Ad Hoc Networks 141 (2023) 103077

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Fully privacy-preserving location recommendation in outsourced environments

Lulu Han [a], Weiqi Luo [a,*], Anjia Yang [a], Yandong Zheng [b], Rongxing Lu [c], Junzuo Lai [a], Yudan Cheng [a]

[a] The College of Cyber Security, Jinan University, Guangzhou 510632, China
[b] The State Key Laboratory of Integrated Services Networks, Xidian University, Xian 710071, China
[c] The Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada

ARTICLE INFO

Keywords:
Privacy-preserving
Location recommendation
Paillier cryptosystem
Secure two-party computation

ABSTRACT

Currently, location-based services (LBS) have been widely used in real-world settings, including restaurant and travel recommendations. To reduce workload and improve query efficiency, a service provider usually outsources its services to a powerful cloud server. However, the service provider's database and users' queries always contain sensitive information, so their leakage to the cloud may raise serious privacy concerns. Although some existing schemes have been proposed to address the privacy problems, they are impractical in real-world LBS due to some issues in privacy, accuracy, or heavy computation costs for query users. In order to overcome these problems, we propose a fully privacy-preserving location recommendation scheme that supports multi-attribute queries and returns accurate results based on the recommendation condition. Specifically, based on the Paillier cryptosystem, we first propose a secure equal test (SET) protocol to check whether two encrypted values are equal. Second, with our proposed protocols, we develop a privacy-preserving location recommendation scheme without revealing anything about the service provider or query users. Finally, we analyze the security of our scheme in the semi-honest model and show that the privacy of the service provider and query users is well protected. Meanwhile, we evaluate the performance of our scheme using synthetic datasets. The experimental results demonstrate that our proposed scheme is practical in real-world applications.

1. Introduction

In recent years, location-based services (LBSs) have become increasingly popular with the rapid development of location acquisition technology. According to a report of the Verified Market Research,[1] the global LBS market size will reach \$137.06 Billion in 2030 from \$19.25 Billion in 2022. Thanks to the flourishing mobile positioning technologies, LBS makes our life more convenient in many aspects, such as restaurant recommendations [1], travel recommendations [2], and friend recommendations [3,4].

Consider the following scenario: when users arrive at a strange place, they always hope to find some restaurants matching their preferences, such as near to their visited restaurants, with the same cuisine and the acceptable average price per head. In this scenario, users can request a query from the dining service provider for finding new restaurants. The dining service provider chooses the best restaurants based on query requirements and recommends the corresponding records (i.e., restaurant coordinate, cuisine, average price) to users. Fig. 1 gives the restaurant records of the dining service provider and the dining records (the visited restaurants' records) of a user. If the user requests a query for the restaurants with British cuisine, he will receive the restaurant record $\{10112, (12, 90), British, 58\}$ from the dining service provider.

However, in the real-world setting, the user wants to find the restaurants matching as much his preferences as possible, so he will request a query with two or more conditions. For example, he requires the recommended restaurants (1) close to one of his visited restaurants within a predefined distance threshold and (2) matching his cuisines. This query can be easily realized in the plaintext domain, while it will be a challenge in the ciphertext domain. We consider the query with two or more conditions as the multi-attribute query.

Meanwhile, as the dining service provider's database (restaurant records) and query users grow, such location recommendation

* Corresponding author.
E-mail addresses: locomotive_crypto@163.com (L. Han), lwq@jnu.edu.cn (W. Luo), anjiayang@gmail.com (A. Yang), zhengyandong@xidian.edu.cn (Y. Zheng), rlu1@unb.ca (R. Lu), laijunzuo@gmail.com (J. Lai), yudan_cheng@foxmail.com (Y. Cheng).

[1] https://www.verifiedmarketresearch.com/.

https://doi.org/10.1016/j.adhoc.2022.103077
Received 8 October 2022; Received in revised form 19 December 2022; Accepted 20 December 2022
Available online 2 January 2023

Dining Service Provider

Rest ID	Coord	Cuisine	Aveg
10112	(12,90)	British	58
90054	(13,28)	Chinese	55
32789	(77,96)	Chinese	78
87103	(89,95)	Indian	92

(a) Restaurant Records

User

Rest ID	Coord	Cuisine	Aveg
123157	(17, 30)	British	80
245901	(92,101)	Chinese	70

(b) Dining Records

Fig. 1. Restaurant records for the dining service provider (a) and dining records for the user (b).

services will be seriously affected in efficiency. To reduce query latency, the dining service provider usually outsources its location recommendation service to the cloud. However, the cloud is not completely trusted [5–8]. In addition, since the restaurant records of the dining service provider, the user's queries, and query results contain a certain amount of sensitive information, the privacy may be compromised when outsourcing this recommendation service to the cloud.

To address the aforementioned privacy problems, some privacy-preserving location-based recommendation schemes [1,9–20] have been put forth. These schemes can be divided into cryptography-based schemes [9–20] and k-anonymity or differential privacy-based schemes [1,4,21,22]. Although cryptography-based schemes [9–20] achieve stronger privacy, but they are computationally inefficient for the query users. Therefore, these schemes are impractical in real-world settings because query users are usually the resource-constrained devices. Schemes based on non-cryptography techniques [1,4,21,22] need trade-off between privacy and accuracy. They only provide weaker privacy if the query results with higher accuracy are required. Meanwhile, for schemes based on non-cryptography techniques, schemes for proximity testing [19,20] or scheme [9] transforming real locations into pseudolocations, it is impossible to return accurate recommendation results.

Additionally, most of them only support single-attribute queries in LBS. Therefore, due to privacy, accuracy, and functionality concerns, these schemes are not practical and applicable for use in real-world applications. In this paper, we propose a fully privacy-preserving location recommendation scheme in outsourced environments, where the privacy of the service provider and users is well protected, and the data access pattern is also hidden from the cloud. Additionally, actual location coordination is used, multi-attribute queries are supported, and accurate results are returned. Specifically, the contributions of our paper are threefold.

• First, we introduce a new and specifical privacy-preserving scenario in the location-based services and propose a secure equal test (SET) protocol based on the Paillier cryptosystem and construct a secure unequal to zero test protocols based on our SET protocol. These two protocols can be used to check whether a restaurant matches the user's preference and whether it satisfies the recommendation condition in a privacy-preserving way, respectively.

• Second, based on proposed secure protocols, we propose a framework to implement the fully privacy-preserving location-based recommendations in outsourced environments. Compared with some existing schemes, our scheme is fully privacy-preserving, supports multiple attributes, has lower computation costs for query users, returns accurate results as in the plaintext domain.

• Third, we analyze the security of our proposed protocols and scheme in the semi-honest model. We evaluate our scheme by conducting extensive experiments. The experimental results show that it is practical in real-world LBS.

The remainder of our paper is organized as follows. In Section 2, we review some existing privacy-preserving schemes related to location-based services. Section 3 formalizes our system model, threat model, and design goal. In Section 4, we introduce some necessary preliminaries used in our scheme. After that, we propose some new building blocks in Section 5. We present our privacy-preserving location recommendation scheme in Section 6. Next, we show the security analysis in Section 7 and report the performance evaluation in Section 8. Finally, we conclude our paper in Section 9.

2. Related works

In this section, we introduce some privacy-preserving schemes used in location-based services. These schemes can be classified as cryptography-based schemes and non-cryptography-based schemes.

• *Cryptography-based approaches*: Both schemes in [9,10] focus on the k-nearest neighbor query in LBS. Based on the Paillier homomorphic encryption, Lien et al. [9] design a private circular query protocol to solve privacy and accuracy issues in LBS. Own to the encryption technique, the privacy of the user's location is protected against the LBS provider during the query process. Guan et al. [10] propose a novel oblivious location-based kNN query scheme based on the modified Paillier cryptosystem. In this scheme, any two queries cannot be linked whenever a user queries twice at the same location. Schemes [11,13,14] mainly solve privacy problems when specific recommendation algorithms are used in real-world LBS. In these schemes, each POI has a historical and numerical rating. Badsha et al. [11] utilize the weighted slope one predictor algorithm to generate user-personalized location recommendations. Compared with previous works, they incorporate users' friendship networks with location preferences. The privacy of LBS provider and users is well protected via the Paillier encryption. However, this scheme generates the recommended results only according to the rating of each location, and no actual location coordinate is involved. Literature [13,14] uses the collaborative filtering (CF) algorithm to provide the prediction services. Ma et al. [13] propose a novel framework to protect the user's sensitive information. In this framework, all historical ratings are encrypted, and the similarities of POIs are computed in ciphertext domain. Based on the Paillier, commutative, and comparable encryption, it generates the recommendation results in a privacy-preserving way. The goal of the scheme [14] is to use CF-based technology to predict the quality of service (QoS) for unobserved Web services based on past QoS experiences and locations of users. To solve the privacy problems, the authors develop a privacy-preserving protocol to predict missing QoS values via the Boneh Goh Nissim (BGN) cryptosystem. Xu et al. [12] propose a privacy-preserving route matching scheme for carpooling services. Based on a Goldwasser-Micali-based equality determination algorithm, the authors construct an accurate similarity computation algorithm. This algorithm allows users to get accurate carpooling results over ciphertexts without revealing the privacy of users and routes. Schemes [15–18] focus on searching encrypted resources in outsourced location-based services. Based on the improved 2DNF cryptosystem, Zhu et al. [15] propose an efficient and privacy-preserving polygons spatial query framework in LBSs. Users can perform any polygon range query to obtain accurate LBS results without revealing their query data to the LBS provider and the cloud server. Li et al. [16] design a novel privacy-preserving LBS search scheme in outsourced environments. Users can acquire accurate LBS results by constructing a query model without divulging their location information and queries to the LBS provider and the cloud server. Based on attribute-based encryption, linear encryption, and RSA encryption, Huang et al. [17] present a privacy-preserving spatio-temporal keyword search framework over outsourced encrypted LBS data. This framework allows users to request LBS queries with spatial range, time interval, and Boolean keyword expression. Li et al. [18] propose the first predicate-only encryption scheme for the inner product range. Based on that, they design an efficient and privacy-preserving spatial range query scheme. To reduce query latency, the authors also construct a privacy-preserving tree index structure. The goals of schemes [19,20] are to perform grid-based proximity testing with privacy preservation in LBS. Narayanan et al. [20] first reduce proximity testing to equality testing and utilize private equality testing to achieve privacy-preserving testing for proximity in LBS. Based on various secure computation protocols, Järvinen et al. [19] mainly design, implement and evaluate several privacy-preserving location proximity testing algorithms.

