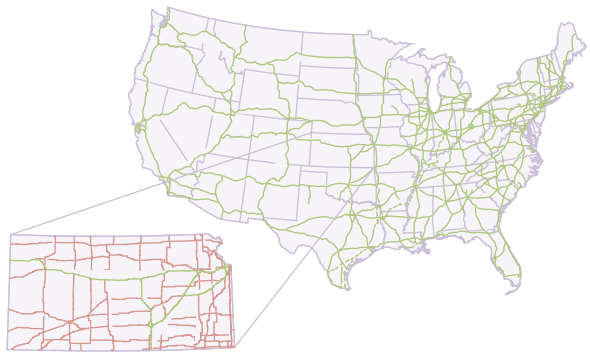


Fig. 5. Example of hierarchical map.

route matching scheme, i.e., TAROT, including five processes: initialization, query token generation, dissimilar route filter, accurate similarity computation, and CU grouping. In addition, TAROT can easily be extended to multiple routes.

### A. Initialization

In the initialization phase, CSP and CUs generate public and private keys. In TAROT, all maps are hierarchical as shown in Fig. 5. When the routes are cross-city, the map is a national map, and locations on the map are generally cities or towns. If it is a trip within a city, then the map is a city map, and locations on the map are generally vital such as intersections or business centers. A hierarchical map will keep the number of location points on the route within a reasonable range. In short, CSP has a map with $\iota$ locations. Each location has a serial number $\zeta_i$. All $\iota$ location points with serial numbers are stored in a list $\mathcal{L}$. The map distances between all location points on the map are stored in a matrix $\mathcal{D}$. Each element $d_{i,j} \in \mathcal{D}$ represents the map distance between $\zeta_i$ and $\zeta_j$, where the map distance is defined as the actual map distance between two location points rather than a straight line distance. After that, CSP generates $\tau$ different RM functions $\Lambda = (\lambda_1, \lambda_2, \ldots, \lambda_\tau)$. At last, CSP publishes public parameters $\mathcal{L}$, $\mathcal{D}$, and $\Lambda$. Regarding CUs, each $\text{CU}_i$ first chooses a security parameter $\kappa$ and generates the public key and private key as $(pk_i, sk_i) \leftarrow \text{KeyGen}(\kappa)$. Then, it publishes the public key $pk_i$.

### B. Query Token Generation

For general CUs, given a starting point and a destination, multiple routes can be generated by common K-Shortest Path (KSP) algorithms, such as Yen's algorithm [38]. In addition, some CUs with personalized needs will consider the requirements such as road conditions and stay points when generating routes. At this time, both KSP algorithm and distance matrix [39] can help CUs generate routes in this process. In other words, $\text{CU}_i$ generates the corresponding route vector set $P_i = \{p_{i,1}, p_{i,2}, \ldots, p_{i,k_i}\}$ locally. $P_i$ includes $k_i$ shortest routes from starting point to destination, where each route $p_{i,u}$ is a vector. For each route $p_{i,u}$ in $P_i$, $\text{CU}_i$ first generates the RM vector $s_{i,u}$ according to $\Lambda$ for $u \in [1, k_i]$ and sends the RM vector tuple $S_i = (s_{i,1}, s_{i,2}, \ldots, s_{i,k_i})$ as the carpooling query token to CSP.

**procedure** COMPSIM ▷ *compute $\psi$s between CUs*
  **for** $i \leftarrow 1, 2, \ldots, n$ **do**
    **for** $j \leftarrow i+1, i+2, \ldots, n$ **do**
      compute $\psi_{i,j}$
**procedure** GENSIMMATRIX ▷ *generate $\Psi$ matrix*
  **for** $i \leftarrow 1, 2, \ldots, n$ **do**
    **for** $j \leftarrow 1, 2, \ldots, n$ **do**
      **if** $i < j$ **then**
        $\Psi[i][j] \leftarrow \psi_{i,j}$
      **else if** $i = j$ **then**
        $\Psi[i][j] \leftarrow 1$
      **else if** $i > j$ **then**
        $\Psi[i][j] \leftarrow \psi_{j,i}$
**procedure** GENCANDCU ▷ *generate candidate CUs*
  **for** $i \leftarrow 1, 2, \ldots, n$ **do**
    $\xi_i \leftarrow$ a few larger $\psi$s in $i$th line of $\Psi$
  **return** candidate CUs tuple

Alg. 2. DRFA.

### C. Dissimilar Route Filter

On receiving all carpooling query tokens, CSP compares the routes between any two CUs using the RM based Jaccard similarity. It further finds dissimilar routes and filters them. The DRFA is shown in Alg. 2. First, CSP computes a similarity matrix $\Psi$ between CUs, where each element is

$$\psi_{i,j} = \max_{u \in [1,k_i], v \in [1,k_j]} \psi(p_{i,u}, p_{j,v}).$$

Then, CSP generates candidate CUs tuple. For each $P_i$, CSP selects a few route vector sets whose $\psi$s are larger than others. As shown in Theorem 1, $\psi$ is greater than $\phi$. CSP filters smaller $\psi$, i.e., smaller $\phi$ is filtered. These $\psi$s' owners are candidate CUs for $\text{CU}_i$ marked as $\xi_i$, i.e., $\phi$s between $\text{CU}_i$ and CUs in $\xi_i$ are very likely to be large. The same operation is performed for each CU, and candidate CUs tuple $\Xi = (\xi_1, \xi_2, \ldots, \xi_n)$ is obtained at the end.

### D. Accurate Similarity Computation

In this process, we accurately compute similarities between each CU and its corresponding candidate CUs. The direction of the two routes is not considered, and it is easy to judge whether the directions are equal by the GMEDA.

To complete accurate similarity computation, we propose an ASCA based on the GMEDA. Specifically, ASCA consists of four steps: 1) $\text{CU}_i$ encrypts points; 2) $\text{CU}_j$ encrypts points; 3) $\text{CU}_i$ gets common points; and 4) $\text{CU}_i$ computes $\phi_{i,j}$. The pseudocode of the ASCA is shown in Alg. 3. In the following, the four steps for the ASCA are described in detail.

*$CU_i$ encrypts points.* Suppose there are $\epsilon_i$ location points in $P_i$. $\text{CU}_i$ generates a set $\Theta_i = \{\theta_i^1, \theta_i^2, \ldots, \theta_i^{\epsilon_i}\}$, which is the set of location points passed by all the route vectors of $\text{CU}_i$. Then, $\text{CU}_i$ performs GMEDA encryption on each element in $\Theta_i$ to generate the corresponding ciphertext set $[\![\Theta_i]\!] = \{[\![\theta_i^1]\!], [\![\theta_i^2]\!], \ldots, [\![\theta_i^{\epsilon_i}]\!]\}$. Finally, $\text{CU}_i$ sends $[\![\Theta_i]\!]$ to $\text{CU}_j$.

