2018 Master Thesis
Privacy-Preserving Recommendation on Location Based Service

Submission Date： 24th July, 2018
Supervisor： Hayato Yamana

Department of Computer Science and Engineering,
School of Fundamental Science and Engineering,
Student ID：5116FG28-6

LYU Qiuyi

Abstract

With the development of big data mining and location-based services (LBS), the convenience in the life of consumers has been enhanced owing to intelligent navigation and recommendation services such as Amazon, Netflix, and Google Maps. Not only do these servers know the locations and needs of users, but they also provide various services corresponding to these locations and requirements. Usually, the users are sensitive to disclose their personal information; there are unavoidable security concerns because confidential information such as the users' financial or health status, sexual orientation, and religious preferences can be easily misused by malicious third parties. The mainstream solution to this problem is employing the privacy k-NN search techniques such as k-anonymous and privacy information retrieval (PIR). However, there are two major bottlenecks corresponding to the privacy k-NN search technique. One is that it only provides the nearest points of interest (POI) to the users without any recommendations based on the users' history behavior. This limited service eventually results in a situation in which no user would prefer to continue using it. The other is that only the users hold the privacy key in privacy k-NN search's scenario; thus, the service providers cannot obtain any user information to analyze which is the service providers' major source of profit. To this end, we propose two solutions that could benefit both consumers and service providers. To solve the first problem, the proposed protocol provides recommendation services without revealing the users' information such as location and POI requirement to the service provider, based on fully homomorphic encryption (FHE). Specifically, the proposed protocol can enable service providers to generate location-based recommendations by combining the encrypted user's information from the user side and other users' preferences stored in an encrypted database, which can be updated. For the second problem, a privacy service provider (PSP) was designed to generate and hold the privacy key. Thus, service providers can homomorphically compute aggregate information concerning user behavior patterns, and send the encrypted results to PSP to ensure decryption, while maintaining the privacy of individual users. Compared with the previous studies that only concentrated on the private k-NN search, the novelty of the proposed protocol is that the design of a commercially valuable privacy recommendation system mechanism on LBS, which can be applied practically.

Keywords --- PPDM, private preserving data mining, LBS, location based service

Contents

1. Introduction

Location-based services (LBS) and their recommendation systems based on big data mining technology provide various intelligent services to the consumers. The services recommend products, entertainments, news, or POIs based on the consumers' purchasing and search history habits. A user can receive a more accurate recommendation if a larger amount of user information is accessible to the service providers. However, the consumers' privacy can be threatened if the service providers integrate and analyze the enormous data that they hold. There is a possibility that the service providers could control and exploit highly sensitive personal information pertaining to users' sexual orientation, religious views, ethnicity, political views, and personal traits, which could lead to critical social problems in the near future. Even though governments have introduced the data privacy policy to regulate the way of collecting and using data, and the service providers provide assurance to protect users' personal information from being revealed, there is still an inherent risk for deliberate or inadvertent misuse; a case in point is the Facebook data privacy scandal that happened in April 2018 and led to the shutdown of Cambridge Analytica. In addition, if the servers are attacked by malicious third parties, there is a high risk for the consumers' personal information to be disclosed, similar to in the worldwide ransomware attack named WannaCry, which happened in May 2017 and targeted computers running the Microsoft Windows operating system. The ransomware encrypted the important files of the user and demanded payments to recover these files.

The conventional encryption algorithms such as AES and DES can protect data effectively, although their limitation is that the ciphertexts must be decrypted before they can be operated upon.

The lattice-based fully homomorphic encryption (FHE) scheme [1], which was introduced by Craig Gentry in 2009 appears to be a vital key to solving the security problem in the cloud field, because it supports arbitrary functions, and a number of operations can be performed on the ciphertexts stored in the cloud without decryption.

Before the homomorphic encryption was introduced, location privacy and privacy-preserving recommendations were two isolated fields because of the different security techniques involved. However, the lattice-based FHE can be applied in both fields. Therefore, this study attempts to design a mechanism that can combine the location privacy and privacy-preserving recommendation to ensure higher practical and commercial value for the LBS privacy.

1.1 Organization

The remaining paper is divided into six sections. In Section 2, the concepts of fully homomorphic encryption, the Hilbert curve, and the collaborative filtering recommender based on the co-occurrence matrix are explained. In Section 3, the related works based on both spatial transformations and cryptographic methods are discussed. In Section 4, the proposed model, and the phases from initialization to recommendation are described. In Section 5, the performance and evaluation of the proposed model are described. The conclusions are presented in Section 6.

2. Background

2.1 Notations

Table 1 describes notations referred to our protocols

TABLE 1
NOTATIONS

Notation	Description
Z	Integer ring
$\Phi_m(x)$	m-th cyclotomic polynomial
R	Polynomial quotient ring R: = Z[x]/ $\Phi_m(x)$
C	Arithmetic circuit
$S_{u,j}$	Preference of user u for item j
$Sim(i,j)$	Similarity between items i and j
$N_{(i)}$	Number of users who like item i
N	Size of the co-occurrence matrix
m	Ring modulus
l	Number of slots in a single cipher text
L	Number of ciphertext modules
λ	Security Level
$\oplus$	Addition over ciphertexts
$\otimes$	Multiplication over ciphertexts

2.2 Fully Homomorphic Encryption (FHE)

FHE, which can perform arbitrary functions and a number of operations over the encrypted data, was first highlighted by Rivest et al. [2], and has been known to be applied in a considerable number of applications such as E-health, cloud computing and LBS.

In 2009, Craig Gentry theoretically demonstrated the possibility of FHE construction based on ideal lattices [3]. To ensure security, noise is added to the ciphertexts. The noise keeps increasing after the homomorphic operations are applied. If the noise size exceeds the limit value, decryption errors could occur. Therefore, Gentry used the bootstrapping technique, which can refresh the ciphertexts by running the decryption function homomorphically to overcome the growth of noise.

In 2012, Brakerski, Gentry, and Vaikuntanathan introduced the BGV scheme [4], which overcomes the growth of noise by using key switching and the modulus switching technique that uses bootstrapping as an optimization. The key switching technology is used to convert

the expanded ciphertexts to fresh ciphertexts, and the modulus switching technology is used to reduce the noise. In addition, among quite a few works [5–9], the BGV scheme is one of the most efficient FHE schemes involving the modulus and key switching techniques. In [5], a small overhead characteristic of the BGV scheme is reported. Brakerski [6] also suggested that the BGV scheme improved the defect of the Brakerski and Vaikuntanathan scheme [10], wherein the scheme requires a large modulus to perform decryption owing to the large noise growth because the modulus switching of BGV scales down the ciphertexts vector after every multiplication. In [5][7], the authors mentioned that more efficient FHE schemes can be achieved by combining SIMD operations and the BGV scheme.

In [10], Smart and Vercauteren introduced a method to decompose the size of ciphertext by packing based on the Chinese remainder theorem (CRT), which enables single-instruction-multiple-data (SIMD) operations to be applied on homomorphic encryption.