• *Non-cryptography-based approaches*: Differential privacy and k-anonymity are also privacy-preserving techniques. Scheme [1] applies k-anonymity to make users and the LBS provider perform the mutual transformation between an actual location and a pseudo location via the spatial transformation parameters with a periodical update. Any knowledge related to users' real location is not learned by the anonymizer without knowing the transforming parameters, so users can obtain POI in a privacy-preserving manner. Schemes [4,21,22] utilize differential privacy to deal with the privacy issues in LBS. Huo et al. [4] propose a geographical location privacy-preserving algorithm to achieve $\langle r, h\rangle$-privacy and a privacy-preserving friend relationship algorithm by adding laplacian distributed noise. With the aid of two proposed privacy-preserving algorithms, the privacy leakage for users is prevented. Chen et al. [21] propose a novel privacy-preserving POI recommendation framework. This recommendation framework consists of a linear model and the feature interaction model. Users' data are kept on their own sides to protect their privacy. For the privacy of the model, users save the linear models locally and the final model is learned by a secure decentralized gradient descent protocol, while the feature interaction model is kept by the recommender. Gao et al. [22] are interested in privacy leakage of users' history footprint when using the CF-based method as the recommendation algorithm. Authors apply the geo-indistinguishability to perturb users' location to achieve ξ_1-differential privacy. In order to protect the privacy of users' history location data, they first collect them and generate a category histogram. Then, they perturb the aggregated histogram to achieve ξ_2-differential privacy. However, the security of schemes based on these techniques is weaker than that based on cryptographic techniques. Meanwhile, due to adding noise or fake location data, these approaches degrade the accuracy of the recommendation results.

3. Models and design goal

3.1. System model

In our system model, we consider a privacy-preserving location recommendation model in outsourced environments, which involves a dining service provider (or data owner), a cloud with two servers (S_A and S_B), and multiple query users as shown in Fig. 2. Our system consists of five steps: ① The service provider generates a key pair and ② encrypts and outsources its data; ③ Query users encrypt and outsource their data; ④ The cloud computes the recommended results; ⑤ Query users obtain the query results.

• *Service Provider*: The service provider (or dining service provider (DSP)) is responsible for generating a key pair (pk, sk) of the Paillier cryptosystem and distributing (pk, sk) to S_A and pk to other entities.

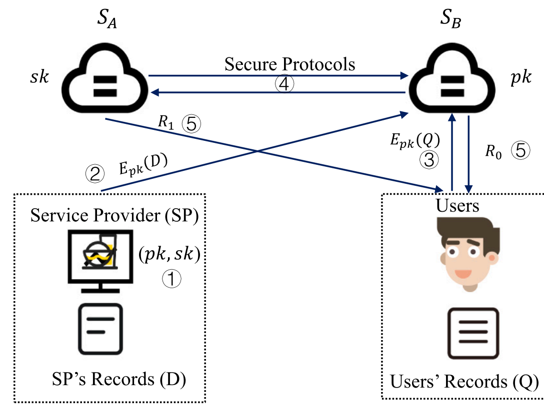


Fig. 2. System model.

In addition, it also has a database with a large number of restaurants' records $D = \{D_1, D_2, \ldots, D_m\}$, where each record consists of the restaurant's unique identity (ID), coordinate (Co), cuisine (Cu), and average price per head (Av), so the ith restaurant record can be denoted by $D_i = \{ID_i, Co_i, Cu_i, Av_i\}$. Here, we assume that all values in the restaurant record D_i are integers. In order to gain benefits, the service provider is willing to provide a location-based recommendation service and recommend the restaurants that match users' preferences to users. However, the service provider's computing power and storage capacity are constrained, so it outsources its database D to the cloud and makes advantage of the cloud to offer users location-based recommendation services. In order to protect the privacy of the restaurant records, the service provider encrypts its database D before uploading it to the cloud.

• *Query Users*: The system has many query users $U = \{U_1, U_2, \ldots, U_t\}$. Users request queries with their records Q and obtain corresponding query results R_1 and R_0 from S_A and S_B, respectively. In this system, we assume that all users must register with the dining service provider. Meanwhile, the cloud will verify whether queries are from registered users or unregistered users. However, in this paper, we ignore the verification procedure and mainly focus on privacy protection. In practice, we can realize that mechanism using cryptographic techniques, such as the digital signature scheme.

• *Cloud with Two Servers (S_A and S_B)*: The cloud has two servers that both have powerful computing capability and large storage space. The cloud is used to store the encrypted restaurant records and respond query users with the recommended restaurants matching users' preferences. For example, these restaurants are close to one of users' visited restaurants within a predefined distance threshold, match users' cuisines, or have acceptable average prices for users.

3.2. Threat model

In our security model, we assume that the dining service provider and users are all trusted. It implies that they will honestly outsource their restaurant records and request location recommendation queries, respectively. However, two servers (S_A and S_B) are semi-honest (or honest-but-curious). Namely, they follow the protocol with their correct inputs but may be curious to derive some sensitive information during the execution of the protocol. In addition, we assume that there is no collusion between S_A and S_B. Meanwhile, they do not collude with other entities. This is a reasonable assumption in real-world settings. For instance, S_A and S_B may be from two different companies. In order to maintain a good reputation for commercial interests, two companies will not collude with each other. Such a two-server model has also gained popularity in existing privacy-preserving works [23–26].

In such threat model, we consider an active adversary to recover the plaintext of the outsourced data, the query data, and the query results.

Table 1
Notations.

Notations	Definition
$\lvert N\rvert$	The bit-length of number N
$\lvert\lvert X-Y\rvert\rvert^2$	Squared euclidean distance between vectors X and Y
$[\![m]\!]$	The ciphertext of message m for the Paillier algorithm
$abs(x)$	The absolute value of x
$a \vee b$	The logical OR of boolean values a and b
$E_{pk}(m)$	Encrypt m with pk for the Paillier algorithm
$D_{sk}(c)$	Decrypt c with sk for the Paillier algorithm
Add_{he}	Homomorphic addition
$SMul_{he}$	Homomorphic scalar multiplication
SM	Secure multiplication
$SSED$	Secure squared euclidean distance
SC	Secure comparison
SET	Secure equal test
$SUEZ$	Secure unequal to zero test

The adversary has the two types of capabilities: ① eavesdrops all the communications to get the exchanging data and ② comprises one of two servers at most.

3.3. Design goal

Our design goal is to develop a privacy-preserving location recommendation scheme in outsourced environments. The proposed scheme achieves the following four objectives:

• *Privacy Protection*: The privacy of restaurant records from the dining service provider and queries from users is protected against two cloud servers (S_A and S_B). In addition, users only acquire the recommended restaurant records matching the users' preferences without knowing other restaurant records. The data access pattern also is hidden from two servers. That is, our scheme is fully privacy-preserving.

• *Query Accuracy*: The cloud should return accurate query results for users. It means that query results generated under the ciphertext domain should be the same as that in the plaintext domain.

• *Query with Multiple Attributes*: Our scheme allows users to request queries with multiple attributes. It makes the recommended results more satisfying for users.

• *Efficiency*: The dining service provider has a large number of restaurant records, so its computation cost and storage burden must be reduced. Meanwhile, we must reduce the computation costs of query users because they are usually the resource-constrained devices.

4. Preliminaries

In this section, we first define our location-based recommendation query in LBS. After that, we introduce the Paillier cryptosystem which supports the homomorphic addition and the homomorphic scalar multiplication operators. Finally, we describe some existing computation primitives based on the Paillier algorithm. Table 1 gives some notations and their corresponding definitions in our paper.

4.1. Location-based recommendation query

The goal of our location-based recommendation query (LBRQ) is to find the best location matching the recommendation condition required by users. In our paper, we focus on the restaurant recommendation problem in LBS. In this scenario, the dining server provider recommends the restaurants to users based on the similarity between the restaurants. That is to say, users request queries with their visited restaurants, and the dining server provider recommends restaurants matching their preferences to them.

As in Section 6, the dining service provider has a set of restaurant records $D = \{D_1, D_2, \dots, D_m\}$ and each restaurant record is denoted by $D_i = \{ID_i, Co_i, Cu_i, Av_i\}(1 \le i \le m)$, where ID_i is the ith restaurant ID or name, Co_i is the D_i's location coordinate, Cu_i is the D_i's cuisine, and Av_i is the D_i's average price. Likewise, the visited restaurants' coordinates and the favorite cuisines for the user are denoted by $C = \{c_1, c_2, \dots, c_n\}$ and $S = \{s_1, s_2, \dots, s_t\}$, respectively. Let A_P and R_C be the acceptable average price per head and the recommendation condition, respectively. We respectively represent the distance threshold $disTh$ and the price threshold $priTh$. The dining service provider will recommend the restaurant D_i to the querying user if it satisfies the recommendation condition R_C. In other words, the number of matched attributes between the user's query and the restaurant D_i is greater than or equal to R_C. The location-based recommendation query can be defined as follows.