*$CU_j$ encrypts points.* $\text{CU}_j$ first generates $\Theta_j$ from $P_j$ and computes $[\![\Theta]\!]_{i,j} = [\![\Theta_i]\!]^T [\![\Theta_j]\!]$. Then, we define a hash

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



**procedure** ENCPOINTS ▷ *$CU_i$ encrypts points*
  $\Theta_i \leftarrow$ all points in $P_i$
  $[\![\Theta_i]\!] \leftarrow$ use GMEDA encrypt $\Theta_i$
  **send** $[\![\Theta_i]\!]$ to $\mathrm{CU}_j$
**procedure** ENCPOINTS ▷ *$CU_j$ encrypts points*
  $\Theta_j \leftarrow$ all points in $P_j$
  $\Gamma_j \leftarrow$ hash points in $\Theta_j$ by hash function $H_j$
  $P'_j \leftarrow$ hash points in $P_j$ by hash function $H_j$
  $[\![\Theta_j]\!] \leftarrow$ encrypt $\Theta_j$ using GMEDA
  $[\![\Theta]\!]_{i,j} \leftarrow [\![\Theta_i]\!]^T [\![\Theta_j]\!]$
  **send** $[\![\Theta]\!]_{i,j}$, $\Gamma_j$, and $P'_j$ to $\mathrm{CU}_i$
**procedure** COMMONPOINTS ▷ *$CU_i$ gets common points*
  $\Gamma^{\cap}_{i,j} \leftarrow$ decrypt $[\![\Theta]\!]_{i,j}$ to get common points
**procedure** GETSIM ▷ *$CU_i$ computes $\phi_{i,j}$*
  $\phi_{i,j} \leftarrow$ compute $\phi$s from $\Gamma^{\cap}_{i,j}$, $P_i$, and $P'_j$
  **send** $\phi_{i,j}$ to CSP

Alg. 3. ASCA.

function $H_j$ that hashes $\theta^v_j \in \Theta_j$ to $\gamma^v_j \in \mathbb{Z}^*_n$. After that, $\mathrm{CU}_j$ hashes $\Theta_j$ according to $H_j$ to obtain $\Gamma_j$ and hashes the location point serial numbers in $P_j$ to obtain $P'_j$. Finally, $\mathrm{CU}_j$ sends $[\![\Theta]\!]_{i,j}$, $\Gamma_j$, and $P'_j$ to $\mathrm{CU}_i$.

*$CU_i$ gets common points.* Firstly, $\mathrm{CU}_i$ uses the GMEDA to decrypt $[\![\Theta]\!]_{i,j}$ to get whether each pair of location numbers are equal. In order to improve the efficiency of decryption, $\mathrm{CU}_i$ decrypts each element in each row/column in turn. When the decrypted results of a certain row/column are equal, continue to decrypt the next row/column value. Finally, $\mathrm{CU}_i$ can learn the points that $\mathrm{CU}_i$ and $\mathrm{CU}_j$ owned in common. In other words, combining $\Gamma_j$, $\mathrm{CU}_i$ gets a set

$$\Gamma^{\cap}_{i,j} = \{\theta^u_i \leftrightarrow \gamma^v_j \mid \theta^u_i = \theta^v_j, u \in [1, \epsilon_i], v \in [1, \epsilon_j]\}$$

that contains some pairs related to the same location points.

*$CU_i$ computes $\phi_{i,j}$.* According to $\Gamma^{\cap}_{i,j}$, $P_i$, and $P'_j$, $\mathrm{CU}_i$ can determine whether the points on each pair of routes are equal. Then, $\mathrm{CU}_i$ can compute the similarity between each two route vectors which are belong to $P_i$ and $P_j$ respectively for all routes. To clearly describe how $\mathrm{CU}_i$ computes $\phi_{i,j}$, we give an example in the following that $\mathrm{CU}_i$ computes a pair of routes, such as $p_{i,u}$ and $p_{j,v}$. In order to compute $\phi(p_{i,u}, p_{j,v})$, $\mathrm{CU}_i$ counts how many elements in $\Gamma^{\cap}_{i,j}$ appear in both $p_{i,u}$ and $p'_{j,v}$. Combined with the length of $p_{i,u}$ and $p'_{j,v}$,

$$\phi(p_{i,u}, p_{j,v}) = \frac{|\Gamma^{\cap}_{i,j} \cap p_{i,u} \cap p'_{j,v}|}{|p_{i,u}| + |p'_{j,v}| - |\Gamma^{\cap}_{i,j} \cap p_{i,u} \cap p'_{j,v}|}$$

can be computed. $\mathrm{CU}_i$ computes the similarity between $\mathrm{CU}_i$ and $\mathrm{CU}_j$, i.e., $\phi_{i,j} = \max_{u \in [1,k_i], v \in [1,k_j]} \phi(p_{i,u}, p_{j,v})$, and sends it to CSP.

The same operation is performed on each CU and corresponding candidate CUs. Finally, CSP has a matrix $\Phi$ that stores the similarity between all CUs.

### E. CU Grouping

In this process, CSP can use $\Phi$ to provide CU grouping results for CUs in any appropriate way.

In our scheme, two simple CU grouping strategies are described below. One strategy is Largest Similarity First (LSF), which always selects the largest element in $\Phi$ at present. This strategy is mainly to ensure global optimization, prioritizing the most suitable carpooling partners. Another strategy is First In First Out (FIFO), which always selects the first CU in $\Phi$ to enter the system at present. This strategy mainly ensures that CU can get a service response in time. When a pair of the carpooling partner is selected, the rows and columns where the carpooling partners are located will be removed. Repeat this process, and successful matching carpooling partners will be produced in pairs. If the largest $\phi$ in $\Phi$ is lower than a certain threshold, the matching process of this round can be stopped. The remaining unsuccessfully matched CUs will compute the similarities in the subsequent matching process.

Furthermore, the above two strategies are relatively simple. In practical application, these two strategies can be combined to consider the whole and the individual's benefits jointly. For example, based on LSF policy, the weight of entry order is added for each CU, and the weight of CUs who enter earlier is higher. Therefore, a new coefficient is defined by combining similarity and order weight. Two suitable CUs with the largest new coefficient are selected for matching each time. Such a strategy ensures that the similarity of successful matching CUs is high enough and makes it possible that no CU cannot find a suitable matching CU all the time.

### F. Extension

In the description above, all the route computations involved are two routes. However, TAROT can easily be extended to multiple routes. The Jaccard similarity is extended to the ratio of intersection and union of multiple routes. Two core algorithms in TAROT, DRFA and ASCA, are extended respectively.

*DRFA.* Similar to the description of the relationship between $\psi$ and $\phi$ in Theorem 1, there may be three cases of elements at the same position of routes in binary, all are 1, some are 0, some are 1, and all are 0. These three cases are marked as $\mathring{x}$, $\mathring{y}$, and $\mathring{z}$. In addition, the number of occurrences of these three situations is denoted as $x$, $y$, and $z$. $\psi = {}^{z}/_{z+y}$ and $\phi = {}^{x}/_{x+y}$ can also be obtained. There still exists $\psi > \phi$. Therefore, the DRFA is still valid for multiple routes.

*ASCA.* There is no essential difference between similarity computation between multiple routes and two routes. The ASCA will call the GMEDA to compare the points in the routes. The similarity computation process is consistent with that on the plaintext. Therefore, the ASCA can run among multiple routes.

In summary, TAROT supports similarity computation among multiple routes. In practical application, CU can choose the number of CUs that can share a car. At this time, it is necessary to compute the similarity between multiple routes.

## VI. SECURITY ANALYSIS

In this section, we analyze the security of TAROT. Since dissimilar route filter and accurate similarity computation are two key processes of TAROT, we show the security of TAROT

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



by respectively proving that dissimilar route filter and accurate similarity computation are privacy-preserving.