IBM Research released the open source of the homomorphic encryption library HElib [11], which implements ring-learning with errors (R-LWE) based on the BGV scheme. It also supports Smart–Vercauteren ciphertexts packing, and contains four key algorithms: key generation (keyGen), encryption (Enc), decryption (Dec), and ciphertext calculation evaluation (evk,C,c1,...,ct). The specific algorithms are as follows: Set $R_q=Z_q[x]/(\Phi_m(x))$, where q is a prime and $\Phi_m(x)$ is the m-th cyclotomic polynomial. When m is the prime, $\Phi_m(x) = \sum_{j=0}^{m-1} x^j$. When m is the multiple of two, $\Phi_m(x) = x^{\frac{m}{2}} + 1$

1. **Setup:** Setup(1^λ,1^L)
   Given the security parameter λ and the circuit depth L as input.
2. **Key generation:** $sk = (1,s) = (1,s_0 + s_1X + s_2X^2 + \dots s_{N-1}X^{N-1}) \in \mathbb{R}_q^2$, $s \in \{0,\pm 1\}$
   $pk = (a_0 , a_1) = ([-(as + te)]_q , a) \in \mathbb{R}_q^2$, e is the small error and a is uniformly sampled from R_q
3. **Encryption:** c ← **Enc**(m , pk)
   $c = (c_0 , c_1) = (a_0v + te_0 + m, a_1v + te_1) \in \mathbb{R}_q^2$ where $v \in R$ is a randomly chosen polynomial with coefficients in $\{0, \pm 1\}$ and e_0, e_1 are small errors
4. **Decryption:** m ← **Dec**(c , sk)
   $c \cdot sk = c_0 + c_1s$
   $= (a_0 + a_1s)\, v + te_0 + te_1s + m$
   $= t\,(-ev + e_0 + e_1s) + m \pmod q$
   After performing modulo t, the correct plaintext can be recovered.

5. **Addition:**
   $m_1+m_2 \leftarrow Dec(Enc(m_1) \oplus Enc(m_2))$

6. **Multiplication:**
   $m_1 m_2 \leftarrow Dec(Enc(m_1) \otimes Enc(m_2))$

FHE scheme is known to be applied in a considerable number of applications such as E-health, Cloud Computing and Location-Based Service.

2.3 Hilbert Curve

The Hilbert Space-Filling curve [12] which is a continuous fractal space-filling curve first described by the German mathematician David Hilbert in 1891. There are two reasons why we adopted the Hilbert Curve among so many curves. One is the capability of partially retaining the neighboring adjacency of the original data. [13] [14] shows the Hilbert curves can achieve the best clustering properties. The other is the it's dimensionality reduction characteristic that it can traverse through all cells in the 2-D space and transform them to 1-D data structure which enables database to store and retrieve the POI's location information in a more effective way.

The basic idea of applying Hilbert curve for POI search in the proposed method is based on the works of [15–17]. Ali et al. [15] proposed a fundamental approach that applies Hilbert curves as efficient one-way transformations to preserve the users' location privacy by encoding the space of all objects and answering the query blindly in the encoding space. In [16], Lien at al. introduced the secret circular shift protocol (SCSP) for k-NN search based on the Hilbert curve and a public key homomorphic cryptosystem. Utsunomiya at el. [17] proposed a lightweight private circular query protocol on the basis of SCSP.

Figure 1 shows an example of the structure of the combination of Hilbert curve and LBS. Figure 1(a) shows a Hilbert curve over the map, Fig. 1(b) shows the POI look-up table stored on the user side, and Fig. 1(c) shows the POI Info table stored on the server side. The look-up table saves the POI's cell number and the relationship between the POI's cell number and block number, whereas the POI Info table saves the information of the POI and the relationship between the POI and block number. In addition, owing to the proposed encrypted database structure, which is introduced in Section 4, we divide the table into a set of blocks.

A user is located in cell number 8 which belongs to Block 1 according to the look-up table in Fig. 1(a), and wishes to request nearby POIs from the server. Thus, the endpoint of the user informs the user's current block number 1 to the LBS server. After the server receives the block number, it can return POIs corresponding to block 1, i.e., p1, p2, p3, to the user's endpoint.

However, the user's location and requests are sensitive, thus, sending such information in plaintext might threaten users' privacy.

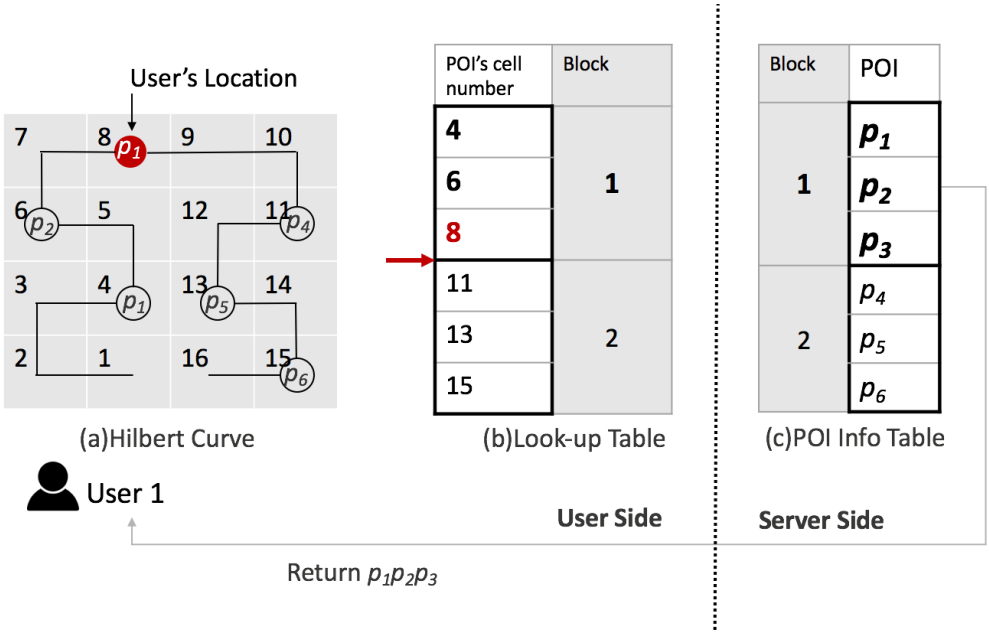


Fig.1. Example of the structure of combination between Hilbert Curve and Location-Based Service

2.4 Collaborative Filtering Recommender Based on Co-occurrence Matrix [18][19]

There are two well-known algorithms in the field of collaborative filtering filed: user-based and item-based. In this study, item-based collaborative filtering (CF) recommender is applied in our proposed protocol for two reasons:

1. Similarity between items is more stable than that between users.
2. Our task is to calculate the similarity between items and provide recommendations.

E-commerce websites such as Amazon or subscription services such as Netflix usually provide users recommendations such as "The customer who bought A also bought B" or "The customer who watched A also watched B." We can summarize the metric from these sentences as follow:

$$sim(i,j) = \frac{|N_{(i)} \cap N_{(j)}|}{N_{(i)}} \qquad (1)$$

$|N_{(i)} \cap N_{(j)}|$ denotes the number of users who like both item i and item j. $N_{(i)}$ denotes the number of users who like item i. The software library HElib that implements homomorphic encryption cannot directly support division or floats at the present time because of technical difficulty; thus, we cannot apply $sim(i,j) = \frac{|N_{(i)} \cap N_{(j)}|}{N_{(i)}}$ when both $|N_{(i)} \cap N_{(j)}|$ and $N_{(i)}$ are encrypted. Instead, we adopt $sim(i,j) = C_{[i][j]} = |N_{(i)} \cap N_{(j)}|$ in the recommendation phrase.

The calculation of $\frac{|N_{(i)} \cap N_{(j)}|}{N_{(i)}}$ is implemented after the recommended result is decrypted by the decryption server. We will provide a detailed explanation of this process in Section 3.

There are two steps to generate a co-occurrence matrix in the ItemCF algorithm.

1. Create a user-item inversion list
2. Traverse the inversion list. If item i and item j are in the inversion list of the same user, then $C_{[i][j]}(i \neq j)$ in the co-occurrence matrix C will be incremented by 1. Furthermore, every time item i appears, $C_{[i][j]}(i = j)$ is incremented by 1 simultaneously.