Definition 1 (*LBRQ*)**.** Given a set of restaurant records D, a set of restaurants' coordinates C, a set of favorite cuisines S, a distance threshold $disTh$, a price threshold $priTh$, and the recommendation condition R_C, LBRQ is to find a set of restaurants $R \in D$ as the recommended objects for a user,

$$\alpha_k = \alpha_{k,1} \vee \alpha_{k,2} \vee \cdots \vee \alpha_{k,n},$$

$$\beta_k = \beta_{k,1} \vee \beta_{k,2} \vee \cdots \vee \beta_{k,t},$$

$$\lambda_k = (abs(Av_k - A_p) \le priTh),$$

$$R = \{D_k | (\alpha_k + \beta_k + \lambda_k) \ge R_C\}$$

where $\alpha_{k,i} = (\|Co_k - c_i\|^2 \le disTh^2)$, $\beta_{k,j} = (Cu_k \stackrel{?}{=} s_j)$, $1 \le k \le m$, $1 \le i \le n$, and $1 \le j \le t$.

Notice that $Cu_k \stackrel{?}{=} s_j$ is an boolean expression and denotes whether the cuisine of the kth restaurant D_k matches the user's favorite cuisine s_j. If they match, the result is 1; Otherwise, it is 0. For $\alpha_k \in \{0, 1\}$, $\beta_k \in \{0, 1\}$, and $\lambda_k \in \{0, 1\}$, they represent whether the dining service provider's restaurant D_k satisfies the user's corresponding preferences. That is, α_k indicates whether the Euclidean distance between one of the user's visited restaurants and the restaurant D_k is no greater than a predefined distance threshold $disTh$, β_k indicates whether one of the user's favorite cuisines matches the cuisine of the restaurant D_k, and λ_k indicates whether the difference of the average price between the restaurant D_k and the user is within a predefined price threshold $priTh$.

4.2. Paillier cryptosystem

The Paillier cryptosystem [27], named after and invented by Pascal Paillier, is a probabilistic public-key encryption scheme based on the decisional composite residuosity assumption. This cryptosystem supports homomorphic addition and homomorphic scalar multiplication properties. In the remainder of our paper, the encryption and decryption functions for the Paillier algorithm are respectively denoted by E_{pk} and D_{sk}, where pk and sk represent the public key and the private key, respectively. Given two messages a and b, Paillier's homomorphic properties can be expressed as follows:

• *Homomorphic Addition* (Add_{he}):

$$D_{sk}(Add_{he}([\![a]\!], [\![b]\!])) = a + b$$

• *Homomorphic Scalar Multiplication* ($SMul_{he}$):

$$D_{sk}(SMul_{he}([\![a]\!], b)) = a * b$$

The Paillier cryptosystem is an indistinguishable encryption algorithm [28]. In other words, it is infeasible for an adversary to distinguish the encryptions of two plaintexts.

4.3. Secure computation primitives

Based on the homomorphic properties of the Paillier, many secure primitives were developed in related literature [23,29–31] and our previous work [32]. Here, we introduce some of them used in our scheme. In these primitives, we always assume that S_A has the key pair (pk, sk) of the Paillier cryptosystem, and S_B only has the public key pk. Additionally, their inputs and outputs are held by S_B in an encrypted format, while S_B knows nothing about them.

• *Secure Multiplication (SM) Protocol*: In this protocol, S_A inputs nothing, and S_B inputs two ciphertexts $[\![m_0]\!]$ and $[\![m_1]\!]$ and outputs $[\![m_0 * m_1]\!]$. During the execution of this protocol, S_B and S_A do not know any information about a and b, and only S_B obtains the output $[\![m_0 * m_1]\!]$.

• *Secure Squared Euclidean Distance ($SSED$) Protocol*: In this protocol, S_A inputs nothing, and S_B inputs two encrypted vectors $[\![X]\!]$ and $[\![Y]\!]$ and outputs $[\![\|X-Y\|^2]\!]$. Here X and Y are two dimensional vectors, namely $[\![X]\!] = ([\![x_1]\!], [\![x_2]\!])$ and $[\![Y]\!] = ([\![y_1]\!], [\![y_2]\!])$. This protocol can be constructed based on the SM protocol. Similar to the SM protocol, the output $[\![\|X-Y\|^2]\!]$ is only known to S_B.

• *Secure Comparison (SC) Protocol*: In this protocol, S_A inputs nothing, and S_B inputs two encrypted values $[\![x]\!]$ and $[\![y]\!]$ and outputs $[\![1]\!]$ (if $x \geq y$) or $[\![0]\!]$ (if $x < y$). Similarly, the output of this protocol is acquired only by S_B.

5. Building blocks

In this section, we present a secure equal test (SET) algorithm and a secure unequal to zero test ($SUEZ$) algorithm as the building blocks in our scheme. The latter can be constructed from the former easily and described at the end of this section.

Our SET algorithm is built upon the Paillier cryptosystem and performed by S_A and S_B, where S_A knows sk and S_B has $[\![x]\!]$ and $[\![y]\!]$. It is employed to test whether $[\![x]\!]$ is equal to $[\![y]\!]$. If $x = y$, it outputs $[\![1]\!]$; Otherwise, it outputs $[\![0]\!]$. During the execution of this algorithm, S_B holds the inputs and outputs in an encrypted format, while S_A only acquires the random intermediate values. Algorithm 1 describes our SET protocol.

Algorithm 1 Secure Equal Test (SET)

Input: S_A has a key pair (sk, pk), and S_B only has pk. S_B has two encrypted values $[\![x]\!]$ and $[\![y]\!]$. Note that x and y are at most v bits, and the message space of the Paillier algorithm is $\mathbb{Z}_N$, where set $n = |N|$.

Output: If $x = y$, S_B gets $[\![1]\!]$; Otherwise, S_B gets $[\![0]\!]$.

1: S_B: Subtracts $[\![x]\!]$ by $[\![y]\!]$: $[\![d]\!] = Add_{he}([\![x]\!], SMul_{he}([\![x]\!], N-1))$.

2: S_A&S_B: Computes the square of d: $[\![h]\!] = SM([\![d]\!], [\![d]\!])$.

3: S_B: Picks a random value α over $\mathbb{Z}_2$, two random integers r_1 and r_2, where $r_1 > r_2$ and $|r_1| = |r_2| = (n/2 - 2v - 1)$. If $\alpha = 0$, calculates $[\![t]\!] = Add_{he}(SMul_{he}([\![h]\!], r_1), SMul_{he}([\![r_2]\!], N-1))$. Otherwise, $[\![t]\!] = Add_{he}(r_2, SMul_{he}(SMul_{he}([\![h]\!], r_1), N-1))$. Sends $[\![t]\!]$ to S_A.

4: S_A: Decrypts $[\![t]\!]$ and compares t with $N/2$. If $t < N/2$, sets $\beta = 0$; Otherwise, sets $\beta = 1$. Encrypts β and sends $[\![\beta]\!]$ to S_B.

5: S_B: If $\alpha = 0$, outputs $[\![\beta]\!]$; Otherwise outputs $[\![1-\beta]\!]$.

S_B first computes the subtraction of $[\![x]\!]$ and $[\![y]\!]$ via the additive homomorphic property of the Paillier cryptosystem, so it gets $[\![d]\!] = [\![x-y]\!]$. Then, S_A and S_B jointly run the SM protocol, and S_B acquires its output $[\![h]\!] = [\![d^2]\!]$. After that, S_B selects a random value $\alpha \in \{0,1\}$ and two random values r_1 and r_2 with special settings. If $\alpha = 0$, S_B sets $[\![t]\!] = [\![r_1(x-y)^2 - r_2]\!]$; Otherwise, S_B sets $[\![t]\!] = [\![r_2 - r_1(x-y)^2]\!]$. At the end of this step, S_B sends $[\![t]\!]$ to S_A. After receiving $[\![t]\!]$, S_A decrypts it with sk and compares t with $N/2$. If $t < N/2$, S_A sets $\beta = 0$; Otherwise, S_A sets $\beta = 1$. Then, S_A encrypts β and sends its ciphertext to S_B. S_B computes the final equality test result $[\![z]\!]$ according to its own α and the received β. If $\alpha = 0$, S_B sets $[\![z]\!] = [\![\beta]\!]$; Otherwise, S_B sets $[\![z]\!] = [\![1-\beta]\!]$.

Correctness. Due to $r_1 > r_2$ and their special settings, we can know that $r_2 < r_1(x-y)^2 < N/2$ always holds. Here, we first prove the correctness of Algorithm 1 when $\alpha = 0$. In this case, if $x = y$, the sign of $(r_1(x-y)^2 - r_2)$ is negative; Otherwise, that expression is positive. When considering this operation over $\mathbb{Z}_N$, it means that $(r_1(x-y)^2 - r_2) > N/2$ if $x = y$, or $(r_1(x-y)^2 - r_2) < N/2$ if $x \neq y$. We set $\beta = 0$ and $\beta = 1$ in these two cases, respectively. The key observation is that the final equality test result z is consistent with β, namely $z = \beta$ when $\alpha = 0$. Similarly, when $\alpha = 1$, we can observe that z is equal to the NOT of β, namely $z = 1 - \beta$. Therefore, the correctness of our SET protocol is verified.

Remark. The $SUEZ$ algorithm is to determine whether $[\![x]\!]$ is unequal to zero. If $x \neq 0$, it outputs 1; Otherwise, it outputs 0. Our $SUEZ$ protocol can be constructed by first running the SET protocol with inputs $[\![x]\!]$ and $[\![0]\!]$, and then computing the NOT of its output without any additional interaction.

6. Our proposed scheme

In this section, we present our privacy-preserving location recommendation scheme. Our scheme consists of the system initialization phase and the location recommendation phase.

6.1. System initialization

In the system initialization phase, the dining service provider first generates a key pair (pk, sk) of the Paillier cryptosystem and distributes them to other entities in our system. Then, to protect the privacy of the restaurant records, the dining service provider encrypts each entry of them before outsourcing them to the cloud server. The details of this phase are as follows:

(1) The dining service provider first generates a key pair (pk, sk). Then, it gives (pk, sk) to S_A and pk to S_B and the registered users.