In some previous work, the starting points and the destinations are often the private information that attackers want to steal [6], [18], [25], [26]. In addition, since the stay points are location points chosen by CU, they are likely to be sensitive locations. Therefore, in TAROT, we consider that the starting points, destinations, and stay points of CU are all privacy information. In other words, if the starting points, destinations, and stay points of CU can be restored, then we believe that privacy has been leaked. On the contrary, if the starting points, destinations, and stay points of CU cannot be restored, then we believe that privacy is protected. In TAROT, $\mathrm{CU}_i$ and $\mathrm{CU}_j$ will complete the similarity computation with the help of CSP.

### A. Security Analysis of Dissimilar Route Filter

Based on the security model of our scheme, we prove that the dissimilar route filter process is privacy-preserving, keeping CUs' location secret from the honest-but-curious CSP. Since CSP knows the value of $\Lambda$, it can determine whether $s_{i,u}^t$ is the first element in $\lambda_t$. If $s_{i,u}^t$ is the first element of $\lambda_t$, CSP can only learn that $s_{i,u}^t$ is the point that $\mathrm{CU}_i$ did not pass. If $s_{i,u}^t$ is not the first element of $\lambda_t$, the elements before $s_{i,u}^t$ are the points where $\mathrm{CU}_i$ passes. Therefore, the CSP can obtain the passing points and non-passing points of $\mathrm{CU}_i$. However, it still has no idea on CUs' location from these information as shown in Theorem 2.

**Lemma 1.** *The CSP has no idea on CUs' starting points, destinations, and stay points based on the leaked passing points in dissimilar route filter.*

*Proof.* First, we only consider the privacy leakage caused by $\lambda_t$ and $p_{i,u}$. Let $s_{i,u}^v$ be the $(X+1)$th element in $\lambda_t$. The $X$ elements before $s_{i,u}^v$ are the points that $\mathrm{CU}_i$ passes. If the length of $p_{i,u}$ is assumed to be $\rho$, $X \in \{0, 1, 2, \ldots, \rho-1, \rho\}$. The probability of the number of leakage passing points is

$$\Pr[X=k] = \frac{\binom{\rho}{k}}{\binom{\iota}{k}} \cdot \frac{\iota-\rho}{\iota-k} = \frac{\rho!(\iota-k)!(\iota-\rho)}{\iota!(\rho-k)!(\iota-k)},$$

where $\binom{\rho}{k}$ is the symbol of $\rho$ choose $k$. For $\lambda_t$ and $p_{i,u}$, the number of passing points that will leak on average is

$$\mathrm{E}(X) = \sum_{k=0}^{\rho} k \cdot \Pr[X=k] = \frac{\rho}{\iota-\rho+1}.$$

Accordingly, the variance

$$\mathrm{Var}(X) = \mathrm{E}(X^2) - \mathrm{E}^2(X) = \frac{\rho(\iota-\rho)(\iota+1)}{(\iota-\rho+2)(\iota-\rho+1)^2}.$$

This means that if one $\lambda$ acts on $p_{i,u}$, there will be $\mathrm{E}(X)$ points leaking. Let $X_t$ represent the number of leakage passing points caused by $\lambda_t$, then $\mathrm{E}(X_t) = \mathrm{E}(X)$ and $\mathrm{Var}(X_t) = \mathrm{Var}(X)$. Let $S_\tau = \sum_{v=1}^{\tau} X_t$, $\mu = \mathrm{E}(X_t)$, and $\sigma = \sqrt{\mathrm{Var}(X_t)}$. Applying the central limit theorem, ${}^{S_\tau - \tau\mu}/_{\sqrt{\tau}\sigma}$ approximately obeys standard normal distribution $\mathcal{N}(0,1)$. Applying the three-sigma rule of thumb, we have $\Pr[S_\tau \leqslant \tau\mu + \sqrt{\tau}\sigma] \approx 0.84134$, $\Pr[S_\tau \leqslant \tau\mu + 2\sqrt{\tau}\sigma] \approx 0.97725$, and $\Pr[S_\tau \leqslant \tau\mu + 3\sqrt{\tau}\sigma] \approx 0.99865$.

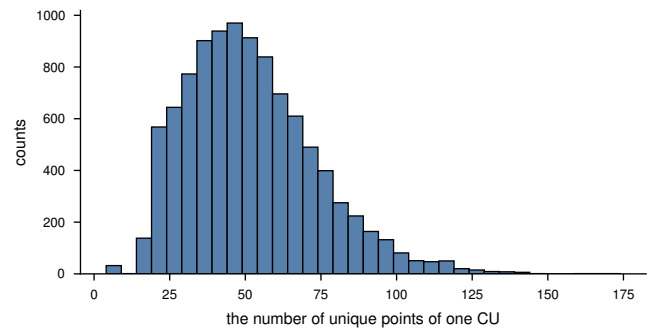



Fig. 6. Distribution related to the number of one CU's unique points.

TABLE II
THRESHOLD OF DIFFERENT $\rho$ UNDER THREE-SIGMA RULE OF THUMB

| $\rho$ | 20 | 35 | 50 | 65 | 80 |
|---|---|---|---|---|---|
| $\tau\mu + \sqrt{\tau}\sigma$ | 1.51 | 2.31 | 3.08 | 3.84 | 4.60 |
| $\tau\mu + 2\sqrt{\tau}\sigma$ | 2.34 | 3.42 | 4.43 | 5.40 | 6.35 |
| $\tau\mu + 3\sqrt{\tau}\sigma$ | 3.17 | 4.53 | 5.77 | 6.95 | 8.10 |
| ${}^{\tau\mu+\sqrt{\tau}\sigma}/_{\rho}$ | 7.53 % | 6.61 % | 6.17 % | 5.91 % | 5.75 % |
| ${}^{\tau\mu+2\sqrt{\tau}\sigma}/_{\rho}$ | 11.68 % | 9.78 % | 8.86 % | 8.30 % | 7.94 % |
| ${}^{\tau\mu+3\sqrt{\tau}\sigma}/_{\rho}$ | 15.83 % | 12.95 % | 11.55 % | 10.69 % | 10.12 % |

Only one route is considered in the above analysis. However, there are $k_i$ routes for $\mathrm{CU}_i$. These routes are generated by the KSP algorithm or distance matrix with the same starting point, destination, and stay points. Most of the points in these routes are repeated. Because we analyze the passing points, it is meaningless to compute separately and repeatedly count for multiple routes. Therefore, multiple routes can be considered as a longer route when analyzing the leakage of passing points.

In order to get the number of unique points in each CU's route vector set, we performed a simulation. In our simulation, 10 000 CUs were randomly generated. 15 routes were generated for each CU. The number of unique points in each CU's route vector set is counted, and the histogram is shown in Fig. 6.