An example is shown in Fig.2:

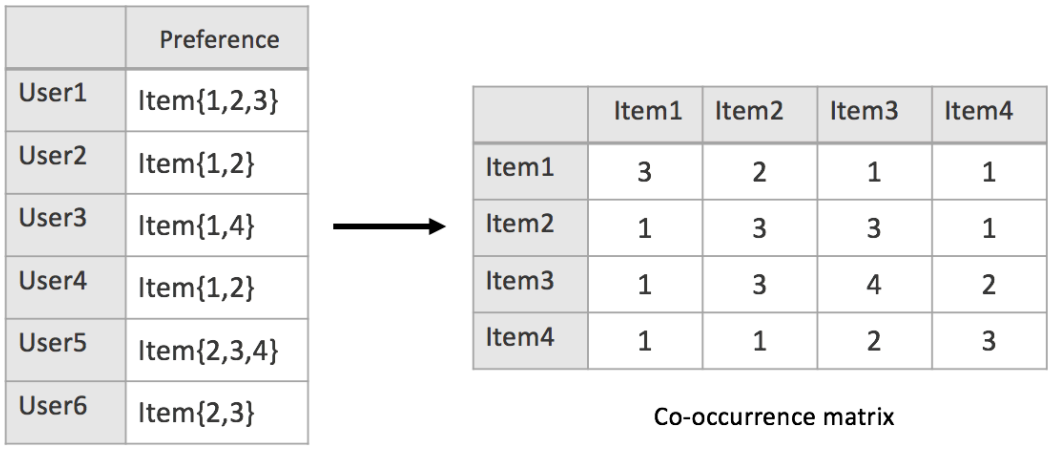


Fig.2 The deform of co-occurrence matrix

As for the generation of recommended results, the co-occurrence matrix will be multiplied by the user's preference array.

The example is shown below (Fig.3):

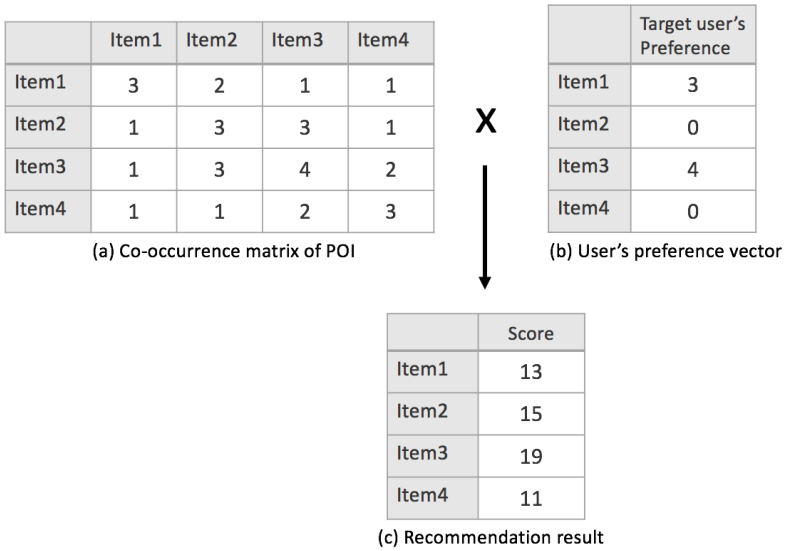


Fig.3 Generation of recommendation result

3. Related Work

This section discusses the related works on privacy-preserving on LBS; such works are divided into two categories. One category is based on spatial transformation that solves the private k-NN search on LBS by using cloaking area and k-anonymous techniques. The other category is based on cryptography, which not only provides a higher security level on the private k-NN search but also privacy-preserving recommendation by FHE.

3.1 LBS based on Spatial Transformations

To protect the user's location data from being disclosed to the server, some existing methods perturb the real data by adding redundant data. For instance, the methods generate a few random fake locations and send redundant queries to the LBS server to hide the real location.

Gruteser et al. introduced k-anonymous ideas into the field of LBS privacy protection in 2003 [20], which adopts spatial and temporal cloaking. The definition of k-anonymous is the a querying user cannot be distinguished from at least k-1 users who also send queries to the server, where k is customized by the user. To achieve this principle, a trusted third party (TTP) is employed. Fig.4, based on [20], shows the structure that includes the anonymizer-trusted third party:

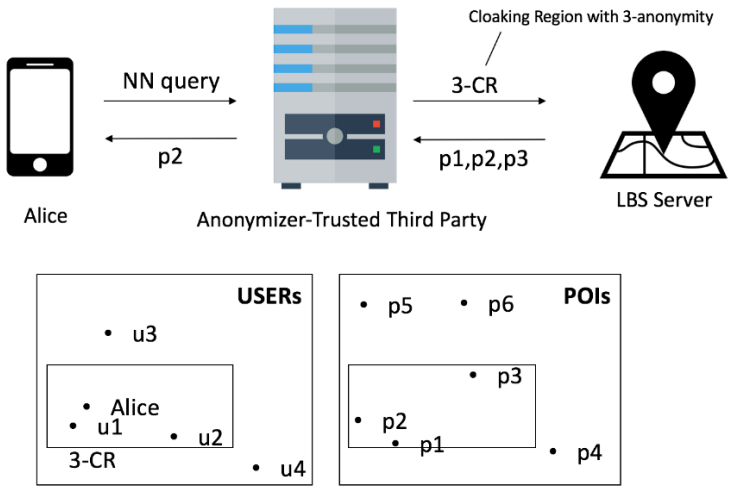


Fig.4 The model of Private-Preserving Recommendation on LBS (based on [20])

The existing methods have several drawbacks:

1. **Weak in correlation attacks:**

Assume a target user continuously sending requirements while he/she is moving. According to the existing methods, the TTP will generate a set of cloaking regions CRi that all contain user u. Malicious third parties can narrow down the size of the cloak region by intersecting all

the continuous CRi generated by the user. Therefore, the probability of identifying the querying user becomes extremely high.

2. **Single point of attack:**

Another drawback of the existing method is that TTP will obtain the single point of attack. This means if TTP is invaded by a malicious third party, then the entire security system will disintegrate.

3. **Hard to construct:**

In many methods that contains TTP, a considerable number of users need to subscribe to the service and send their locations to the TTP continuously; otherwise, cloaking areas with k-anonymous cannot be constructed.

3.2 LBS based on Cryptographic methods

a. Private Information Retrieval (PIR)

In order to improve security and protect against correlation attacks, Ghinita et al. proposed a higher security method called private information retrieval (PIR) [19] to address the privacy protection problem on LBS in 2008. It allows users to retrieve the POI privately from the database without informing the server what POI the user has requested based on quadratic residuosity assumption (QRA). It also abolishes the TTP structure to protect against single point attack. Another cryptographic method to solve the private nearest-neighbor search is homomorphic encryption. A private circular query protocol (PCQP) proposed by Lien et al. [15] and a lightweight protocol proposed by Utsunomiya [16] are state-of-the-art protocols for privacy-preserving k-NN search based on HE and Hilbert curve.

All these existing works described above have a major disadvantage: the service is limited because they only provide privacy k-NN search without privacy-preserving recommendation. The consumers not only want to know which POI is the nearest to them but also which POI is the best among the POIs available near them.

b. Privacy-preserving recommendation

Homomorphic encryption provides an easier and more flexible way to solve the privacy-preserving recommendation because it can process calculations and analyses over the ciphertexts. Badsha et al. [22] introduced privacy-preserving item-based CF based on the ElGamal cryptosystem in 2016. One year later, Badsha et al. [23] introduced a protocol that protects the privacy of user's online services by performing the BGN scheme.

However, there is no specialized privacy-preserving mechanism for a location-aware recommender system. The technical difficulty faced by the system is that it not only needs to provide recommendations over encrypted history records but also needs to deal with the encrypted real-time location data created by users

4. Proposed Model

There exist two problems in previous privacy k-NN search techniques used for POI searching. One is that it only returns the nearest POIs to the user without computing the similarity between the POIs near the user to generate the recommendations. The other is that service providers cannot obtain any user behavior statistics to make a profit because only users hold the privacy key in privacy k-NN search.

To solve the first problem, we designed privacy-preserving recommendation on LBS Server (PPRS) in the proposed model to generate recommendation services over an encrypted database (ED) that can be updated.

