SHELBRS: Location-Based Recommendation Services Using Switchable Homomorphic Encryption

Mishel Jain[1(✉)], Priyanka Singh[1(✉)], and Balasubramanian Raman[2(✉)]

[1] Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar, Gujarat, India
{201911052,priyanka_singh}@daiict.ac.in
[2] Indian Institute of Technology, Roorkee, Roorkee, Utharakhand, India
bala@cs.iitr.ac.in

Abstract. Location-Based Recommendation Services (LBRS) has seen an unprecedented rise in its usage in recent years. LBRS facilitates a user by recommending services based on his location and past preferences. However, leveraging such services comes at a cost of compromising one's sensitive information like their shopping preferences, lodging places, food habits, recently visited places, etc. to the third-party servers. Losing such information could be crucial and threatens one's privacy. Nowadays, the privacy-aware society seeks solutions that can provide such services, with minimized risks. Recently, a few privacy-preserving recommendation services have been proposed that exploit the fully homomorphic encryption (FHE) properties to address the issue. Though, it reduced privacy risks but suffered from heavy computational overheads that ruled out their commercial applications. Here, we propose SHELBRS, a lightweight LBRS that is based on switchable homomorphic encryption (SHE), which will benefit the users as well as the service providers. A SHE exploits both the additive as well as the multiplicative homomorphic properties but with comparatively much lesser processing time as it's FHE counterpart. We evaluate the performance of our proposed scheme with the other state-of-the-art approaches without compromising security.

Keywords: Homomorphic encryption · Location-Based Recommendation Services (LBRS) · Co-occurrence Matrix (CM)

1 Introduction

Location-Based Recommendation Services (LBRS) grant users access to relevant information about their surroundings based on their location and history. For instance, a person searching for a coffee shop nearby his/her location. The service providers would provide the best search results considering his/her present location and previous history. However, availing of such services risks the user's privacy as the shared sensitive information could be misused by these third-party servers to their advantage, causing serious losses to the user [15]. This fact

L. Batina et al. (Eds.): SPACE 2021, LNCS 13162, pp. 63–80, 2022.
https://doi.org/10.1007/978-3-030-95085-9_4

kind of delimits the privacy-aware society rushing to leverage such services and creates an urgent need for privacy-preserving recommendation services.

The real-time location information of the user is handled by location based services (LBS). It also provides recommendation over encrypted history preferences which ensures the user's privacy. Lyu et al. proposed one such state-of-the-art protocol. They adopt the Hilbert curve [16] as a mapping tool, collaborative filtering recommender based on the co-occurrence matrix as a recommendation technique [11,17], and Brakerski-Gentry-Vaikuntanathan (BGV) fully homomorphic encryption (FHE) as an encryption scheme [2]. However, it is still infeasible for the use of commercial recommendation services due to the high processing time.

In this paper, we propose SHELBRS, a lightweight LBRS that will benefit the users as well as the service providers. Instead of FHE, we employ switchable homomorphic encryption (SHE) that securely switches between partially homomorphic encryption (PHE) schemes. Specifically, Paillier Homomorphic Encryption and ElGamal Homomorphic Encryption for performing additions and multiplications on the encrypted data. PHE evaluates arithmetic operations more efficiently at least 2–3 order of magnitude compared to FHE. SHE supports an arbitrary number of additions and multiplications over encrypted data and serves the principle of FHE with better efficiency. The overall computation and communication cost required in switching between the PHE's is reasonable for real-life applications. It overall reduces the processing time without compromising the security.

The remainder of this paper is structured as follows: Sect. 2 discusses some of the related works. Section 3 gives an overview of the Hilbert curve, collaborative filtering based on Co-occurrence Matrix (CM), PHE, and the SHE schemes. Section 4 presents the LBRS using FHE [9] while Sect. 5 details the proposed SHELBRS scheme. Section 6 discusses the experimental results and the security analysis of the proposed scheme. Section 7 concludes the work along with some future directions.

2 Related Work

Lattice-based FHE scheme introduced by Craig Gentry in 2009, is a milestone research that opened doors for proposing possible solutions for encrypted data. It was made possible as this scheme supported computation of arbitrary functions and operations on the ciphertext, without the need of actually decrypting it [3].