Here, we use specific values directly, i.e., $\iota = 1200$ and $\tau = 40$. In order to show privacy leakage of passing points more intuitively, we computed the passing points range of different $\rho$ in TABLE II. We can see that there are no more than 6 passing points of leakage with 84.134 % probability, no more than 8 passing points with 97.725 % probability, and no more than 10 points with 99.865 % probability. The theoretical value will be on the high side because we do not consider the repetition of elements when we compute $S_\tau$. Combined with the value range of $\rho$, CSP cannot restore the starting points, destinations, and stay points. □

**Lemma 2.** *The CSP has no idea on CUs' starting points, destinations, and stay points based on the leaked non-passing points in dissimilar route filter.*

*Proof.* We analyze the leaked private information of non-passing points in detail. Every leaking non-passing point is an RM value, i.e., $s_{i,u}^t$, which is produced by $\lambda_t$ acting on

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



TABLE III
NUMBER OF LEAKED NON-PASSING POINTS ON AVERAGE

| $k_i$ | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|
| $\mathrm{E}(Y)$ | 340.3 | 368.5 | 395.8 | 422.1 | 447.6 | 472.3 |
| $\mathrm{E}(Y)/\iota$ | 28.4 % | 30.7 % | 33.0 % | 35.2 % | 37.3 % | 39.4 % |

the route vector $p_{i,u}$. Since every $\lambda$ is randomly generated, we think that $s_{i,u}^t$ is randomly selected from $\lambda_t$. The number of $\lambda$ in $\Lambda$ is $\tau$. For $p_{i,u}$, $\tau$ RM values are generated. There are $k_i$ routes for $\mathrm{CU}_i$. The points that each route produces without passing through may be different, so $k_i$ routes need to be considered. Therefore, for $\mathrm{CU}_i$, to get the total number of leaking non-passing points, in fact, it is to compute how many unique values are taken out by randomly taking one value from each of $k_i \cdot \tau$ $\lambda$s.

In short, there are $\iota$ numbers, and each number is different. We randomly select a number with equal probability each time and record it, and repeat this process $k_i \cdot \tau$ times. Note that the number that has been selected before can continue to be selected. That is to say, for each selection, the probability of each number being selected is $1/\iota$. There may be cases where some numbers are equal among these selected numbers. Repeated numbers are deleted, and a set of unique elements are obtained. Finally, the amount of numbers obtained is $Y$. The $Y$ value obtained by the above process is the number of leaked points that have not passed. The probability of the number of leakage non-passing points is

$$\Pr[Y=k] = \frac{\left\{ {k_i \cdot \tau \atop k} \right\} \cdot \mathrm{P}(\iota, k)}{\iota^{k_i \cdot \tau}} = \frac{\iota! \sum\limits_{i=0}^{k} \frac{(-1)^i (k-i)^{k_i \cdot \tau}}{i!(k-i)!}}{\iota^{k_i \cdot \tau} (\iota - k)!},$$

where $\left\{ {k_i \cdot \tau \atop k} \right\}$ is the symbol of Stirling numbers of the second kind which counts the number of ways to partition a set of $k_i \cdot \tau$ elements into $k$ nonempty subsets, and $\mathrm{P}(\iota, k)$ is the number of $\iota$-permutations of $k$. For $\mathrm{CU}_i$, the number of non-passing points that will leak on average is

$$\mathrm{E}(Y) = \sum_{k=0}^{k_i \cdot \tau} k \cdot \Pr[Y=k] = \frac{\iota!}{\iota^{k_i \cdot \tau}} \sum_{k=0}^{k_i \cdot \tau} \frac{k \sum\limits_{i=0}^{k} \frac{(-1)^i (k-i)^{k_i \cdot \tau}}{i!(k-i)!}}{(\iota - k)(\iota - k - 1)!}.$$

It can be seen that there are three values $k_i$, $\iota$, and $\tau$, that affect the value of $\mathrm{E}(Y)$. For a more intuitive description here, for $\iota$ and $\tau$, we use specific values directly, i.e., $\iota = 1200$ and $\tau = 40$. We select the appropriate value of $k_i$, i.e., $k_i \in \{10, 11, 12, 13, 14, 15\}$. In order to show privacy leakage of non-passing points more intuitively, we computed the number of leaked non-passing points on average of different $k_i$ in TABLE III. We can see that the leaked non-passing points account for about 28 % to 40 % of all points on the map. According to the rules for generating points that are not leaked, we can know that these points are randomly distributed on the map. Through these points, CSP cannot restore CU's routes. That is to say, CSP cannot restore the starting points, destinations, and stay points of CU. □

**Theorem 2.** *The CSP has no idea on CUs' starting points, destinations, and stay points in dissimilar route filter.*

*Proof.* The privacy leakage of passing points and non-passing points is analyzed in Lemma 1 and Lemma 2. As shown in TABLE II and TABLE III, CSP gets 5.75 % to 15.83 % of $\mathrm{CU}_i$'s all possible route points and the points that $\mathrm{CU}_i$ has not passed, which account for 28 % to 40 % of the map. For a single CU, only through these information, CSP cannot analyze CU's starting points, destinations, and stay points. As for the joint analysis of multiple CU data, such as $\mathrm{CU}_i$ and $\mathrm{CU}_j$. In order to perform joint analysis, $\phi_{i,j}$ needs to be large so that CSP can synthesize the information of $\mathrm{CU}_i$ and $\mathrm{CU}_j$ to get more information. CSP can compute $\psi_{i,j}$ from $S_i$ and $S_j$. Owing to $\psi_{i,j} > \phi_{i,j}$, the larger $\psi_{i,j}$ will not reflect the real $\phi_{i,j}$, and the smaller $\psi_{i,j}$ can infer that $\phi_{i,j}$ is smaller. However, the smaller $\phi_{i,j}$ has no meaning for the joint analysis. Therefore, the joint analysis cannot analyze the starting point, destination and stay points. In summary, CSP cannot obtain the starting point, destination and stay points of CU in dissimilar route filter. □

### B. Security Analysis of Accurate Similarity Computation

Based on the security model of our scheme, we prove that the accurate similarity computation process is privacy-preserving, keeping CUs' location secret from the honest-but-curious CSP as shown in Theorem 3.

**Theorem 3.** *The CSP has no idea on CUs' starting points, destinations, and stay points in accurate similarity computation.*

*Proof.* The ASCA we designed is wholly based on the GMEDA. In the ASCA, the data in TAROT is converted to the format in the GMEDA, and the result is computed interactively. The security of the ASCA is entirely equivalent to the security of the GMEDA. The security of GMEDA and GM algorithm is consistent. The specific security analysis of the GM algorithm can be found in [36]. Therefore, CU in accurate similarity computation cannot obtain the privacy of each other and CSP also cannot obtain the starting point, destination, and stay points of CU. □

## VII. PARAMETER SELECTION

In the above description of TAROT, some operating parameters involved in some parts have not been determined. Specifically, it includes three parameters: 1) number of $\lambda$s; 2) number of routes selected by each CU; and 3) number of candidate CU selected by each CU. Next, we have to determine the selection of each parameter in turn through simulation. In order to make the simulation more general, $30 \times 40$ rectangular map is used in this section. The distance between the points on the map is randomly generated.

### A. The Number of RM Functions

In our simulation, 40 CUs are randomly generated. The $k_i$ value generated by each CU is taken as $5, 10, 15, \ldots, 100$ respectively.

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



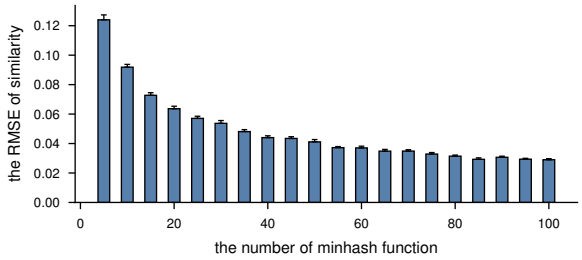



Fig. 7. Influence of different numbers of RM functions on similarity.

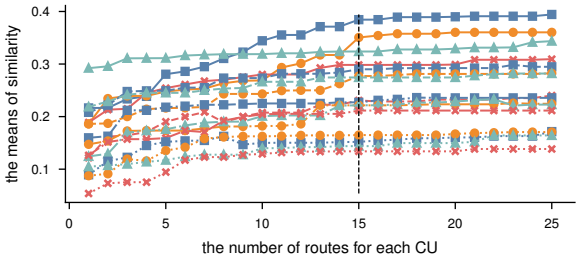



Fig. 8. Influence of different numbers of routes for each CU on similarity.