To solve the second problem, we designed a privacy service provider (PSP) to generate and hold the privacy key. Service providers can perform computations of user behavior patterns over ED homomorphically and send the encrypted results to the PSP for decryption. This design makes it possible to analyze aggregate information about user behavior patterns while maintaining the privacy of individual users.

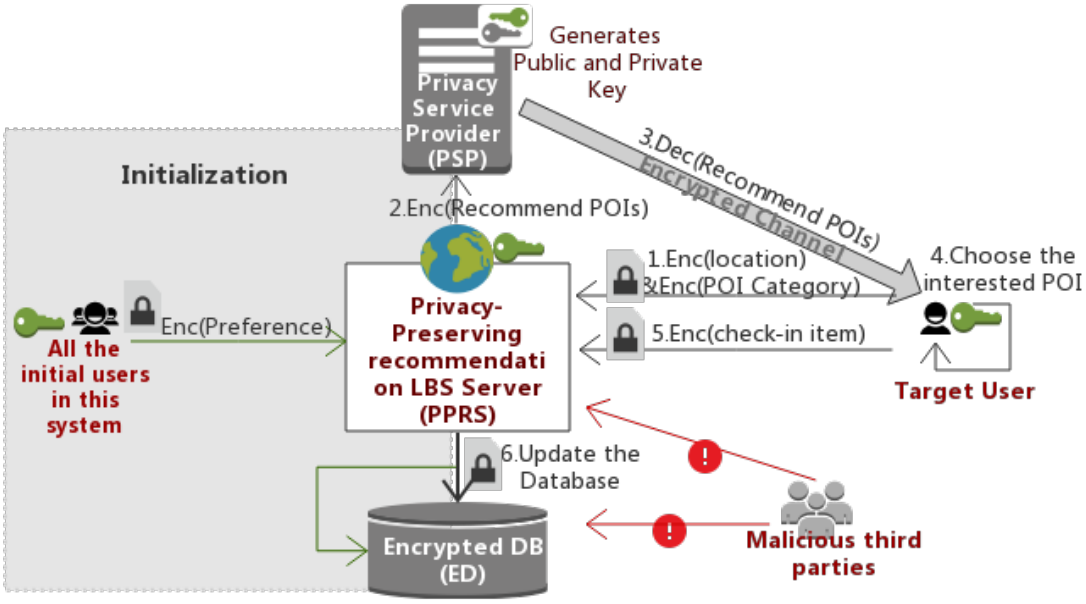


Fig.5 The model of Privacy-preserving recommendation on LBS

4.1 Structure

The structure of the proposed model is shown in Fig.5; it includes three main parts: PSP, PPRS, and ED. The overview of the process is described below.

Target users send their encrypted location and POI requests to the PPRS (Step 1). The PPRS computes the encrypted recommendation list and sends it to the PSP to decrypt (Step 2).

After decryption, the PSP transforms the recommendation lists into plaintext and sends them to the users through an encrypted channel and sends them to the users (Step 3). The users choose the POI in which they are interested from the recommendation list (Step 4). Then, they encrypt the result and send them to the ED through the PPRS (Step 5 and 6) for further recommendation on the PPRS.

4.1.1 Privacy Service Provider(PSP):

The PSP has two main tasks: generate a set of public and private keys and decryption. It needs to decrypt two types of data from the PPRS: recommendation result and aggregate information about user behavior patterns. In addition, the recommendation result in plaintext is sent to the target user through an encrypted channel.

4.1.2 Privacy-Preserving Recommendation on LBS Server (PPRS):

PPRS is the main part in the proposed model and performs two tasks. The first task involves receiving both the encrypted POI requests and the selection matrix corresponding to the user's location from the user. The second task involves generating the recommendation list based on four factors: POI requests, selection matrix, Co-occurrence Matrix, and User History Behavior Matrix. (Fig.6)

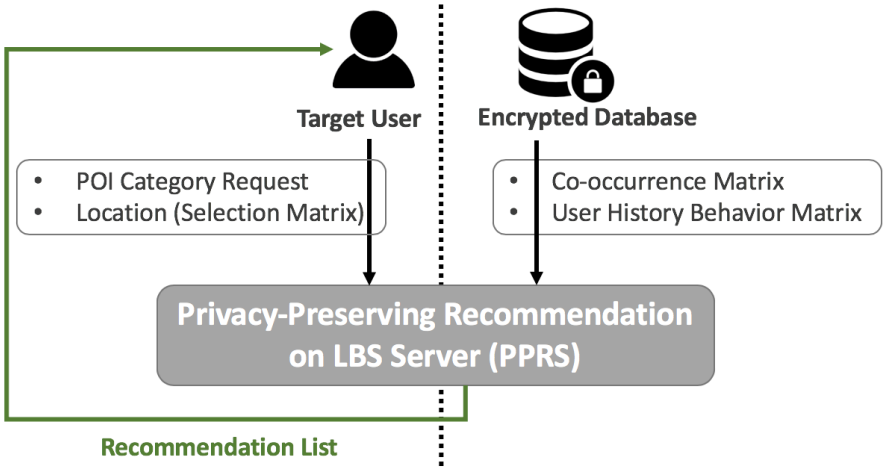


Fig.6 Generation of the recommendation list

It also takes responsibility for calculating the aggregate information about user behavior patterns over encrypted database.

4.1.3 Encrypted Database (ED):

ED stores two kinds of matrixes encrypted by FHE, one is the co-occurrence matrix of POI (Fig.7) and the other is users' history behavior matrix (Fig.8), where $\varepsilon(\bullet)$ represents the encrypted data. The novelty of this encrypted database is that it can be updated so that it can provide the recommendation by lasted data.

4.1.3.1 Co-occurrence Matrix of POI:

As shown in Fig.7, the co-occurrence matrix records the number of times each pair of items appears together in the user-item inversion list. The sequence of items is the same as that in the POI-table generated by the Hilbert Curve according to the map so that users can locate the exact small square corresponding to their location secretly. The co-occurrence matrix of POI is divided into fixed-length squares. The length can be customized by the users; the value depends on the security level or the desired recommendation range of the users.

	I1	I2	I3	I4
I1	ε(3)	ε(2)	ε(1)	ε(1)
I2	ε(2)	ε(3)	ε(3)	ε(1)
I3	ε(1)	ε(3)	ε(1)	ε(2)
I4	ε(1)	ε(1)	ε(2)	ε(3)

Fig.7 Co-occurrence matrix of POI

4.1.3.2 User Behavior History Matrix:

As shown in Fig.8, the user behavior history matrix is the user-item inversion list where (1) denotes that users like the item (POI) while (0) denotes that users do not like.

	U1	U2	U3	U4
I1	ε(1)	ε(0)	ε(0)	ε(0)
I2	ε(1)	ε(1)	ε(1)	ε(0)
I3	ε(0)	ε(1)	ε(0)	ε(1)
I4	ε(0)	ε(0)	ε(0)	ε(0)

Fig.8 User Behavior History Matrix

4.2 Process

The processes of the proposed model are described in this subsection.

4.2.1 Initialization Phase

The main task during the initialization phase is to create an initial item-item co-occurrence matrix to provide recommendations. All the initial users generate their personal co-occurrence matrices that contain information of POIs in which the users are interested. Then, the users encrypt them with the public key that is sent from the PSP and send the encrypted co-occurrence matrices to PPRS. After that, the PPRS generates the initial item-item co-occurrence matrix by adding all the encrypted co-occurrence matrices from the initial users.

Algorithm : Initialization

Input: User-Item tuple set UI $= \{(u, i) | u \in \mathrm{U}, i \in \mathrm{I}\}$, User set U, Item INDEX set I

// User-Item tuple (u,i) represents user u has visited POI i.