(2) The dining service provider encrypts each restaurant record $D_j = \{ID_j, Co_j, Cu_j, Av_j\}$ with the public key pk and sends $[\![D_j]\!] = \{[\![ID_j]\!], [\![Co_j]\!], [\![Cu_j]\!], [\![Av_j]\!]\}$ to S_B, where $0 \leq j \leq m$ and m is the maximum number of restaurants.

Notice that in our system, all messages between any two entities are transferred via the secure channel established by the cryptographic protocols (e.g., SSL/TLS). Additionally, we encode each entry of the restaurant record as an integer. For instance, we can compute the hash value of the cuisine "Chinese" using the hash function SHA-1. Then, we module this value with 2^{32} and regard its result as the encoded integer.

6.2. Location recommendation

In the location recommendation phase, the query user first encrypts his queries (restaurants' coordinates, cuisines, and average prices) and uploads them to S_B. Second, S_A and S_B generate the recommended results (or restaurants' information) in the encrypted domain by performing some secure computation protocols interactively. Then, the two cloud servers partly return the generated results to the query user. Finally, the query user performs some simple computations locally and recovers the real restaurant information matching his preferences. The process of this phase is as follows:

(1) The query user encrypts the visited restaurants' coordinates $C = \{c_1, c_2, \ldots, c_n\}$, the favorite cuisines $S = \{s_1, s_2, \ldots, s_t\}$, and the acceptable average price A_P with pk. Meanwhile, the query user encrypts a distance threshold $disTh$ and a price threshold $priTh$ used for recommendation. He also encrypts a value $R_C \in \{1,2,3\}$, named recommendation condition. Given a restaurant D_j, it will be recommended to the query user if the number of its features matching the user's preferences

D_j	ID_j	Co_j	Cu_j	Av_j
D_1	10112	(12,90)	British	58
D_2	90054	(13,28)	Chinese	55
D_3	32789	(77,96)	Chinese	78
D_4	87103	(89,95)	Indian	92

$D_j = \{ID_j, Co_j, Cu_j, Av_j\}$

The dining service provoder's records

c_i	s_k	A_P	$disTh$	$priTh$	R_C
$c_1 = (17,30)$	$s_1 =$ British	75	100	25	3
$c_2 = (92,101)$	$s_2 =$ Chinese				

$Q = \{C, S, A_P, disTh, PriTh, R_C\}$,
$C = \{c_1, c_2\}$,
$S = \{s_1, s_2\}$

(1) The user requests a query Q

$d_{i,j}$	$x_{i,j}$	X_i	U_i	$y_{k,j}$	Y_j	V_j	w_j	W_j	H_j	F_j
$d_{1,1} = 3625$	$x_{1,1} = 0$	$X_1 = 0$	$U_1 = 0$	$y_{1,1} = 1$	$Y_1 = 1$	$V_1 = 1$	$w_1 = 289$	$W_1 = 0$	$H_1 = 1$	$F_1 = 0$
$d_{1,2} = 6521$	$x_{1,2} = 0$			$y_{1,2} = 0$						
$d_{2,1} = 20$	$x_{2,1} = 1$	$X_2 = 1$	$U_2 = 1$	$y_{2,1} = 0$	$Y_2 = 1$	$V_2 = 1$	$w_2 = 400$	$W_2 = 0$	$H_2 = 2$	$F_2 = 0$
$d_{2,2} = 11570$	$x_{2,2} = 0$			$y_{2,2} = 1$						
$d_{3,1} = 9256$	$x_{3,1} = 0$	$X_3 = 1$	$U_3 = 1$	$y_{3,1} = 0$	$Y_3 = 1$	$V_3 = 1$	$w_3 = 9$	$W_3 = 1$	$H_3 = 3$	$F_3 = 1$
$d_{3,2} = 50$	$x_{3,2} = 1$			$y_{3,2} = 1$						
$d_{4,1} = 1154$	$x_{4,1} = 0$	$X_4 = 0$	$U_4 = 0$	$y_{4,1} = 0$	$Y_4 = 0$	$V_4 = 0$	$w_4 = 289$	$W_4 = 0$	$H_4 = 0$	$F_4 = 0$
$d_{4,2} = 4804$	$x_{4,2} = 0$			$y_{4,2} = 0$						
Step 1	*Step* 2	*Step* 3	*Step* 4	*Step* 5	*Step* 6	*Step* 7	*Step* 9	*Step* 10	*Step* 11	*Step* 12

R_j^1	$R_j^{ID'}$	$R_j^{Co'}$	$R_j^{Cu'}$	$R_j^{Av'}$
R_1^1	41095	66	71	82
R_2^1	31872	52	63	89
R_3^1	45679	(111,186)	155	123
R_4^1	78912	(78,65)	93	37

R_j^0	R_j^{ID}	R_j^{Co}	R_j^{Cu}	R_j^{AV}
R_1^0	41095	66	71	82
R_2^0	31872	52	63	89
R_3^0	12890	(34,90)	56	45
R_4^0	78912	(78,65)	93	37

Step 14 and *Step* 15

D_j'	ID_j'	Co_j'	Cu_j'	Av_j'
D_1'	0	0	0	0
D_2'	0	0	0	0
D_3'	32789	(77,96)	Chinese	78
D_4'	0	0	0	0

Encode "Chinese" to an integer 99. *Step* 13

(2) Two servers compute the recommended result step by step

$D_j^{\sim}$	$ID_j^{\sim}$	$Co_j^{\sim}$	$Cu_j^{\sim}$	$Av_j^{\sim}$
$D_1^{\sim}$	0	0	0	0
$D_2^{\sim}$	0	0	0	0
$D_3^{\sim}$	32789	(77,96)	Chinese	78
$D_4^{\sim}$	0	0	0	0

(3) The user recovers the recommended result

Fig. 3. Numerical example for proposed scheme.

is greater than or equal to R_C. The query user aggregates these encrypted data as a query $[\![Q]\!] = \{[\![C]\!], [\![S]\!], [\![A_P]\!], [\![disTh]\!], [\![priTh]\!], [\![R_C]\!]\}$ and sends $[\![Q]\!]$ to S_B.

We assume that the price threshold and the distance threshold are thought to be the squares of their respective actual values.

(2) After receiving a query $[\![Q]\!]$ from the query user, S_A and S_B jointly compute the recommended restaurants that best match the query user's preferences as follows.

Step 1: S_A and S_B run the $SSED$ protocol to compute the squared euclidean distance $d_{i,j}$ between the restaurant with the coordinate c_i and the restaurant D_j with the coordinate Co_j, where $1 \le i \le n$ and $1 \le j \le m$.

$$[\![d_{i,j}]\!] = SSED([\![c_i]\!], [\![Co_j]\!])$$

Step 2: S_A and S_B compare $[\![d_{i,j}]\!]$ and $[\![disTh]\!]$ via the SC protocol and determine whether the distance from the restaurant D_j to the restaurant with the coordinate c_i is within the distance threshold $disTh$.

$$[\![x_{i,j}]\!] = SC([\![disTh]\!], [\![d_{i,j}]\!])$$

Step 3: S_B locally accumulates $[\![x_{1,j}]\!], \dots, [\![x_{n,j}]\!]$ via the additive homomorphic property.

$$[\![X_j]\!] = [\![x_{1,j} + \dots + x_{n,j}]\!] = Add_{he}(Add_{he}([\![x_{1,j}]\!], \dots), [\![x_{n,j}]\!])$$

Step 4: S_A and S_B test whether $[\![X_j]\!]$ is unequal to zero via the $SUEZ$ protocol. If it is true ($U_j = 1$), it means that the restaurant D_j is close to one or more restaurants visited by the query user within the distance threshold $disTh$.

$$[\![U_j]\!] = SUEZ([\![X_j]\!])$$

Step 5: S_A and S_B run the SET protocol to determine whether the cuisine of the restaurant D_j matches the query user's favorite cuisine s_k, where $1 \le k \le t$.

$$[\![y_{k,j}]\!] = SET([\![s_k]\!], [\![Cu_j]\!])$$

Step 6: S_B locally accumulates $[\![y_{1,j}]\!], \dots, [\![y_{t,j}]\!]$ via the additive homomorphic property.

$$[\![Y_j]\!] = [\![y_{1,j} + \dots + y_{t,j}]\!] = Add_{he}(Add_{he}([\![y_{1,j}]\!], \dots), [\![y_{t,j}]\!])$$

Step 7: S_A and S_B test whether $[\![Y_j]\!]$ is unequal to zero via the $SUEZ$ protocol. If it is true ($V_j = 1$), it implies that the cuisine of the restaurant D_j matches one of the user's favorite cuisines.

$$[\![V_j]\!] = SUEZ([\![Y_j]\!])$$

Step 8: Based on the homomorphic properties, S_B locally computes the difference between the average price per head $[\![Av_j]\!]$ of the restaurant D_j and the acceptable price $[\![A_P]\!]$ of the query user.

$$[\![z_j]\!] = [\![Av_j - A_P]\!] = Add_{he}([\![Av_j]\!], SMul_{he}([\![A_P]\!], N-1))$$

Step 9: S_A and S_B compute the square of $[\![z_j]\!]$ using the SM protocol.

$$[\![w_j]\!] = [\![z_j^2]\!] = SM([\![z_j]\!], [\![z_j]\!])$$

Step 10: S_A and S_B compare $[\![priTh]\!]$ and $[\![w_j]\!]$ by performing the SC protocol. If the comparison result is true ($W_j = 1$), it indicates that the average price per head $[\![Av_j]\!]$ of the restaurant D_j's differs from the acceptable price $[\![A_P]\!]$ of the query user by no more than the price threshold $[\![priTh]\!]$.

$$[\![W_j]\!] = SC([\![priTh]\!], [\![w_j]\!])$$

Step 11: S_B adds $[\![U_j]\!]$, $[\![V_j]\!]$ and $[\![W_j]\!]$ locally based on the additive homomorphic property.