According to Theorem 1, the minhash method mentioned in Theorem 1 and the RM method use the same hash functions. Owing to the minhash method can accurately compute the Jaccard similarity, evaluating the number of hash functions used in the minhash method is more reasonable. The number of hash functions obtained is also the number of the RM functions. Each similarity between each pair of CUs is computed separately by the minhash method and the definition method in Definition 1. In addition, the Root Mean Square Error (RMSE) is computed between the two sets of data computed by these two methods. Obviously, the smaller the RMSE, the more accurate the similarity computed by the minhash method.In Fig. 7, we conducted four repeated experiments.

Finally, we can find that starting from the number of RM functions taking $40$, the RMSE basically no longer changes. Therefore, we will choose $40$ as the number of $\lambda$s in the subsequent simulation.

### B. The Number of Routes for Each CU

More routes will significantly bring in more overheads. Therefore, balancing between a larger similarity and a smaller computational costs is necessary.

In our simulation, 20 CUs are randomly generated, and each CU generates $k_i$ routes to compute the similarity. These 20 CUs generate a $20 \times 20$ similarity matrix. According to the principle of LSF, each CU is assigned a matching CU, and at the same time, each CU gets a corresponding similarity value.

As shown in Fig. 8, 16 independent repeated experiments are performed, and each experiment generates a new map, new CUs, new starting points, new destinations, and new routes. For each curve in Fig. 8, the horizontal axis represents the number of routes generated by each CU. At the same starting point and destination, $k_i$ takes from 1 to 25 in sequence. The ordinate represents the mean of the similarity obtained by these 20 CUs under a certain number of routes.

Therefore, under the conditions of this simulation, we can point out that each CU can take $15$ routes to achieve higher similarity with lower computational costs.

### C. The Percentage of Candidate CUs for Each CU

In order to study the situation under different CU numbers, we tested 6 different numbers of CUs. As shown in Fig. 9, the numbers of CU pairs are $45$, $190$, $435$, $780$, $1225$ and $1770$. Each CU generates $15$ routes to compute the similarity. According to the principle of LSF, each CU is assigned a matching CU, and at the same time, each CU gets a corresponding similarity value. So far, we have a vector of which each element is the similarity of the corresponding CU. This vector is used as a benchmark vector to select the number of candidate CUs.

For each curve in Fig. 9, it represents an independent repeated experiment. The abscissa represents the percentage of candidate CUs selected by each CU, that is, the ratio of the number of candidate CUs to the total number of CUs. Under the corresponding number of candidate CUs, the computed similarity value of each CU can form a vector. The ordinate value is the RMSE between this vector and the benchmark vector. The smaller the ordinate value, the better the matching effect. This means the closer to the matching result obtained from the original true similarity matrix.

Therefore, under the conditions of this simulation, we can point out that each CU can take $20\,\%$ of the candidate CUs to achieve closer to the true similarity with lower computational costs.

## VIII. Experimental Evaluation

In this section, the leakage of passing and non-passing points in dissimilar route filter is evaluated, and performance evaluation is performed in terms of computational costs and communication overheads.

### A. Simulation Environment

In order to make the simulation more convenient, we used Python 3 for programming. We use workstations to simulate CSP and CUs' terminals. The workstations are equipped with Intel® Xeon® Silver 4110 Processor ($2.10\,$GHz), which are connected through IEEE 802.11ac WLAN. The security parameter in the GMEDA, i.e., the bit length of $n$, is $1024$.

### B. Evaluation of Leaked Points in Dissimilar Route Filter

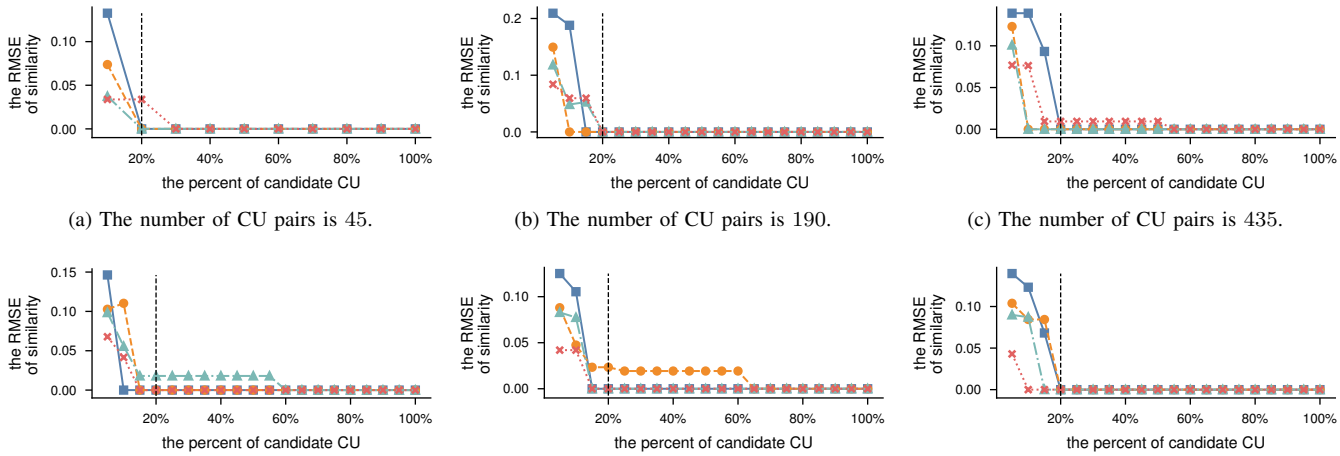
In the analysis of Theorem 2 and Theorem 3, the computed results are upper limit rather than the exact value. In order to evaluate the leakage of passing and non-passing points in dissimilar route filter in practice, extensive experiments are conducted.

In order to evaluate the number of leaked passing points, $100$ random maps of $30 \times 40$ are generated. In each map, there are $300$ random CUs. Each CU generates $15$ original routes and uses $40$ $\lambda$s to generate the corresponding RM route vector. The theoretical and observed value of the number of leaked

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal






(a) The number of CU pairs is 45.

(b) The number of CU pairs is 190.

(c) The number of CU pairs is 435.

(d) The number of CU pairs is 780.

(e) The number of CU pairs is 1225.

(f) The number of CU pairs is 1770.

Fig. 9. Influence of different numbers of candidates for each CU on similarity.

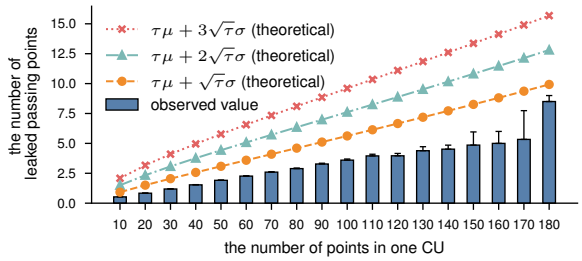



Fig. 10. Leaked passing points produced by different numbers of points.