Output: Co-occurrence Matrix CM

// CM represents how many users visited the same pair of POSs.

function AggregateCoMatrix (a set of CM_u)

// Executed at PPRS with FHE

//calculate # of co-rated(overlapping) users between items

1. **Assign** CM[1..|I|][1..|I|] = 0 //co-occurrence matrix
2. **foreach** user u in U
3. **for** i=1 to |I|
4. **for** j=1 to |I|
5. CM[i][j]+= CM_u [i][j]
6. **endfor**
7. endfor
8. endforeach
9. return CM

function CoMatrix (UI,u,I)

// Executed at each user u without FHE

//calculate # of co-rated(overlapping) users between items

1. **Assign** CM_u[1..|I|][1..|I|] = 0 // co-occurrence matrix for user u
2. **for** i=1 to |I|-1
3. **if** ($(u, i) \in UI$) CM_u [i][i] +=1 **endif**
4. **for** j=i+1 to |I|
5. **if** ($(u, i) \in UI$ **and** $(u, j) \in UI$ **and** $i \neq j$)
6. CM_u [i][j]+=1, CM_u [j][i]+=1
7. **endif**
8. **endfor**
9. endfor
10. **encrypt** CM_u with pk
1. **return** CM_u

4.2.2 Selection Phrase

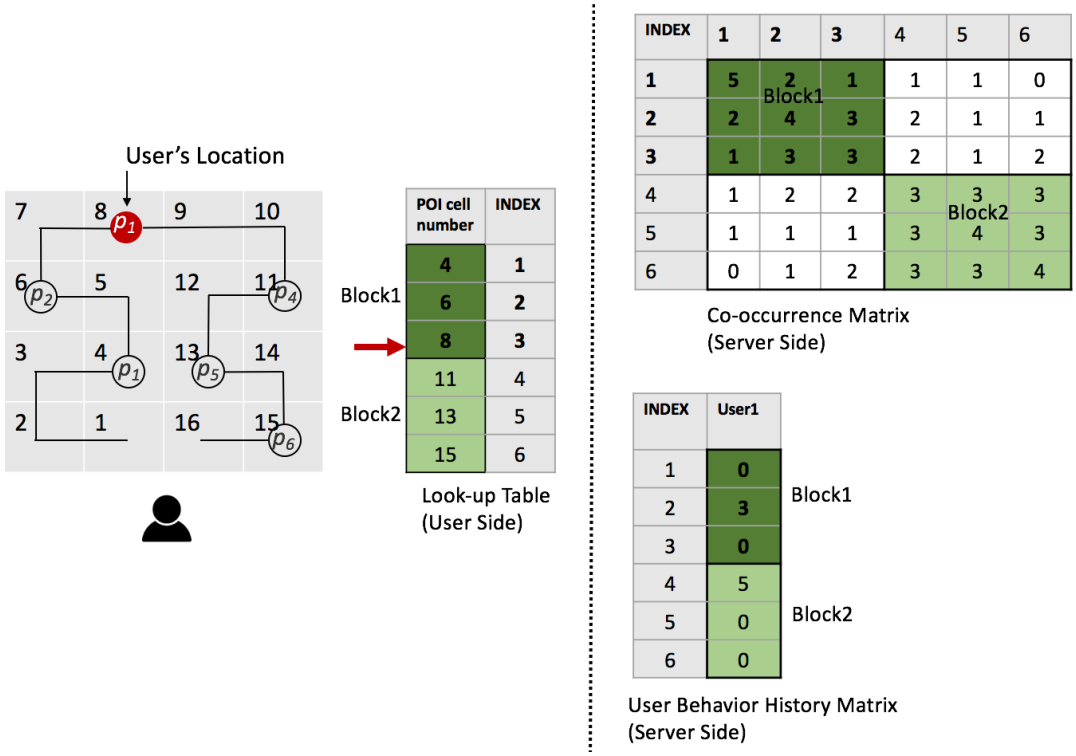


Fig.9 An example of Selection Phrase

Because the original co-occurrence matrix is large and only a small block corresponding to a user's location is needed for computation, the main task of the selection phase is to scale down the co-occurrence matrix secretly. Here, each POI is mapped to one index of the co-occurrence matrix; however, one index may include several POIs, where an index represents a small area, as shown in Fig.9.

Assume user u is in block B_u which is defined as: $B_u = \{(b_i, .., b_j) | b_i, b_j \in I,\ i \leq j\}$, where I represents for the item INDEX set.

User u generates two selection matrixes according to the block B_u. One is for selecting block B_u from co-occurrence matrix denoted as SM_{CO}. The other is for selecting block B_u from user history behavior matrix denoted as SM_{UB}.

$$SM_{CO}[i][j] = \begin{cases} \varepsilon(1), & if\ all\ the\ index\ between\ b_i\ to\ b_j \in B_u \\ \varepsilon(0), & otherwise \end{cases}$$

$$SM_{UB}[i] = \begin{cases} \varepsilon(1), & if\ all\ the\ index\ between\ b_i\ to\ b_j \in B_u \\ \varepsilon(0), & otherwise \end{cases}$$

Next, Enc(SM_{CO}) and Enc(SM_{UB}) will be sent to the PPRS. Co-occurrence Matrix will be multiplied by Enc(SM_{CO}) while User History Behavior Matrix will be multiplied by Enc(SM_{UB}).

An example of selecting block 1 is shown in Fig.9 and Fig.10:

In Fig.9, a map is divided into 2 blocks (b1, b2). The user is in cell number 8, which belongs to block 1. Thus, the user generates the encrypted selection matrix shown in Fig.10 (b) and sends it to the PPRS. After the multiplication of these two encrypted matrixes Fig10(a)(b), block 1 can be secretly retrieved.

Algorithm : Generate Selection Matrix

Input: Co-occurrence Matrix CM, target user's selected Block $B_u = \{(b_i, .., b_j) | b_i, b_j \in I, i < j, I: \text{item INDEX set}\}$

// the size of CM is |I|*|I|

Output: Selection Matrix SM_{CO} and SM_{UB}

// SM_{CO} is the selection matrix for retrieving B_u from co-occurrence matrix CM

// SM_{UB} is the selection matrix for retrieving B_u from the user's behavior history matrix BM_u

function GenSelectionMatrix (CM,B_u,I)

```
11. Assign  SMCO[1..|I|][1..|I|] = 0
12. for i=1 to |I|
13.     for j=1 to |I|
14.       if  (i∈ Bu and j∈ Bu)
15.           SMCO[i][j]=ε(1)
16.       else
17.           SMCO[i][j]=ε(0)
18.       endif
19.     endfor
20. endfor
21. Assign  SMUB [1..|I|]= 0
22. for i=1 to |I|
23.     if  (i∈ Bu)
24.         SMUB [i]=ε(1)
25.     else
26.         SMUB [i]=ε(0)
```

---

27. **endif**

28. **endfor**

29. **return** SM_{CO}, SM_{UB}

4.2.3 Recommendation Phase

In the previous selection phase, the selected matrix M_{SCO} from the co-occurrence matrix and M_{SUB} from the user history behavior matrix are generated. After multiplication and addition of M_{SCO} and M_{SUB}, the primary stage of recommendation result will be generated.

In Section 2.4, we mentioned that the similarity between items i and j is defined as below:

$$sim(i,j) = \frac{|N_{(i)} \cap N_{(j)}|}{N_{(i)}}$$

The primary stage of recommendation computation just gets the result of $|N_{(i)} \cap N_{(j)}|$ homomorphically instead of $\frac{|N_{(i)} \cap N_{(j)}|}{N_{(i)}}$ because HElib applied cannot directly support division or floats at the present time. In addition, the sorting calculation of the recommendation result over encrypted data has technical difficulties and incurs overhead computation cost.

Therefore, in our proposed the method, the PSP, which holds the privacy key, will implement the final stage of recommendation and sorting calculation after decrypting the primary recommendation result by the private key.