$$[\![H_j]\!] = Add_{he}(Add_{he}([\![U_j]\!], [\![V_j]\!]), [\![W_j]\!])$$

Step 12: S_A and S_B run the SC protocol to compare $[\![H_j]\!]$ and $[\![R_C]\!]$ and determine whether the restaurant D_j matches the recommendation condition $[\![R_C]\!]$. If $F_j = 1$, the restaurant D_j should be recommended to the query user.

$$[\![F_j]\!] = SC([\![H_j]\!], [\![R_C]\!])$$

Step 13: S_A and S_B generate the recommended restaurant record R_j via the SM protocol, so they set $R_j = \{ID_j, Co_j, Cu_j, Av_j\}$ ($F_j = 1$) or $R_j = \{0, 0, 0, 0\}$ ($F_j = 0$).

$$[\![ID'_j]\!] = SM([\![ID_j]\!], [\![F_j]\!])$$

$$[\![Co'_j]\!] = SM([\![Co_j]\!], [\![F_j]\!])$$

$$[\![Cu'_j]\!] = SM([\![Cu_j]\!], [\![F_j]\!])$$

$$[\![Av'_j]\!] = SM([\![Av_j]\!], [\![F_j]\!])$$

S_B sets $[\![R_j]\!] = \{[\![ID'_j]\!], [\![Co'_j]\!], [\![Cu'_j]\!], [\![Av'_j]\!]\}$. Notice that Co_j is a two-dimension coordinate (x, y), so $SM([\![Co_j]\!], [\![F_j]\!])$ is equivalent to $SM([\![Co_j.x]\!], [\![F_j]\!])$ and $SM([\![Co_j.y]\!], [\![F_j]\!])$.

Step 14: S_B applies the homomorphic property to randomize $[\![R_j]\!]$ with five random values R_j^{ID}, $R_j^{Co.x}$, $R_j^{Co.y}$, R_j^{Cu}, and R_j^{Av} from $\mathbb{Z}_N$ before sending it to S_A.

$$[\![ID^*_j]\!] = Add_{he}([\![ID'_j]\!], [\![R_j^{ID}]\!])$$

$$[\![Co^*_j.x]\!] = Add_{he}([\![Co'_j.x]\!], [\![R_j^{Co.x}]\!])$$

$$[\![Co^*_j.y]\!] = Add_{he}([\![Co'_j.y]\!], [\![R_j^{Co.y}]\!])$$

$$[\![Cu^*_j]\!] = Add_{he}([\![Cu'_j]\!], [\![R_j^{Cu}]\!])$$

$$[\![Av^*_j]\!] = Add_{he}([\![Av'_j]\!], [\![R_j^{Av}]\!])$$

S_B sets $[\![R^*_j]\!] = \{[\![ID^*_j]\!], [\![Co^*_j]\!], [\![Cu^*_j]\!], [\![Av^*_j]\!]\}$, where $[\![Co^*_j]\!] = ([\![Co^*_j.x]\!], [\![Co^*_j.y]\!])$, and $R_j^0 = \{R_j^{ID}, R_j^{Co}, R_j^{Cu}, R_j^{Av}\}$, where $R_j^{Co} = (R_j^{Co.x}, R_j^{Co.y})$. S_B sends $[\![R^*_j]\!]$ to S_A and R_j^0 to the user, respectively.

Step 15: After receiving $[\![R^*_j]\!]$, S_A decrypts it with the private key sk. S_A cannot learn any information of the recommended restaurant record R_j from R^*_j because each entry of R^*_j is a random value.

$$R_j^{ID'} = R_j^{ID} + ID_j * F_j = D_{sk}([\![ID^*_j]\!])$$

$$R_j^{Co'.x} = R_j^{Co.x} + Co_j.x * F_j = D_{sk}([\![Co^*_j.x]\!])$$

$$R_j^{Co'.y} = R_j^{Co.y} + Co_j.y * F_j = D_{sk}([\![Co^*_j.y]\!])$$

$$R_j^{Cu'} = R_j^{Cu} + ID_j * F_j = D_{sk}([\![Cu^*_j]\!])$$

$$R_j^{Av'} = R_j^{Av} + ID_j * F_j = D_{sk}([\![Av^*_j]\!])$$

S_A sets $R_j^1 = \{R_j^{ID'}, R_j^{Co'}, R_j^{Cu'}, R_j^{Av'}\}$, where $R_j^{Co'} = (R_j^{Co.x'}, R_j^{Co.y'})$. S_A sends R_j^1 to the query user.

(3) After receiving R_j^0 and R_j^1 from S_B and S_A respectively, the query user subtracts R_j^1 by R_j^0 to recover the jth restaurant record.

$$ID_j^{\sim} = R_j^{ID'} - R_j^{ID} = ID_j * F_j$$

$$Co_j^{\sim}.x = R_j^{Co'.x} - R_j^{Co.x} = Co_j.x * F_j$$

$$Co_j^{\sim}.y = R_j^{Co'.y} - R_j^{Co.y} = Co_j.y * F_j$$

$$Cu_j^{\sim} = R_j^{Cu'} - R_j^{Cu} = Cu_j * F_j$$

$$Av_j^{\sim} = R_j^{Av'} - R_j^{Av} = Av_j * F_j$$

The query user sets $D_j^{\sim} = \{ID_j^{\sim}, Co_j^{\sim}, Cu_j^{\sim}, Av_j^{\sim}\}$, where $Co_j^{\sim} = (Co_j^{\sim}.x, Co_j^{\sim}.y)$.

In the expressions listed above, F_j indicates whether the restaurant D_j satisfies the user's query condition. If $F_j = 1$, each entry of $D_j^{\sim}$ is equal to the corresponding entry of the restaurant D_j; Otherwise, each entry in $D_j^{\sim}$ is zero.

Remark. To clarify the processes of our proposed scheme, we give a numerical example in Fig. 3. The dining service provider's restaurant records and the user's query data are from Fig. 1, where the acceptable average price A_P for the user is 75 ($(80 + 70)/2 = 75$). We assume that the distance threshold $disTh$ is 100, the price threshold $priTh$ is 25, and the recommendation condition R_C is 3.

Fig. 3 shows the complete numerical examples of the location recommendation phase. It consists of three stages: (1) The user requests a query; (2) Two servers compute the recommended result; (3) The user recovers the recommended result. In this example, the recommendation condition R_C is 3. Therefore, the recommended restaurants must match all preferences of the user. Namely, $U_j + V_j + W_j \geq R_C$, hence $U_j = 1$, $V_j = 1$, and $W_j = 1$. This is also equivalent to $F_j = 1$. It means that only $j = 3$ satisfies the user's request. Thus, the cloud only recommends the restaurant D_3 to the user.

Specifically, from Step 1 to Step 4, the cloud with two servers (S_A and S_B) first determines whether the distance between one of the visited restaurants by the user and the restaurant D_j is less than or equal to a predefined distance threshold. If this condition is satisfied, the cloud sets U_j to 1; Otherwise, the cloud sets U_j to 0. Second, from Step 5 to Step 7, the cloud computes whether one of the user's favorite cuisines matches the cuisine of the restaurant D_j. If such condition is satisfied, the cloud sets V_j to 1; Otherwise, the cloud sets V_j to 0. Next, from Step 8 to Step 10, the cloud determines whether the difference between the average price of the restaurant D_j and the user is within a predefined price threshold. If this condition is satisfied, the cloud sets W_j to 1; Otherwise, the cloud sets W_j to 0. In Steps 11 and 12, the cloud calculates the recommendation flag $F_j \in \{0, 1\}$ for the restaurant

D_j. In this numerical example, the recommendation condition R_C is 3. Namely, the restaurant D_j will be recommended to the query user if and only if $U_j + V_j + W_j \geq R_C = 3$. Therefore, the recommended restaurant is D_3 because only the recommendation flag F_3 is 1 and other corresponding flags (F_1, F_2, and F_4) are 0. In Step 13, the cloud does not change the entries of the restaurant D_3 and sets the entries of other restaurants to 0.

At this moment, S_B obtains the recommended (or match) results $[\![R_1]\!]$, $[\![R_2]\!]$, $[\![R_3]\!]$, and $[\![R_4]\!]$ in the encrypted format. One way for returning the query results is that S_B sends each $[\![R_j]\!]$ to S_A for decryption and then let S_A send the decrypted results to the query user. However, in the threat model of our scheme, we assume that two servers S_A and S_B are not fully trusted. Hence, the privacy of query results in such way is not guaranteed because S_A can acquire the recommended results completely. To avoid such privacy leakage, in Step 14 S_B first adds some random values to each element of $[\![R_j]\!]$ and generates a new blinded version of $[\![R_j]\!]$ (denoted by $[\![R_j^*]\!]$). Meanwhile, in Step 14 S_B S_B sets these random values as a new vector R_j^0, where $R_j^* = R_j + R_j^0$. After that, S_B sends $[\![R_j^*]\!]$ to S_A and R_j^0 to the query user, respectively. In Step 15, S_A decrypts $[\![R_j^*]\!]$: $R_j^1 = D_{sk}([\![R_j^*]\!])$ and sends R_j^1 to the query user. Finally, the query requester computes the match results: $D_j^{\sim} = R_j^1 - R_j^0$. In this case, each element in R_j^* is random, so there is no privacy leakage against S_A.

7. Security analysis

In this section, we first analyze the security of our proposed algorithms SET and $SUEZ$. Second, we prove that our proposed privacy-preserving location recommendation scheme satisfies our design goals.

In this paper, we assume that both two servers (S_A and S_B) are semi-honest. The property of a secure computation protocol under the semi-honest model is defined as follows.

Definition 2 (*[28]*). Let $\Pi^{\mathcal{R}}(\pi)$ be the execution image of the protocol π, and $\Pi^{S}(\pi)$ be the corresponding simulated image. If $\Pi^{\mathcal{R}}(\pi)$ is computationally indistinguishable from $\Pi^{S}(\pi)$, we say that the protocol π is secure.

In the above definition, an execution image mainly includes the input, the output, and the message exchanged during an execution of a protocol.