Many LBS were proposed in the literature to search nearest Point of Interests (POI)'s to the user's private location. In 2003, K-anonymous based technique was introduced which adopts temporal and spatial cloaking [4]. It acquires accuracy but requires a trusted third party to hide the user's location. Private Information Retrieval (PIR) [10], Private Circular Query Protocol (PCQP) [6] and Lightweight Private Circular Query Protocol (LPCQP) [19] are the LBS based on the cryptography methods. In PIR scheme, the user receives POI from the server's database based on Quadratic Residuosity Assumption (QRA) without server's knowledge of which POI a user is interested in. It provides security

but takes high execution time for searching POIs. PCQP proposed by Lien et al. is an effective k-NN search algorithm based on Paillier cryptosystem and Hilbert curve. It secretly shifts the POI-info circularly which is stored on the server. LPCQP, proposed by Utsunomiya et al. is a lightweight protocol that removes unnecessary POI information from the requesting user to reduce computational cost. PIR, PCQP, and LPCQP are secure against single point failures and Denial of Service (DOS) attacks. In 2012, Pingley et al. proposed a context-aware scheme for privacy-preserving LBS [13]. It projects the user's location on various-grid-length Hilbert curve and uses location perturbation technique to prevent user's privacy from the LBS server. Gang et al. proposed location-based social network (LBSN) towards privacy preservation for "check-in" services in 2019 [18]. It designs the framework using k-anonymity based algorithms without using a trusted third-party server. It guarantees secure access and preserves user's location privacy.

The detail of recommender systems and discussion about different recommendation algorithms based on traditional and network approaches was studied by Lu et al. in 2012 [8]. It compares the performance of different recommendation algorithms. Badsha et al. introduced an Elgamal cryptosystem based privacy-preserving item-based Collaborative Filtering (CF) in 2016 [1]. It provides recommendations based on the user's average ratings and the similarities between the items. Zhang et al. proposed Factorization Machines (FM) based recommendation algorithm for Rural Tourism in 2018 [20]. It provides recommendations based on geographical distribution and seasonal features such as the user's best suitable season for traveling, how many kilometers is the traveling spot away from the city, and other user's reviews or comments for the particular location. In 2018, Horowitz et al. proposed a mobile recommender system named "EventAware" for the events [5]. It provides recommendations on the basis of both context-aware and tag-based algorithms. In 2019, Qi et al. proposed a time-aware and privacy-preserving distributed recommendation service based on a locality-sensitive hashing (LSH) to provide most accurate recommended results [14]. Papakyriakopoulos et al. analyzed the political networks based on hybrid collaborative filtering and deep learning recommender algorithms in 2020 [12]. It shows how hyperactive users influence recommendations. It also compares the results based on likes and comments with and without the inclusion of the hyperactive users along with the rest of the users in the datasets.

Combining both location privacy and privacy-preserving recommendations, Lyu et al. proposed privacy-preserving recommendations for LBS in 2019 [9]. It provides suggestions over encrypted previous histories considering user's live location data. This protocol uses one trusted third party server where sensitive data such as crypto keys are stored and one honest but curious server where all the computations are performed. Compared to Lyu et al. protocol, the proposed SHELBRS protocol requires less computation cost as it uses SHE instead of FHE and maintains security while sharing sensitive data between the servers.

3 Preliminaries

3.1 Hilbert Curve

Hilbert curve is a mapping tool that is used to transform 2-D space into 1-D space. It preserves the adjacency of the neighboring points and has best clustering properties. In order to preserve the adjacency property, the orientation is not retained [11,16]. As we increase the order of a pseudo-Hilbert curve, a given point on the line converges to a specific point. The two data points which are close to each other in 2-D space are also close to each other after mapping into 1-D space.

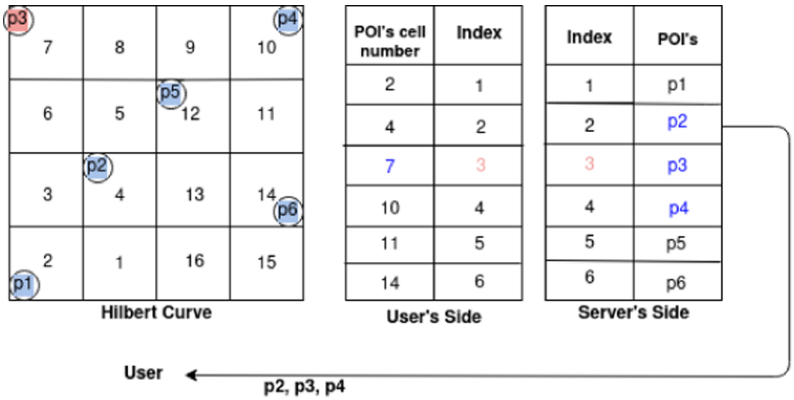

Fig. 1. The combination of continuous Hilbert space-filling curve and location-based services

Figure 1 shows a Hilbert curve over the landscape, storing the POI look-up table at the user end and the POI information table at the server end. The user $p3$ is located on cell 7 and he/she is requesting POIs from the server. At the user's side, p3 is located at Index-3 and after sending index information to the server, the server sends nearby POIs i.e. p2, p3, p4 corresponding to the current index back to the user. This is how the nearest POIs to the user's location is suggested when the landscape is mapped on the Hilbert curve. However, sending such information in plaintext does not ensure the user's privacy.