INDEX	1	2	3	4
1	**ε(3)**	**ε(2)**	ε(1)	ε(1)
2	**ε(2)**	**ε(3)**	ε(3)	ε(1)
3	ε(1)	ε(3)	ε(1)	ε(2)
4	ε(1)	ε(1)	ε(2)	ε(3)

(a) Co-occurrence matrix of POI

X

(elementwise product)

INDEX	1	2	3	4
1	**ε(1)**	**ε(1)**	ε(0)	ε(0)
2	**ε(1)**	**ε(1)**	ε(0)	ε(0)
3	ε(0)	ε(0)	ε(0)	ε(0)
4	ε(0)	ε(0)	ε(0)	ε(0)

(b) Selection matrix from User

INDEX	1	2	3	4
1	**ε(3)**	**ε(2)**	ε(0)	ε(0)
2	**ε(2)**	**ε(3)**	ε(0)	ε(0)
3	ε(0)	ε(0)	ε(0)	ε(0)
4	ε(0)	ε(0)	ε(0)	ε(0)

(c) The selected matrix of co-occurrence matrix

Fig.10 An example of generating the recommendation result

Algorithm : Generate The Primary Recommendation Result

Input: Co-occurrence Matrix CM', User Behavior History Matrix BM'_u, Item INDEX set I

Output: Recommendation Result Array Re

// Re[j] represents item j's primary recommendation score without dividing $N_{(i)}$ after adopting item-based collaborative filtering with the target user's behavior history BM'_u

function Recommend (CM', BM'_u, I)

1. **Assign** Re[1..|I|] = 0
2. **for** j=1 to |I|
3. **for** i=1 to |I|
4. Re[j]+=CM'[j][i]* BM'_u [j]
5. **endfor**
6. **endfor**
7. **return** Re

4.2.4 Encrypted Database Update

Updating the ED is also one of the novel parts of the proposed protocol. In the proposed model (Fig.5), steps 4–6 (4. Choose the interested POI, 5. Enc(check-in item), 6. Update the database) show how the updating part works. The user chooses the desired POI p from the recommendation lists and generates the co-occurrence matrix p combining with the user's own behavior history saved in the user's endpoint (step 4), encrypt this co-occurrence matrix and send it back to the PPRS (step 5). The PPRS processes the updating calculation between the ciphertexts from the user and the encrypted data in the database.

An example of updating the ED is shown as below:

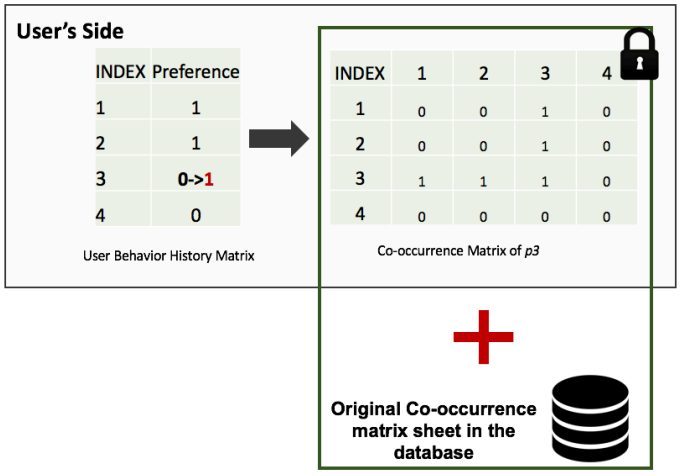


Fig.11 An example of updating encrypted database

5. Experimental Evaluation

5.1 Experimental Environment

We implemented our proposed protocol with the dataset in C++. As for the encryption part, we used HElib, which is based on BGV full homomorphic scheme.

The experiments were executed on two machines. The client side had configurations of 64-bit CentOS 6, Intel Xeon E7-8880 v3 @ 2.30 GHz x 72, and 1 TB memory, the server side had configurations of 64-bit CentOS 6.7, Intel Xeon CPU E5-2643 v3 @ 3.40 GHz and 512 GB RAM. 10-Gbps Ethernet connected these two machines.

5.2 Dataset

In this work, we only consider sensitive POIs such as hospital, clinic, church, and LGBT bar. Fig. 12 includes the survey result of Medical Institution Information of Tokyo from Japan Medical Analysis Platform. Moreover, according to Agency for Cultural Affairs, and GClick, which is a LGBT bar guide in Japan, the number of LGBT bars and churches in Tokyo are 637 and 833, respectively. As we can see, the number of sensitive POIs (medical institution) in

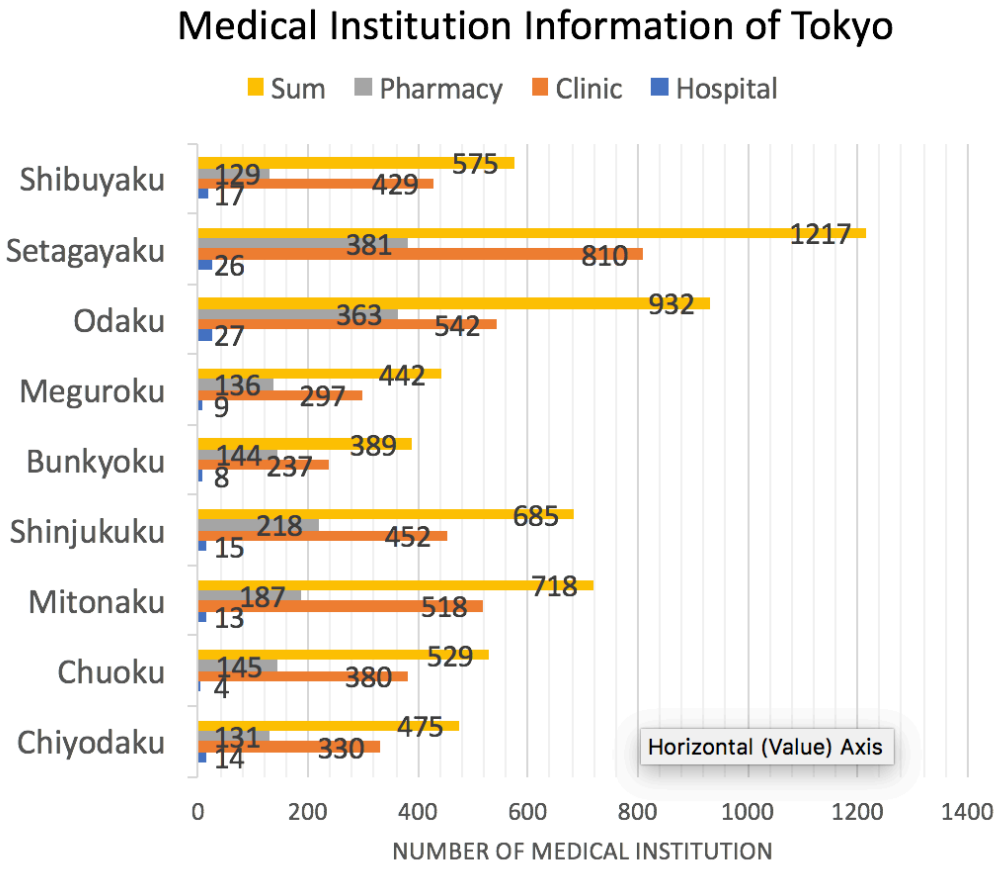


Fig.12 The data of medical institution Information in Tokyo

each ward of Tokyo is between 500 and 1300. The number of LGBT bars or churches in the entire Tokyo area is not more than 1000.

Therefore, we used an artificial dataset with dimension N ranging in {5,10,20,40,80,100,1000}. In this situation, which ward of Tokyo the user is located in can be protected from the PPRS.