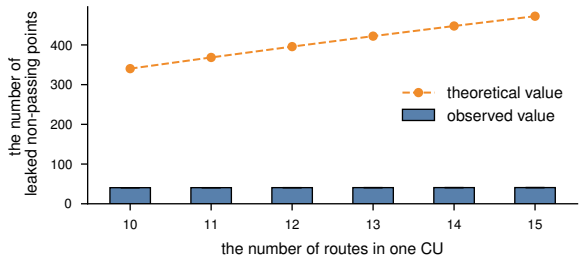



Fig. 11. Leaked non-passing points produced by different numbers of routes.

passing points is shown in Fig. 10. It can be seen that the theoretical value is close to the observed value.

In order to evaluate the number of leaked non-passing points, 100 random maps of $30 \times 40$ are generated. There are 500 random CUs on each map. Each CU generates 10, 11, 12, 13, 14 and 15 original routes and uses 40 $\lambda$s to generate the corresponding RM route vector. The theoretical value and observed value of the number of leaked non-passing points is shown in Fig. 11. It can be seen that the observed value is far less than the theoretical value, averaging a little more than 40. Owing to most of a CU's routes are very similar, which cannot be considered quantitatively in our theoretical analysis.

Combining Theorem 2, Theorem 3, and the above evaluation, we know that CUs' starting points and destinations are protected.

### C. Computational Costs

In terms of computational costs in TAROT, we mainly focus on the computational costs of the CU side. In CU route generation and accurate similarity computation, there are many computations on the CU side.

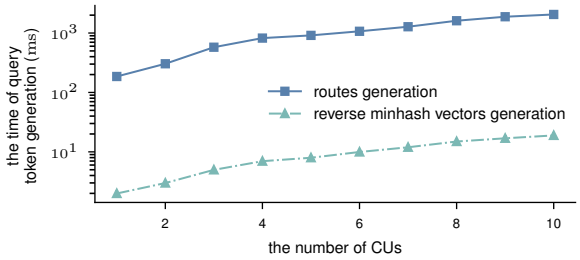
In the CU route generation process, CU will generate two different forms of routes, namely the original route and the RM route vector. We tested the runtime and communication data generated by multiple CUs. The experiment is still using a $30 \times 40$ random map to randomly generate a corresponding number of CUs, and each CU generates 15 original routes. 40 $\lambda$s are used to generate the corresponding RM route vector. As shown in Fig. 12a, in the route generation process, for each CU, the most time-consuming is the original route generation algorithm. In this part, we use the Yen's algorithm, which can be optimized or replaced by a more efficient KSP algorithm in practical applications. The runtime in the RM vector generation part is very short and can be almost ignored.

In accurate similarity computation, we propose the GMEDA and apply it to our ASCA. In the ASCA, $\mathrm{CU}_i$ and $\mathrm{CU}_j$ communicate via CSP relay twice. Assuming that $\mathrm{CU}_i$ is the party initiating communication, $\mathrm{CU}_i$ will perform two computations, and $\mathrm{CU}_j$ will perform one computation. The runtime of these three computations and the initial step is shown in Fig. 12b. The initial step is used to generate public and private key pairs. $\mathrm{CU}_i$ can compute the first step immediately after generating the routes, and it only needs to be computed once for each CU. The computation time of the initial, first, and second steps is completely negligible. The second step requires $\mathrm{CU}_j$ to complete the computation. The third step computation performed by $\mathrm{CU}_i$ is relatively the most time-consuming. On average, the third step between a pair of CU takes 400 ms. In addition, due to the introduction of the DRFA, the actual amount of computation required for accurate similarity computation is greatly reduced. Therefore, on the whole, the computational costs of TAROT is completely acceptable.

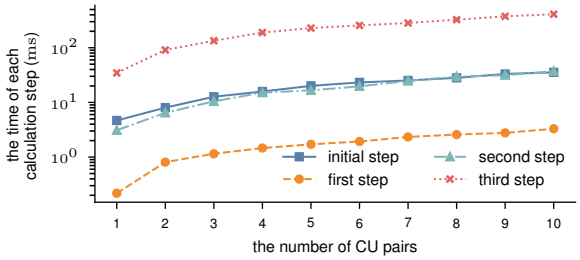
2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



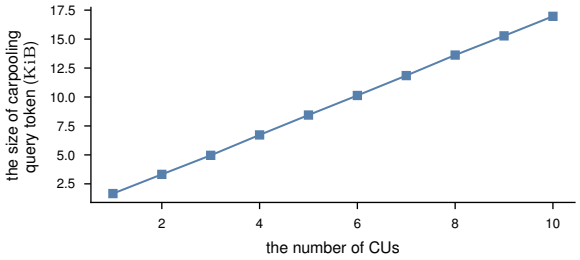



(a) Runtime of query token generation.

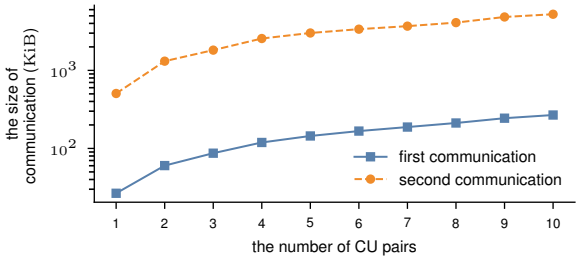



(b) Runtime of accurate similarity computation.

Fig. 12. Computational costs in TAROT.




(a) Communication overheads of query token generation.




(b) Communication overheads of accurate similarity computation.

Fig. 13. Communication overheads in TAROT.

### D. Communication Overheads

In TAROT, there are actually two types of communication processes without considering CSP forwarding communication. One is communication between CUs and CSP, and the other is communication between CUs. The former occurs after CU generates the routes, and CU needs to send the RM route vectors to CSP. The latter occurs in the process of computing the accurate similarity, and CUs perform interactive computations to complete accurate similarity computation. During this period, there will be two communications between each pair of CUs.

In the CU route generation step, each CU will generate $15$ routes and generates $40$ RM route vectors according to $40$ $\lambda$s. Finally, the $40$ RM route vectors are sent to CSP as a carpooling query token. We test the size of multiple CUs' query token packets and plot the data. As shown in Fig. 13a, the communication overheads of CU's carpooling query token are not large.

In the accurate similarity computation process, for a pair of $\mathrm{CU}_i$ and $\mathrm{CU}_j$, there are two communications. Assuming that $\mathrm{CU}_i$ is the initiator of the interaction, $\mathrm{CU}_i$ encrypts the points passed in $15$ routes and sends them to $\mathrm{CU}_j$. $\mathrm{CU}_j$ fuses the passing points of his $15$ routes with the ciphertext of each point sent by $\mathrm{CU}_i$. Finally, $\mathrm{CU}_j$ sent it to $\mathrm{CU}_i$. We test the sum of the data of multiple CUs and plot the data. As shown in Fig. 13b, the second communication volume is much larger than the first communication volume. Nevertheless, it is acceptable.

## IX. Related Work

In this section, we introduce some privacy-preserving technologies in ride scenes. At present, there are mainly carpooling (ride-sharing) and ride-hailing.