Definition 3 (*[29]*). The SM and $SSED$ protocols are secure under the semi-honest model.

7.1. Security analysis for our building blocks

We provide the security proof for our SET protocol based on the standard simulation argument [28]. The main idea of the proof is that the input and output for this secure primitive are encrypted and only known by S_B (without knowing the private key sk), and all intermediates received by S_A are random, so no party learns inputs and outputs. Therefore, it satisfies the property of Definition 2.

Theorem 1. *The SET protocol is secure under the semi-honest model.*

Proof. Let $\Pi^{\mathcal{R}}_{S_A}(SET) = \{t, \{m_0, m_1\}\}$ be the execution image of S_A, where m_0 and m_1 are intermediates generated by the SM protocol. Likewise, let $\Pi^{S}_{S_A}(SET) = \{t_r, \{m_0', m_1'\}\}$ be the simulated image of S_A, where t_r, m_0', and m_1' are all randomly selected from $\mathbb{Z}_N$. From the process of the SET protocol, we know that $t = r_1(x-y)^2 - r_2$ (if $\alpha = 0$) or $t = r_2 - r_1(x-y)^2$ (if $\alpha = 1$), where r_0 and r_1 are two random numbers. Therefore, t and t_r are computationally indistinguishable. Likewise, based on Definition 3, we can conclude that $\{m_0, m_1\}$ is also computationally indistinguishable from $\{m_0', m_1'\}$. Therefore, $\Pi^{\mathcal{R}}_{S_A}(SET)$ and $\Pi^{S}_{S_A}(SET)$ are computationally indistinguishable.

Similarly, let $\Pi^{\mathcal{R}}_{S_B}(SET) = \{[\![x]\!], [\![y]\!], [\![t]\!], [\![\beta]\!], \{[\![z_0]\!], [\![z_1]\!], [\![z_2]\!]\}\}$ be the execution image of S_B, where $[\![z_0]\!]$, $[\![z_1]\!]$, and $[\![z_2]\!]$ are intermediates generated by the SM protocol. Let $\Pi^{S}_{S_B}(SET) = \{x', y', t', \beta', \{z_0', z_1', z_2'\}\}$ be the simulated image of S_B, where all values are chosen from $\mathbb{Z}_{N^2}$. Based on the security of the SM protocol in Definition 3, we can conclude that $\{[\![z_0]\!], [\![z_1]\!], [\![z_2]\!]\}$ is computationally indistinguishable from $\{z_0', z_1', z_2'\}$. Also, we can conclude that $\{[\![x]\!], [\![y]\!], [\![t]\!], [\![\beta]\!]\}$ and $\{x', y', t', \beta'\}$ are computationally indistinguishable because the Paillier algorithm is an indistinguishable encryption scheme [27].

According to the above analysis, $\Pi^{\mathcal{R}}_{S_B}(SET)$ and $\Pi^{S}_{S_B}(SET)$ are computationally indistinguishable. Based on Definition 2, we state that our proposed SET protocol is secure under the semi-honest model.

Remark. Compared with the SET protocol, our $SUEZ$ protocol only needs two additional homomorphic operators. Therefore, the security of our $SUEZ$ protocol can be deduced from that of the SET protocol.

7.2. Security analysis for our scheme

Theorem 2. *Our proposed scheme can protect the privacy of the dining service provider's restaurant records, users' queries (or dining records), and query results against two servers (S_A and S_B). Meanwhile, query users only acquire the recommended restaurant records matching their preferences and know nothing about other restaurant records. Also, the data access pattern is hidden from the cloud.*

Proof. Our proposed scheme consists of the system initialization phase and the location recommendation phase. In the system initialization phase, the dining service provider encrypts its restaurant records and outsources them to S_B. Since the encryption scheme is an indistinguishable encryption algorithm [27], S_B cannot derive anything from these encrypted records. Similarly, in the location recommendation phase, users encrypt their queries and upload them to S_B, so the privacy of users' queries is protected against S_B. After receiving users' queries, S_A and S_B interactively run various secure computation protocols, including SM, $SSED$, SC, SET, and $SUEZ$. When running these protocols in sequence, S_A only acquires some random values in $\mathbb{Z}_N$, and S_B always holds the inputs and outputs of them in an encrypted format. Due to the security of these secure primitives in the semi-honest model, we can conclude that no information is disclosed to S_A and S_B during their executions. At the end of this stage, query users only obtain the values of the recommended restaurant records from the returned results of S_A and S_B, while all values in other restaurant records are zero. Therefore, our scheme can protect the privacy of restaurant records, users' queries, and query results against S_A and S_B. Also, query users only acquire the recommended restaurant records and know nothing about other restaurant records. Meanwhile, in each query, two servers return query results with the same size as restaurant records to users and know nothing about them. Thus, the relationships between any two queries and their corresponding results are protected from both S_A and S_B. It implies that the data access pattern is protected, namely our scheme is fully privacy-preserving.

8. Performance evaluation

In this section, we first analyze the computation and communication costs of our proposed privacy-preserving location recommendation scheme. Second, we report the overheads of basic operators of the Paillier cryptosystem and building blocks under different parameter settings. Finally, we evaluate the performance of our proposed scheme using synthetic datasets.

Table 2
The computation cost of each entity in our proposed scheme.

	E_{pk}	D_{sk}	$SMul_{he}$	Add_{he}^{II}	SC	$SSED$	SET	SM	$SUEZ$
The DSP	$5m$	–	–	–	–	–	–	–	–
The user	$2n+t+4$	–	–	–	–	–	–	–	–
The cloud	–	$5m$	m	$m(n+t+6)$	$m(n+2)$	mn	mt	$6m$	$2m$

Table 3
The communication cost of each entity in our proposed scheme.

	$SSED$	SC	$SUEZ$	SET	SM	$\|\mathbb{Z}_{N^2}\|$	$\|\mathbb{Z}_N\|$
The DSP	–	–	–	–	–	$5m$	–
The user	–	–	–	–	–	$2n+t+4$	$10m$
The cloud	mn	$m(n+2)$	$2m$	mt	$6m$	$10m+2n+t+4$	$10m$

8.1. Performance analysis

We measure the performance of our scheme by counting the number of basic operators and secure computation primitives called by each entity. The basic operators only involve computation costs (See Table 4), whereas secure computation primitives involve computation and communication costs (See Tables 5 and 6).

Here, we assume that the dining service provider's restaurant records and the users' query are respectively denoted by $D = \{D_1, D_2, \ldots, D_m\}$ and $Q = \{C, S, A_P, disTh, priTh, R_C\}$, where $D_i = \{ID_i, CO_i, Cu_i, Av_i\}$, $C = \{c_1, c_2, \ldots, c_n\}$, and $S = \{s_1, s_2, \ldots, s_t\}$. Therefore, the computation and communication costs of each entity in our proposed scheme are given in Tables 2 and 3.

Table 2 shows that the computation costs for the dining service provider (DSP) and the user are only encryption operators for their data. The number of encryption operators depends on the size of restaurant records and the number of the user's visited restaurants and favorite cuisines. However, for a large number of queries, the dining service provider merely performs encryption once, so this one-time cost can be ignored and afforded by itself. The computation costs for the cloud ($S_A \& S_B$) consist of basic operators and secure computation primitives related to the Paillier cryptosystem. Notice that Table 2 ignores the computation cost for the user to recover the query result because this process only involves simple subtraction operators. Meanwhile, Add_{he}^{II} represents the homomorphic addition of two ciphertexts.

Table 3 demonstrates that the communication cost of the dining service provider is uploading their data to the cloud, which is linear with the size of restaurant records. However, the communication cost of the user consists of uploading query data and receiving query results, which depends on both the size of restaurant records and the number of the user's visited restaurants and favorite cuisines. The communication costs for the cloud ($S_A \& S_B$) consist of two parts. One part is receiving data from the dining service provider and the user and returning query results to the user; The other is running secure computation primitives (e.g., $SSED$, SC, and $SUEZ$) to compute the query results.

8.2. Implementation

We first implement all secure primitives built on the Paillier algorithm. Then, based on these primitives, we implement our proposed privacy-preserving location recommendation scheme.

• *Implementation Details*: We implement our proposed scheme using the C++ programming language. For quicker implementation, we directly use the NTL[2] library which includes several significant number-theoretic algorithms. To speed up our scheme, we adopt the GMP[3] library as the backend of the NTL library. Meanwhile, we also use the Boost[4] library to support network communication. Our implementation[5] is open source and available at Gitee, where it contains some secure two-party computation primitives and can be used for other applications.

• *Parameters Settings*: Our proposed scheme is built upon the Paillier cryptosystem, whose security depends on the factorization of large composite number N. Therefore, the length of N should be set to 1024, 2048, or 3072 bits in accordance with the current NIST minimum recommendation rules on cryptographic key length[6] if we choose the security parameter λ as 80, 112, or 128 bits. We assume that all values of restaurant records and users' queries are 32-bit integers.

• *Experimental Environments*: Our experiments are performed on Ubuntu 21.04 operator system with an Intel(R) Core(TM) i7-8750H 2.20 GHz CPU processor and 16 GB memory. When evaluating the communication costs, we ignore the network delay and use the console application nload[7] to count the network traffics under different parameter settings.

• *Computation Costs of Basic Operators of the Paillier Cryptosystem*: Table 4 presents the total computation costs of basic operators of the Paillier cryptosystem when running 1000 times under different parameter settings, respectively. In this table, the homomorphic addition of a plaintext and a ciphertext is represented by Add_{he}^{I}, whereas the homomorphic addition of two ciphertexts is represented by Add_{he}^{II}. Table 4 shows that, with the exception of Add_{he}^{II}, these basic operators nearly have the same overheads.

• *Computation and Communication Costs of Secure Computation Primitives*: Tables 5 and 6 show the computation and communication costs of secure computation primitives when running 1000 times under different parameter settings. According to Table 5, the SM protocol requires

Table 4
The total computation costs of basic operators of the Paillier cryptosystem when they are performed 1000 times under different parameter settings.