5.3 Experiment Results and Evaluation

TABLE 2 THE PARAMETER OF HELIB

M	L	#slot	security	Plaintext Space
16384	10	4096	132	8191^3

The experiments consist of two parts: the selection phrase and recommendation phrase, both of which need real-time computing. As for the ED update part, since it can be finished offline regularly, the efficiency does not have a significant impact on user experience. Table 2 lists the parameters of the HElib used in this experiment. Table 2 lists the parameters of the HElib.

Selection phrase:

In the selection phrase, the item-based co-occurrence matrix CM and user's behavior history matrix BM_u is element-wisely multiplied by each selection matrix. Here, the SV packing technique is adopted to reduce the number of ciphertexts which converts a vector of integers to one ciphertext. Thus, multiplication by N times is executed in the selection phrase, where N is the size of the co-occurrence matrix CM. Table 3 shows the exeution time of selection phase.

Recommendation phrase:

In the recommendation phrase, the multiplication between CM' and BM'_u is executed. Table 4 shows the execution time of recommendation phrase.

TABLE 3 THE EXECUTION TIME OF SELECTION PHASE

#of elements	Encryption[s]	Selection[s]	SUM[s] (Selection+Encryption)
5	0.05	0.06	0.11
10	0.10	0.12	0.22
20	0.21	0.24	0.45
40	0.34	0.48	0.82
80	0.67	0.95	1.62
100	0.86	1.19	2.05
1000	8.45	11.87	20.32

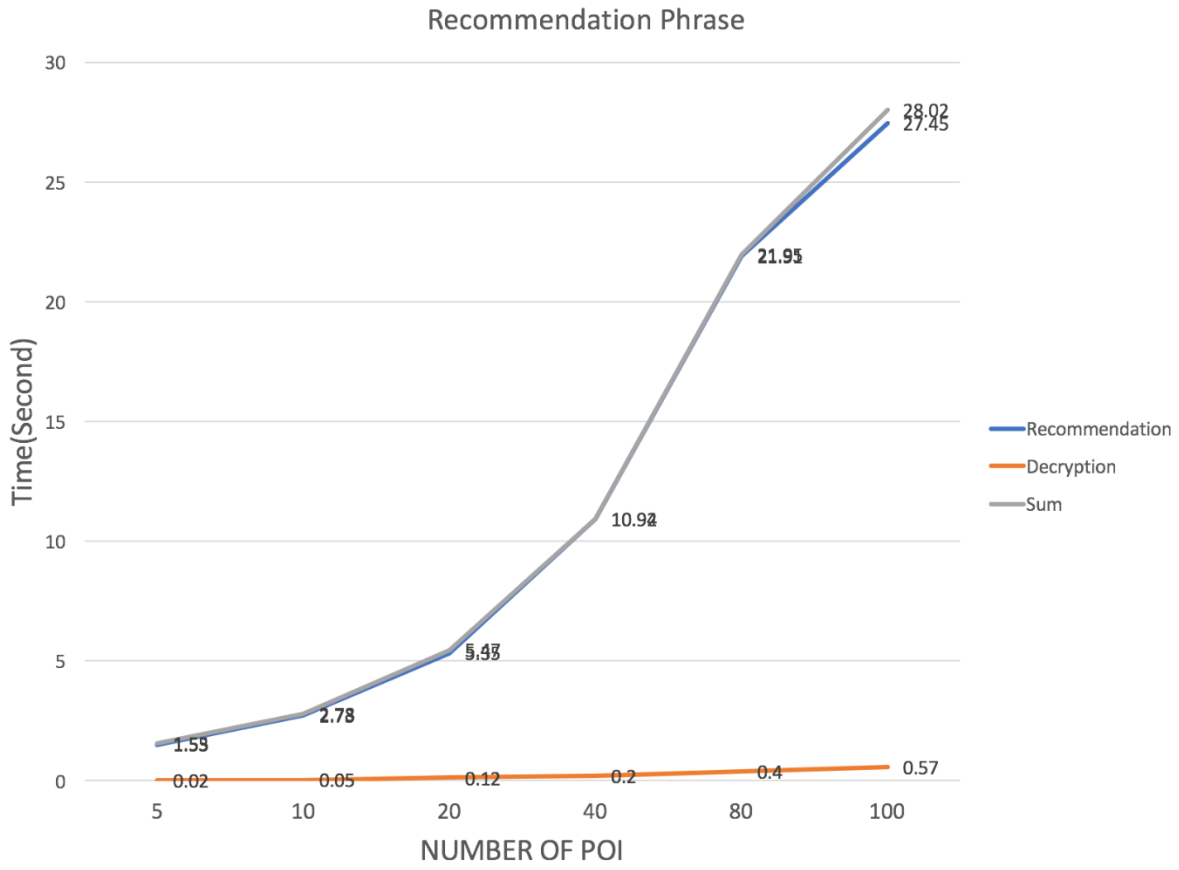
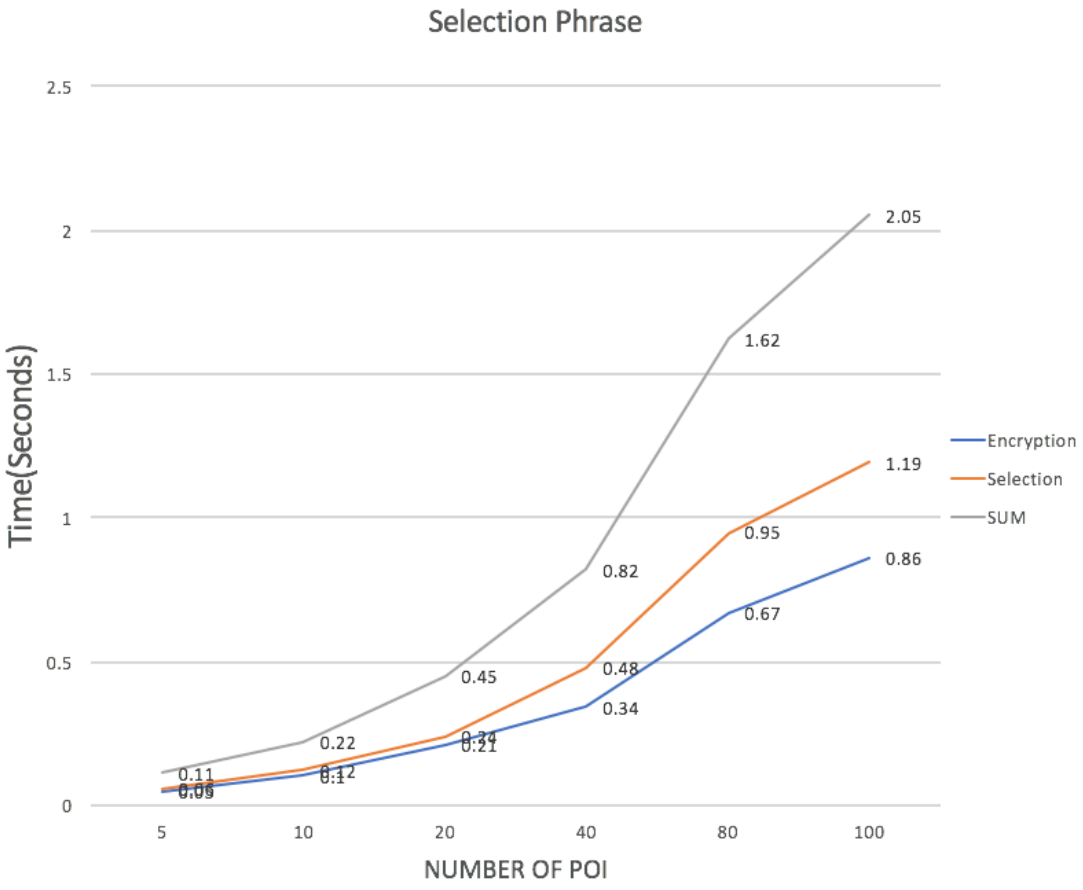


TABLE 4 THE EXECUTION TIME OF RECOMMENDATION PHASE

#of elements	Recommend[s]	Decryption[s]	SUM[s] (Recommend+Decryption)
5	1.53	0.01	1.55
10	2.73	0.05	2.78
20	5.35	0.12	5.47
40	10.92	0.20	11.12
80	21.91	0.40	22.31
100	27.45	0.57	28.02
1000	263.96	5.50	284.28



5.4 Communication Cost

Since the slot size l is 4096 and size of the co-occurrence matrix of the POI is N, the user will send two encrypted selection matrices (one is to select the user-based co-occurrence matrix and the other is to select user behavior history matrix) to the server, and the communication cost is O(Nl).