*Ride-sharing.* He *et al.* [25] designed a selection mechanism based on spatial region. With the help of encrypted trips of drivers and passengers, the ride-sharing server determines the appropriate ride-sharing partners, which saves the expenses of passengers and drivers. Goel *et al.* [18] proposed a negotiation based model, which can hide the exact location information data for system participants, and at the same time, implement privacy-preserving ride-sharing. Use the concept of imprecision, i.e., imprecision of the user's position in the set of $n$ positions, and follow the idea of confusion, that is, higher inaccuracy and higher privacy. Ni *et al.* [40] proposed an anonymous mutual authentication protocol to solve the contradiction between safety and privacy preservation by using BBS+ signature. In anonymous mutual authentication protocol, passengers and drivers can verify each other's identities without revealing their actual identities, but they can display their membership in trusted groups. Hallgren *et al.* [32] proposed a ride-sharing model for privacy-preserving. The secure multiparty computation technology for location and trajectory matching is developed, so that the third party can be trusted. Users can learn about the ride segments they can share, but they cannot get anything about the locations of other users. Aïvodji *et al.* [26] proposed a privacy-preserving protocol for ride-sharing, which solved the matching problem of dynamic ride-sharing systems. Depends on secure filtering protocols to compute feasible matches. Sherif *et al.* [6] proposed a privacy-preserving scheme for ride-sharing. Similarity measurement technology is used for encrypted data to protect the privacy of travel data. The server can measure the similarity of users' travel data, and find users that can be shared without knowing the real data. Pagnin *et al.* [27] proposed a simple yet powerful method which represent a private set intersection on trips as sets of consecutive road segments, and kept time preference without relying on third parties.

*Ride-hailing.* Yu *et al.* [28] proposed a secure comparison protocol to efficiently optimize the schedules of taxis over encrypted data which can protect the location privacy of both riders and taxis in online ride-hailing (ORH) services.

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal



TABLE IV
COMPARISON AMONG OUR TAROT SCHEME AND OTHER PRIVACY-PRESERVING CARPOOLING SCHEMES

| | TAROT | [25] | [18] | [40] | [32] | [26] | [6] | [27] |
|---|---|---|---|---|---|---|---|---|
| Accuracy | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ |
| Map Distance | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ |
| Efficiency Filter | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✓ | ✓ |
| User Role Convertible[1] | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ |
| Aiming at Travel Saving | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |

[1] User can be either a driver or riders, which is more general.

Wang *et al.* [41] proposed two efficient and secure spatial query algorithms for ORH service. All location-related data are encrypted by its owner before being sent out and are calculated without decryption during the spatial query process. Xie *et al.* [42] used a distance calculation technique to securely compute the user distance while protecting the location privacy of both riders and drivers in ORH services. Luo *et al.* [29] designed a scheme to estimate the shortest distances between riders and drivers in road networks approximately without revealing the location privacy of riders and drivers in ORH services. Yu *et al.* [30] used the modified Paillier cryptosystem to compute approximate road distance over encrypted data in ORH services and designed a secure comparison method to compare two distances over ciphertexts without revealing the actual distances.

We selected seven prior works about privacy-preserving carpooling and compared them with our scheme from the five key aspects: accuracy, map distance, efficiency filter, user role convertible, and aiming at travel saving. As shown in TABLE IV, in these aspects, TAROT has advantages. Different from existing ride-sharing or ride-hailing schemes, two route matching algorithms are proposed to balance efficiency and accuracy in TAROT. On the premise of ensuring the accuracy of similarity computation, the computational costs and communication overheads are significantly reduced.

## X. CONCLUSION

In this paper, we propose an efficient and privacy-preserving route matching scheme for carpooling services on a large scale, called TAROT. TAROT archives carpooling services without leaking CUs' accurate location and route information based on our proposed DRFA and ASCA. Specifically, before being sent out, all of the location and route related data were masked into ciphertexts by its owner and were computed without decryption during the carpooling query process. Therefore, CSP cannot obtain CUs' sensitive information, and each CU cannot obtain other sensitive information. Meanwhile, the proposed scheme significantly improved the carpooling query efficiency based on the DRFA. Detailed security analysis showed its privacy-preserving ability, and extensive experiments demonstrated its efficiency.

## XI. AVAILABILITY

The implementation of the proposed scheme and relevant information can be downloaded at https://www.xdzhuhui.com /demo/TAROT/.

## REFERENCES


[1] N. A. H. Agatz, A. L. Erera, M. W. P. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," *Eur. J. Oper. Res.*, vol. 223, no. 2, pp. 295–303, 2012.

[2] Y. Hou, W. Zhong, L. Su, K. F. Hulme, A. W. Sadek, and C. Qiao, "TASeT: Improving the efficiency of electric taxis with transfer-allowed rideshare," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9518–9528, 2016.

[3] PYMNTS. (2021, Dec.) Consumer ridesharing spending expected to reach $937B by 2026. Accessed in Internet Archive. [Online]. Available: https://www.pymnts.com/news/ridesharing/2021/consumer-ridesharing-spending-expected-reach-937-billion-dollars-2026/

[4] J. Menze. (2021, Apr.) BlaBlaCar raises $115M to expand carpool and bus services. Accessed in Internet Archive. [Online]. Available: https://www.phocuswire.com/blablacar-raises-115-million-to-expand-carpool-and-bus-services/

[5] L. Sommerfeld. (2021, Dec.) Let Waze offer a little seasonal levity to your drive. Accessed in Internet Archive. [Online]. Available: https://driving.ca/column/lorraine/let-waze-offer-a-little-seasonal-levity-to-your-drive/

[6] A. B. T. Sherif, K. Rabieh, M. M. E. A. Mahmoud, and X. Liang, "Privacy-preserving ride sharing scheme for autonomous vehicles in big data era," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 611–618, 2017.

[7] M. Mcnulty. (2021, Jul.) Doorbell video shows man outside Las Vegas home saying 'open the door, I want to rape and kill you'. Accessed in Internet Archive. [Online]. Available: https://www.dailymail.co.uk/news/article-9818815/

[8] J. Devanesan. (2020, Sep.) A cause for concern? Grab slapped with (another) data privacy fine. Accessed in Internet Archive. [Online]. Available: https://techwireasia.com/2020/09/a-cause-for-concern-grab-slapped-with-another-data-privacy-fine/

[9] A. Benveniste. (2021, Jul.) China bans Didi, its biggest ride-hailing service, from app stores. Accessed in Internet Archive. [Online]. Available: https://www.cnn.com/2021/07/04/tech/china-app-store-didi/

[10] European Parliament and Council of the European Union. (2016, Apr.) General Data Protection Regulation. Accessed in Internet Archive. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj/

[11] California Secretary of State. (2020, Nov.) The California Privacy Rights Act of 2020. Accessed in Internet Archive. [Online]. Available: https://vig.cdn.sos.ca.gov/2020/general/pdf/topl-prop24.pdf

[12] M. Chan, H. Elsherbini, and X. Zhang, "User density and spatial cloaking algorithm selection: Improving privacy protection of mobile users," in *Sarnoff Symposium*. IEEE, 2016, pp. 1–2.

[13] H. Jadallah and Z. A. Aghbari, "Spatial cloaking for location-based queries in the cloud," *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 9, pp. 3339–3347, 2019.

[14] N. Cui, X. Yang, and B. Wang, "A novel spatial cloaking scheme using hierarchical hilbert curve for location-based services," in *WAIM (2)*, ser. Lecture Notes in Computer Science, vol. 9659. Springer, 2016, pp. 15–27.

[15] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *CCS*. ACM, 2013, pp. 901–914.

[16] M. Zurbarán, K. Ávila, P. M. Wightman, and M. Fernández, "Near-Rand: Noise-based location obfuscation based on random neighboring points," in *LATINCOM*. IEEE, 2014, pp. 1–6.

[17] M. Zurbarán, P. M. Wightman, M. A. Brovelli, D. Oxoli, M. Iliffe, M. Jimeno, and A. Salazar, "NRand-K: Minimizing the impact of location obfuscation in spatial analysis," *Trans. GIS*, vol. 22, no. 5, pp. 1257–1274, 2018.