$\|N\|$	E_{pk}	D_{sk}	Add_h^I	Add_{he}^{II}	$SMul_{he}$
1024	2478 ms	2481 ms	2491 ms	1 ms	2471 ms
2048	16 569 ms	16 694 ms	16 605 ms	9 ms	16 336 ms
3072	49 239 ms	50 517 ms	48 814 ms	20 ms	50 208 ms

Table 5
The total communication costs of secure computation primitives when they are performed 1000 times under different parameter settings.

$\|N\|$	SM	$SSED$	SC	SET	$SUEZ$
1024	1.28 MB	2.64 MB	2.19 MB	2.19 MB	2.19 MB
2048	2.09 MB	4.49 MB	3.38 MB	3.36 MB	3.36 MB
3072	2.81 MB	5.95 MB	4.70 MB	4.69 MB	4.69 MB

Table 6
The total computation costs of secure computation primitives when they are performed 1000 times under different parameter settings.

$\|N\|$	SM	$SSED$	SC	SET	$SUEZ$
1024	22.32 s	36.01 s	38.67 s	19.19 s	24.14 s
2048	155.91 s	367.01 s	265.03 s	129.62 s	162.53 s
3072	428.59 s	905.18 s	799.05 s	386.47 s	485.92 s

2 https://libntl.org/.
3 https://gmplib.org/.
4 https://www.boost.org/.
5 https://gitee.com/locomotive_crypto/locrec.
6 https://www.keylength.com/en/4/.
7 https://github.com/rolandriegel/nload.

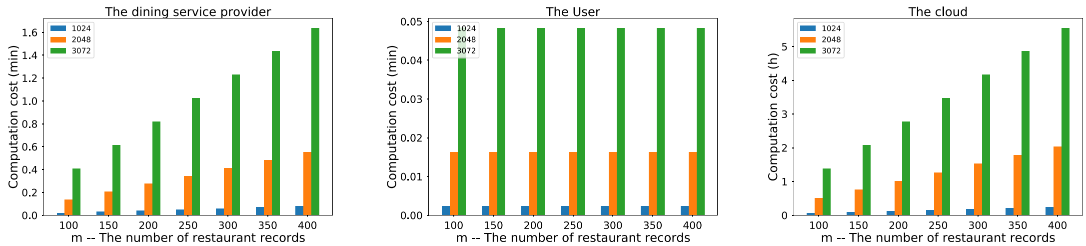


Fig. 4. The computation cost of each entity in our scheme under different security levels.

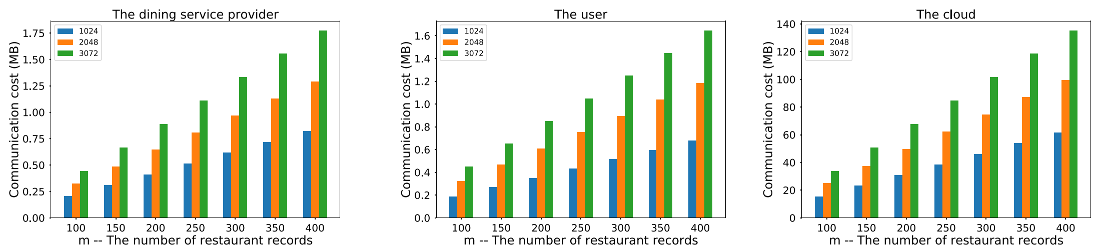


Fig. 5. The communication cost of each entity in our scheme under different security levels.

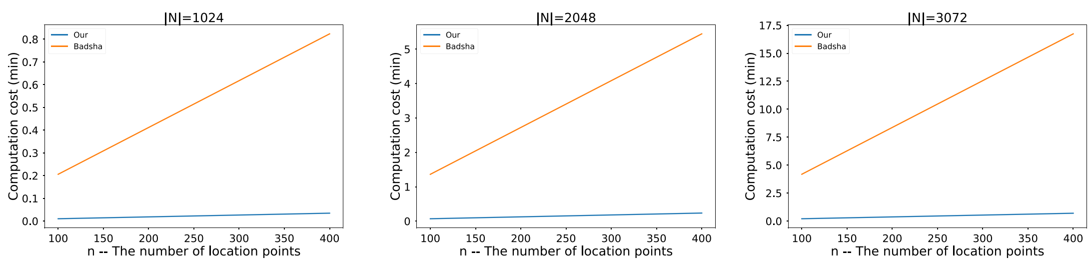


Fig. 6. The computation costs of query users between our scheme and Badsha et al.'s scheme under different security levels.

the least amount of communication, while the $SSED$ protocol requires the most. The communication costs for other protocols (i.e., SC, SET, and $SUEZ$) are essentially the same. Similarly, the SM protocol executes the fastest, while the SC protocol executes the slowest as seen in Table 6.

• *Performance of Our Scheme*: Our work is fully privacy-preserving, which means that the data access pattern is protected. Since the performances of encryption, decryption, and homomorphic operators are nearly data-independent, the performance of our scheme is independent of data distribution and depends on the data size. Therefore, we only evaluate the performance of each entity in our scheme using synthetic datasets. We assume that the number n of the user's visited restaurants is 25, and the user's favorite cuisines is 5, namely $t = 5$. For the number m of restaurant records, we respectively set it as $\{100, 150, 200, 250, 300, 350, 400\}$.

Fig. 4 demonstrates the computation cost of each entity with the increase in the number of restaurant records. Both the dining service provider and the user require very little computation costs. When $n = 25$, $t = 5$, and $m = 400$, the costs for the dining service provider are 0.082 min, 0.55 min, and 1.64 min under three different security levels, respectively. However, the computation cost for the user is not affected by the number of restaurant records. When $n = 25$ and $t = 5$, the computation costs for the user are 0.002 min, 0.016 min, and 0.048 min under three different security levels, respectively. The numbers of restaurant records, the user's visited restaurants, and favorite cuisines all affect the cloud's computation cost. Compared with other entities, the computation cost of the cloud is extremely high. When $n = 25$, $t = 5$, and $m = 400$, the computation costs for the cloud are $0.25h$, $2.04h$, and $5.56h$ under three different security levels, respectively.

Fig. 5 shows the communication cost of each entity with the increase in the number of restaurant records. Different from the computation cost, the communication cost of each entity depends on both the number of restaurant records as well as the size of the user's query. Obviously, the communication costs of the dining service provider and the user are much lower than that of the cloud. For instance, when $n = 25$, $t = 5$, $m = 400$, and $|N| = 3072$, the communication costs for them are 1.77 MB, 1.65 MB, and 135.33 MB, respectively.

• *Comparison With the Existing Schemes*: Compared with the existing works, our scheme is computationally efficient in user side and only needs one round communication for query users. To demonstrate our conclusion, we compare our scheme with the existing work [11] in computation costs of query users because they are the most similar schemes in the data dimension and use the same homomorphic encryption to protect the sensitive information. For fair comparison, we assume the visited location points in our and Badsha et al.'s works are n. The number of users held by the recommendation server in Badsha et al.'s work and the favorite cuisines in our work are fixed (both denoted by t). In our actual experiments, we set $t = 50$ and $n = 10, 15, 20, 25, 30, 35, 40$, respectively. Fig. 6 shows the computation costs for query users varying with n under different security levels. The experimental results demonstrate that our work outperforms that of Badsha et al. [11] in the computation costs of query users.

9. Conclusion

In this paper, we propose a fully privacy-preserving location recommendation scheme in outsourced environments, where the data access pattern is well protected. Our scheme simultaneously takes into account data privacy, queries with multiple attributes, query accuracy, and computation efficiency for query users. As a result, our scheme is more practical and feasible than previous related works in real-world LBS. To achieve this goal, we first propose a secure equal test protocol to check whether two encrypted values are equal to each other. Based on this protocol, we construct a secure unequal to zero test protocol to check whether an encrypted value is not equal to zero. Second, with these proposed protocols, we design our privacy-preserving location recommendation scheme. Finally, we prove the security of our scheme in the semi-honest model and show that the privacy of restaurant records, users' queries, and query results are protected against two servers. Also, query users only acquire the recommended restaurant records without learning any information about other restaurant records. Meanwhile, we conduct extensive evaluation experiments and the corresponding results confirm the efficiency of our scheme.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

Weiqi Luo was partially supported by National Key R&D Program of China (2022YFC3303604), NSFC, China (62077028, 61877029), the Science and Technology Planning Project of Guangdong, China (2021B0101420003, 2020B0909030005, 2020B1212030003, 2020ZDZX3013), the Science and Technology Planning Project of Guangzhou (202206030007), and Key Laboratory of Smart Education of Guangdong Higher Education Institutes, Jinan University, China (2022LSYS003). Anjia Yang was partially supported by National Key R&D Program of China under Grant 2021ZD0112802, Key-Area Research and Development Program of Guangdong Province under Grant 2020B0101090004 and 2020B0101360001, and National Natural Science Foundation of China under Grant 62072215. Yandong Zheng was partially supported by China Postdoctoral Science Foundation (2022M722498). Junzuo Lai was partially supported by National Natural Science Foundation of China (U2001205), Major Program of Guangdong Basic and Applied Research Project under Grant No. 2019B030302008, National Joint Engineering Research Center of Network Security Detection and Protection Technology, and Guangdong Key Laboratory of Data Security and Privacy Preserving.

References

[1] T. Peng, Q. Liu, G. Wang, Enhanced location privacy preserving scheme in location-based services, IEEE Syst. J. 11 (1) (2017) 219–230.

[2] Y. Zheng, X. Xie, Learning travel recommendations from user-generated GPS traces, ACM Trans. Intell. Syst. Technol. 2 (1) (2011) 2:1–2:29.