TABLE 5 THE EXPERIMENT RESULT OF COMMUNICATION SIZE

#of elements	Client -> Server Selection Matrix[MB]	Server -> Client Recommendation Result[MB]
5	6.8	8
10	14.0	16
20	27	32
40	54	64
80	108	127
100	135	169
1000	1433.6	1638.4

5.5 Memory Cost

In table 6, we listed both virtual memory usage and resident memory usage of both the selection phrase and the recommendation phase.

TABLE 6 THE EXPERIMENT RESULT OF MEMORY SIZE

	Selection Phase		Recommendation Phase	
#of elements	Virtual Memory Usage [MB]	Resident Memory Usage [MB]	Virtual Memory Usage [MB]	Resident Memory Usage [MB]
5	543	521	746	725
10	650	629	763	742
20	770	749	783	762
40	829	808	810	789
80	889	868	870	849
100	953	932	901	880
1000	2814	2765	2283	2253

5.6 Security Proof

In the proposed model, PSP (Privacy Service Provider) is trusted and PPRS (Privacy-Preserving Recommendation LBS Server) is honest-but-curious. Only the PSP holds the private keys which means no one can decrypt the message except PSP. After the PSP decrypting the recommendation result from the PPRS, the result in plaintext will be sent to the user by Secure Channel. In addition, the database is also encrypted. Thus, the malicious users are not able to learn any information even though database were leak.

6.Conclusion

In this paper, we proposed the protocol of privacy-preserving recommendation on the location-based service by combining three main techniques. They are Hilbert curve, collaborative filtering recommender based on the co-occurrence matrix and FHE. Compared with the previous protocol, which concentrated only on privacy-preserving k-NN search, the novelty of our proposed protocol is that the protocol can benefit both service users and service providers, which means it has commercial value and can be applied practically. It not only protects users' information from being disclosed to malicious parties, but also provides the newest recommendation result by the database which can be updated.

The preliminary evaluation showed that the process of our proposed privacy-preserving recommendation for LBS with 1,000 POIs requires 304.6 seconds. It is still long for the use of commercial recommendation services so that further speedup, larger scalability, and higher efficiency are our future work. Besides, in this implementation, a naïve recommendation was adopted and only the item-based collaborative filtering within the same block, i.e., geographical area, is considered. However, we may extend our scheme to item-based collaborative filtering by using whole POIs relationship to have more accurate recommendation result.

7.References

[1] C. Gentry, “Fully Homomorphic Encryption Using Ideal Lattices,” Stoc'09: Proceedings of the 2009 Symposium on Theory of Computing, pp. 169-178, 2009.

[2] R. L. Rivest, L. Adleman, and M. L. Dertouzos, “On data banks and privacy homomorphisms,” Foundations of secure computation, vol. 4, no. 11, pp. 169-180, 1978.

[3] C. Gentry, and D. Boneh, A fully homomorphic encryption scheme: Stanford University Stanford, 2009.

[4] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) fully homomorphic encryption without bootstrapping,” ACM Transactions on Computation Theory (TOCT), vol. 6, no. 3, pp. 13, 2014.

[5] C. Gentry, S. Halevi, and N. P. Smart, “Fully Homomorphic Encryption with Polylog Overhead,” Advances in Cryptology - Eurocrypt 2012, vol. 7237, pp. 465-482, 2012.

[6] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical GapSVP," Advances in cryptology–crypto 2012, pp. 868-886: Springer, 2012.

[7] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," Advances in Cryptology–CRYPTO 2012, pp. 643-662: Springer, 2012.

[8] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," Advances in Cryptology–CRYPTO 2013, pp. 75-92: Springer, 2013.

[9] N. P. Smart, and F. Vercauteren, “Fully homomorphic SIMD operations,” Designs, codes and cryptography, vol. 71, no. 1, pp. 57-81, 2014.

[10] Z. Brakerski, and V. Vaikuntanathan, “Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages,” Advances in Cryptology - Crypto 2011, vol. 6841, pp. 505-524, 2011.

[11] S. Halevi, and V. Shoup, “Algorithms in HElib,” Advances in Cryptology - Crypto 2014, Pt I, vol. 8616, pp. 554-571, 2014.

[12] H. Sagan, Space-filling curves: Springer Science & Business Media, 2012.

[13] B. Moon, H. V. Jagadish, C. Faloutsos, and J. H. Saltz, “Analysis of the clustering properties of the Hilbert space-filling curve,” IEEE Transactions on knowledge and data engineering, vol. 13, no. 1, pp. 124-141, 2001.

[14] H. V. Jagadish, “Linear clustering of objects with multiple attributes,” Proceedings of the 1990 Acm Sigmod International Conference on Management Data, vol. 19, pp. 332-342, 1990.

[15] A. Khoshgozaran, and C. Shahabi, “Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy,” Advances in Spatial and Temporal Databases, Proceedings, vol. 4605, pp. 239-257, 2007.
[16] I. T. Lien, Y. H. Lin, J. R. Shieh, and J. L. Wu, “A Novel Privacy Preserving Location-Based Service Protocol with Secret Circular Shift for k-NN Search,” IEEE Transactions on Information Forensics and Security, vol. 8, no. 6, pp. 863-873, Jun, 2013.
[17] Y. Utsunomiya, K. Toyoda, I. Sasase, and Ieee, "LPCQP: Lightweight Private Circular Query Protocol for Privacy-Preserving k-NN Search," IEEE Consumer Communications and Networking Conference. pp. 59-64, 2015.
[18] S. Owen, “Mahout in action,” 2012.
[19] T. Dunning, and E. Friedman, “Practical Machine Learning: Innovations in Recommendation,” 2014.
[20] M. Gruteser, D. Grunwald, and Usenix, “Anonymous usage of location-based services through spatial and temporal cloaking,” Proceedings of Mobisys 2003, pp. 31-42, 2003.
[21] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, “Private queries in location based services: anonymizers are not necessary,” Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp.121-132, 2008.
[22] S. Badsha, X. Yi, and I. Khalil, “A practical privacy-preserving recommender system,” Data Science and Engineering, vol. 1, no. 3, pp. 161-177, 2016.
[23] S. Badsha, X. Yi, I. Khalil, and E. Bertino, "Privacy Preserving User-based Recommender System," IEEE International Conference on Distributed Computing Systems. pp. 1074-1083, 2017.

Acknowledgement

I would like to express my deepest gratitude to Professor Yamana who has supervised me doing research for two years. Without his advices on my proposed protocol and the comments on my contents of thesis, I would not be able to finish this master thesis. Professor Yamana always encourages us to challenge the cutting-edge technologies and make contributions to the society which has inspired me a lot in my student life in Waseda University. I will keep it in mind and bring this passion to my work and life after graduation as well.

Also, I would like to thank my senior Yu Ishimaki for the continuous supports on the encryption algorithms and experiments. His professionalism, kindness and patience played an important role in my research.

Last but not least, I would like to thank all the lab members. Thanks for sharing your new ideas with me in the meetings. Thanks for the helping in all aspects. It is my great honor to be the lab mates with you.

Publication

LYU Qiuyi, Yu Ishimaki and Hayato Yamana: “Privacy-Preserving Recommendation on LBS”, WAHC 2018. 6-th Workshop on Encrypted Computing and Applied Homomorphic Cryptography (submitted).