[18] P. Goel, L. Kulik, and K. Ramamohanarao, "Privacy-aware dynamic ride sharing," *ACM Trans. Spatial Algorithms Syst.*, vol. 2, no. 1, pp. 4:1–4:41, 2016.

[19] J. Tang, H. Zhu, R. Lu, X. Lin, H. Li, and F. Wang, "DLP: Achieve customizable location privacy with deceptive dummy techniques in lbs applications," *IEEE Internet Things J.*, pp. 1–16, 2021.

[20] B. Niu, Z. Zhang, X. Li, and H. Li, "Privacy-area aware dummy generation algorithms for location-based services," in *ICC*. IEEE, 2014, pp. 957–962.

[21] H. Liu, X. Li, H. Li, J. Ma, and X. Ma, "Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services," in *INFOCOM*. IEEE, 2017, pp. 1–9.

[22] H. J. Do, Y. Jeong, H. Choi, and K. Kim, "Another dummy generation technique in location-based services," in *BigComp*. IEEE Computer Society, 2016, pp. 532–538.


2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2022.3168661, IEEE Internet of Things Journal




[23] M. Baza, M. Mahmoud, G. Srivastava, W. Alasmary, and M. Younis, "A light blockchain-powered privacy-preserving organization scheme for ride sharing services," in *VTC Spring*. IEEE, 2020, pp. 1–6.

[24] X. Wang, "One-round secure fair meeting location determination based on homomorphic encryption," *Inf. Sci.*, vol. 372, pp. 758–772, 2016.

[25] Y. He, J. Ni, X. Wang, B. Niu, F. Li, and X. Shen, "Privacy-preserving partner selection for ride-sharing services," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5994–6005, 2018.

[26] U. M. Aïvodji, K. Huguenin, M. Huguet, and M. Killijian, "SRide: A privacy-preserving ridesharing system," in *WISEC*. ACM, 2018, pp. 40–50.

[27] E. Pagnin, G. Gunnarsson, P. Talebi, C. Orlandi, and A. Sabelfeld, "TOPPool: Time-aware optimized privacy-preserving ridesharing," *Proc. Priv. Enhancing Technol.*, vol. 2019, no. 4, pp. 93–111, 2019.

[28] H. Yu, X. Jia, H. Zhang, X. Yu, and J. Shu, "PSRide: Privacy-preserving shared ride matching for online ride hailing systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 18, no. 3, pp. 1425–1440, 2021.

[29] Y. Luo, X. Jia, S. Fu, and M. Xu, "pRide: Privacy-preserving ride matching over road networks for online ride-hailing service," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 7, pp. 1791–1802, 2019.

[30] 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 Trans. Veh. Technol.*, vol. 68, no. 11, pp. 10 418–10 428, 2019.

[31] P. A. Hallgren, M. Ochoa, and A. Sabelfeld, "InnerCircle: A parallelizable decentralized privacy-preserving location proximity protocol," in *PST*. IEEE Computer Society, 2015, pp. 1–6.

[32] P. A. Hallgren, C. Orlandi, and A. Sabelfeld, "PrivatePool: Privacy-preserving ridesharing," in *CSF*. IEEE Computer Society, 2017, pp. 276–291.

[33] C. Dong, L. Chen, and Z. Wen, "When private set intersection meets big data: an efficient and scalable protocol," in *CCS*. ACM, 2013, pp. 789–800.

[34] O. Goldreich, *The Foundations of Cryptography - Volume 2: Basic Applications*. Cambridge University Press, 2004.

[35] P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, no. 2, pp. 37–50, 1912.

[36] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[37] A. Z. Broder, "On the resemblance and containment of documents," in *SEQUENCES*. IEEE, 1997, pp. 21–29.

[38] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quart. Appl. Math.*, vol. 27, no. 4, pp. 526–530, 1970.

[39] G. Weyenberg and R. Yoshida, "Reconstructing the phylogeny: Computational methods," in *Algebraic and Discrete Mathematical Methods for Modern Biology*, 2015, pp. 293–319.

[40] J. Ni, K. Zhang, X. Lin, H. Yang, and X. S. Shen, "AMA: anonymous mutual authentication with traceability in carpooling systems," in *ICC*. IEEE, 2016, pp. 1–6.

[41] F. Wang, H. Zhu, X. Liu, R. Lu, F. Li, H. Li, and S. Zhang, "Efficient and privacy-preserving dynamic spatial query scheme for ride-hailing services," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11 084–11 097, 2018.

[42] H. Xie, Y. Guo, and X. Jia, "A privacy-preserving online ride-hailing system without involving a third trusted server," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 3068–3081, 2021.

**Qi Xu** was born in China in 1995. He received the B.Eng. degree in information security from Xidian University, Xi'an, Shaanxi, China, in 2017. He is currently pursuing the Ph.D. degree in cyberspace security at Xidian University, Xi'an, Shaanxi, China.

His research has been concerned with privacy-preserving location based services.

**Hui Zhu** (M'13–SM'19) received the B.Sc. degree from Xidian University, Xi'an, Shaanxi, China, in 2003, the M.Sc. degree from Wuhan University, Wuhan, Hubei, China, in 2005, and the Ph.D. degree from Xidian University, Xi'an, Shaanxi, China, in 2009.

He was a Research Fellow with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore, in 2013. Since 2016, he has been a Professor with the School of Cyber Engineering, Xidian University. His current research interests include applied cryptography, data security, and privacy.

**Yandong Zheng** received her M.S. degree from the Department of Computer Science, Beihang University, China, in 2017 and she is currently pursuing her Ph.D. degree in the Faculty of Computer Science, University of New Brunswick, Canada.

Her research interest includes cloud computing security, big data privacy and applied privacy.

**Jiaqi Zhao** was born in China, in 1997. He received the B.Eng. degree in information security from Xidian University, Xi'an, Shaanxi, China, in 2020. He is currently pursuing the Ph.D. degree in cyberspace security at Xidian University, Xi'an, Shaanxi, China.

His research has been concerned with federated learning.

**Rongxing Lu** (S'09–M'11–SM'15–F'21) is an Associate Professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. His research interests include applied cryptography, privacy enhancing technologies, and the IoT-big data security and privacy. He was the Winner of the 2016–2017 Excellence in Teaching Award from the FCS, UNB. He was the recipient of nine best (student) paper awards from some reputable journals and conferences. Currently, he serves as the Chair for the IEEE ComSoc Communications and Information Security Technical Committee (CIS-TC), and the Founding Co-Chair for the IEEE TEMS Blockchain and Distributed Ledgers Technologies Technical Committee (BDLT-TC).

**Hui Li** (M'10) Received his B.Sc. degree from Fudan University in 1990, M.Sc. and Ph.D. degrees from Xidian University in 1993 and 1998, respectively.

Since 2005, he has been the professor in the school of Telecommunication Engineering, Xidian University, China. His research interests are in the areas of cryptography, wireless network security, information theory and network coding.

Dr. Li served as TPC co-chair of ISPEC 2009 and IAS 2009, general co-chair of E-Forensic 2010, ProvSec 2011 and ISC 2011, honorary chair of NSS 2014, ASIACCS 2016.

2327-4662 (c) 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: University of New Brunswick. Downloaded on June 09,2022 at 00:41:58 UTC from IEEE Xplore. Restrictions apply.