[3] F. Yu, N. Che, Z. Li, K. Li, S. Jiang, Friend recommendation considering preference coverage in location-based social networks, in: J. Kim, K. Shim, L. Cao, J. Lee, X. Lin, Y. Moon (Eds.), Advances in Knowledge Discovery and Data Mining - 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part II, in: Lecture Notes in Computer Science, vol. 10235, 2017, pp. 91–105.

[4] Y. Huo, B. Chen, J. Tang, Y. Zeng, Privacy-preserving point-of-interest recommendation based on geographical and social influence, Inform. Sci. 543 (2021) 202–218.

[5] A. Yang, J. Xu, J. Weng, J. Zhou, D.S. Wong, Lightweight and privacy-preserving delegatable proofs of storage with data dynamics in cloud storage, IEEE Trans. Cloud Comput. 9 (1) (2021) 212–225.

[6] A. Yang, J. Weng, K. Yang, C. Huang, X. Shen, Delegating authentication to edge: A decentralized authentication architecture for vehicular networks, IEEE Trans. Intell. Transp. Syst. 23 (2) (2022) 1284–1298.

[7] Y. Zheng, R. Lu, Y. Guan, J. Shao, H. Zhu, Efficient privacy-preserving similarity range query with quadsector tree in ehealthcare, IEEE Trans. Serv. Comput. (2021).

[8] Y. Zheng, R. Lu, Y. Guan, J. Shao, H. Zhu, Towards practical and privacy-preserving multi-dimensional range query over cloud, IEEE Trans. Dependable Secure Comput. (2021).

[9] I. Lien, Y. Lin, J. Shieh, J. Wu, A novel privacy preserving location-based service protocol with secret circular shift for k-NN search, IEEE Trans. Inf. Forensics Secur. 8 (6) (2013) 863–873.

[10] Y. Guan, R. Lu, Y. Zheng, J. Shao, G. Wei, Toward oblivious location-based k-Nearest neighbor query in smart cities, IEEE Internet Things J. 8 (18) (2021) 14219–14231.

[11] S. Badsha, X. Yi, I. Khalil, D. Liu, S. Nepal, E. Bertino, Privacy preserving location recommendations, in: Web Information Systems Engineering - WISE 2017 - 18th International Conference, Puschino, Russia, October 7-11, 2017, Proceedings, Part II, in: Lecture Notes in Computer Science, vol. 10570, 2017, pp. 502–516.

[12] Q. Xu, H. Zhu, Y. Zheng, J. Zhao, R. Lu, H. Li, An efficient and privacy-preserving route matching scheme for carpooling services, IEEE Internet Things J. (2022).

[13] X. Ma, H. Li, J. Ma, Q. Jiang, S. Gao, N. Xi, D. Lu, APPLET: a privacy-preserving framework for location-aware recommender system, Sci. China Inf. Sci. 60 (9) (2017) 92101.

[14] S. Badsha, X. Yi, I. Khalil, D. Liu, S. Nepal, E. Bertino, K. Lam, Privacy preserving location-aware personalized web service recommendations, IEEE Trans. Serv. Comput. 14 (3) (2021) 791–804.

[15] H. Zhu, F. Liu, H. Li, Efficient and privacy-preserving polygons spatial query framework for location-based services, IEEE Internet Things J. 4 (2) (2017) 536–545.

[16] D. Li, J. Wu, J. Le, X. Liao, T. Xiang, A novel privacy-preserving location-based services search scheme in outsourced cloud, IEEE Trans. Cloud Comput. (2021).

[17] Q. Huang, J. Du, G. Yan, Y. Yang, Q. Wei, Privacy-preserving spatio-temporal keyword search for outsourced location-based services, IEEE Trans. Serv. Comput. (2021).

[18] L. Li, R. Lu, C. Huang, EPLQ: efficient privacy-preserving location-based query over outsourced encrypted data, IEEE Internet Things J. 3 (2) (2016) 206–218.

[19] K. Järvinen, Á. Kiss, T. Schneider, O. Tkachenko, Z. Yang, Faster privacy-preserving location proximity schemes, in: J. Camenisch, P. Papadimitratos (Eds.), Cryptology and Network Security - 17th International Conference, CANS 2018, Naples, Italy, September 30 - October 3, 2018, Proceedings, in: Lecture Notes in Computer Science, vol. 11124, 2018, pp. 3–22.

[20] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, D. Boneh, Location privacy via private proximity testing, in: Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011, 2011.

[21] C. Chen, J. Zhou, B. Wu, W. Fang, L. Wang, Y. Qi, X. Zheng, Practical privacy preserving POI recommendation, ACM Trans. Intell. Syst. Technol. 11 (5) (2020) 52:1–52:20.

[22] S. Gao, X. Ma, J. Zhu, J. Ma, APRS: a privacy-preserving location-aware recommender system based on differentially private histogram, Sci. China Inf. Sci. 60 (11) (2017) 119103:1–119103:3.

[23] Y. Elmehdwi, B.K. Samanthula, W. Jiang, Secure k-nearest neighbor query over encrypted data in outsourced environments, in: IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014, 2014, pp. 664–675.

[24] P. Mohassel, Y. Zhang, SecureML: A system for scalable privacy-preserving machine learning, in: 2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017, 2017, pp. 19–38.

[25] Y. Zheng, R. Lu, Y. Guan, S. Zhang, J. Shao, H. Zhu, Efficient and privacy-preserving similarity query with access control in eHealthcare, IEEE Trans. Inf. Forensics Secur. 17 (2022) 880–893.

[26] S. Zhang, S. Ray, R. Lu, Y. Zheng, Y. Guan, J. Shao, Towards efficient and privacy-preserving interval skyline queries over time series data, IEEE Trans. Dependable Secure Comput. (2022).

[27] P. Paillier, Public-key cryptosystems based on composite degree residuosity classes, in: International Conference on the Theory and Applications of Cryptographic Techniques, 1999, pp. 223–238.

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

[29] B.K. Samanthula, Y. Elmehdwi, W. Jiang, K-Nearest neighbor classification over semantically secure encrypted relational data, IEEE Trans. Knowl. Data Eng. 27 (5) (2015) 1261–1273.

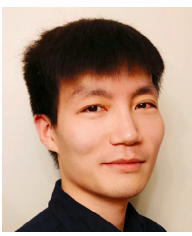
[30] Y. Zheng, H. Duan, X. Yuan, C. Wang, Privacy-aware and efficient mobile crowdsensing with truth discovery, IEEE Trans. Dependable Secur. Comput. 17 (1) (2020) 121–133.

[31] X. Liu, R.H. Deng, W. Ding, R. Lu, B. Qin, Privacy-preserving outsourced calculation on floating point numbers, IEEE Trans. Inf. Forensics Secur. 11 (11) (2016) 2513–2527.

[32] Privacy-preserving travel recommendation, https://gitee.com/locomotive_crypto/stay_points.

Lulu Han received the B.S. degree from the School of Computer Science, Shanxi Normal University, Xian, China, in 2019. He is currently working toward the Ph.D. degree with the College of Cyber Security, Jinan University, Guangzhou, and his research interests include secure multiparty computation and data security and privacy.

Weiqi Luo received his B.S. degree from Jinan University in 1982 and Ph.D. degree from South China University of Technology in 2000. Currently, he is a professor with School of Information Science and Technology in Jinan University, Guangzhou. His research interests include network security, big data, artificial intelligence, etc. He has published more than 100 high-quality papers in international journals and conferences.



Anjia Yang (M'17) received the Ph.D. degree in department of Computer Science from the City University of Hong Kong in 2015. He held a post-doctoral position in the City University of Hong Kong from 2015 to 2016, and in Jinan University from 2016 to 2019, respectively. From 2018 to 2019, he was a visiting scholar in BBCR group in University of Waterloo. He is currently an associate professor in Jinan University, Guangzhou. His research interests include security and privacy in internet of things, vehicular networks, blockchain and cloud computing, etc. He has published over 30 international papers including journals and conferences, such as IEEE TDSC, IEEE TMC, IEEE TPDS, IEEE TSC, IEEE TCC, IEEE TITS, IEEE TVT, ESORICS, WiSec et al. He served as PC members or organizers for more than 20 international conferences. He also serves as an academic editor for Security and Communication Networks.

Yandong Zheng received the M.S. degree from the Department of Computer Science, Beihang University, China, in 2017, and received the Ph.D. degree from the Department of Computer Science, University of New Brunswick, Canada, in 2022. Since 2022, she has been a lecturer with the School of Cyber Engineering, Xidian University. Her research interest includes cloud computing security, big data privacy, and applied privacy.

Rongxing Lu (S'09-M'11-SM'15-F'21) is a University Research Scholar, an associate professor at the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Canada. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from April 2013 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious Governor Generals Gold Medal when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. Dr. Lu is an IEEE Fellow. His research interests include applied cryptography, privacy enhancing technologies, and IoTBig Data security and privacy. He has published extensively in his areas of expertise (with H-index 78 from Google Scholar as of Jan 2022), and was the recipient of 9 best (student) paper awards from some reputable journals and conferences. Currently, Dr. Lu serves as the Chair of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee), and the founding Co-chair of IEEE TEMS Blockchain and Distributed Ledgers Technologies Technical Committee (BDLT-TC). Dr. Lu is the Winner of 2016-17 Excellence in Teaching Award, FCS, UNB.

Junzuo Lai is a professor in the College of Information Science Technology, Jinan University, China. He received the Ph.D. degree in computer science and technology from Shanghai Jiao Tong University, China, in 2010. From August 2008 to April 2013, he was a research fellow in Singapore Management University. His research interests include cryptography and information security. He has published over 40 papers in international conferences and journals such as EUROCRYPT, ASIACRYPT, PKC, IEEE TDSC and IEEE TIFS.

Yudan Cheng received the B.S. and M.S. degrees from the College of Computer Science and Engineering, Northwest Normal University, Lanzhou, China, in 2016 and 2019, respectively. She is currently working toward the Ph.D. degree with the College of Cyber Security, Jinan University, Guangzhou, China. Her current research interests include trust management and privacy preservation in vehicular networks.