3.2 Collaborative Filtering (CF) Recommender Based on Co-occurrence Matrix (CM)

CF Recommender consists of two well-known algorithms: user-based CF and item-based CF. We used an item-based CF recommender as similarities between items are more stable than that of users. It finds the similarity between items and provides the best recommendation. Some E-commerce websites such as Amazon

provide recommendations such as "A person that bought product A also bought product B" or "A person that liked the cafe A also liked cafe B".

CM contains the visited POIs information. It computes the number of times each pair of items occurs together in the user-item inversion list. To generate CM, the first step is to generate a user-item inversion list and the second step is to traverse the list and follow the algorithm as described:

- $CM[i][j](i! = j)$ is increased by 1 if item i and the item j are in the same user's inversion list.
- $CM[i][j](i == j)$ is increased by 1 for every item i.

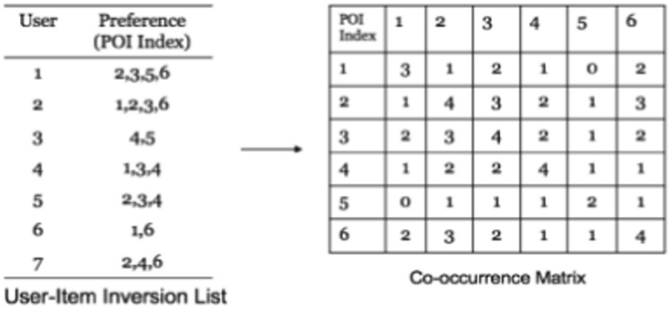

Fig. 2. Formation of the final CM [9]

Figure 2 shows the formation of CM based on the user-item inversion list. According to the user-item inversion list, $User_1$ has the preference for indices 2, 3, 5, 6. The value at all possible pairs of indices in CM such as $CM[2][3]$, $CM[2][5]$, $CM[2][6]$, $CM[3][2]$, $CM[3][5]$, $CM[3][6]$, $CM[5][2]$, $CM[5][3]$, $CM[5][6]$, $CM[6][2]$, $CM[6][3]$, $CM[6][5]$ is incremented by 1. Every time an item occurs in the list, the value in CM such as $CM[2][2]$, $CM[3][3]$, $CM[5][5]$, and $CM[6][6]$ is incremented by 1. Likewise for all the users, the above algorithm is performed to get the final CM.

3.3 Partially Homomorphic Encryption (PHE)

PHE schemes are a kind of encryption schemes that allow only certain types of operations on the encrypted data. If we decrypt the processed encrypted data, the results would be the same as if calculated over the corresponding plaintext values. Based on the type of operations supported, it can be categorized as additive PHE or multiplicative PHE. For instance, Paillier is an example of additive PHE and ElGamal, an example of multiplicative PHE. We will briefly describe each of them along with their homomorphic properties.

Paillier Encryption as Additive PHE

- **KeyGen($1^n, +$):** On input a security parameter 1^n, the algorithm chooses (N, p, q) where $N = p * q$, p and q are n bit primes, and $\phi(N) = (p-1) * (q-1)$. The Paillier ADD scheme public-private key pair:

$$\langle pk^+, sk^+ \rangle = \langle N, (N, \phi(N)) \rangle \tag{1}$$

- **Enc(pk^+, m):** The algorithm takes a plaintext m and a public key N as input. It chooses a random $r \in Z_N^*$ and outputs the ciphertext:

$$c^+ = (1+N)^m.(r)^N \mod N^2 \tag{2}$$

- **Dec(sk^+, c^+):** The algorithm takes a ciphertext c^+ and a private key $(N, \phi(N))$ as input and outputs the message:

$$m = \frac{((c^+)^{\phi(N)} \mod N^2) - 1}{N}.\phi(N)^{-1} \mod N \tag{3}$$

- **Paillier homomorphic properties:**
  1. **Addition:** The product of two encrypted ciphertexts results in the sum or addition of their corresponding plaintexts:

$$Dec(E^+(m_1) * E^+(m_2) \mod N^2) = (m_1 + m_2) \mod N \tag{4}$$

  2. **Scalar Multiplication:** Raising a scalar to the power of encrypted ciphertext results in the product of the scalar and the corresponding plaintext:

$$Dec(E^+(m_1)^k) \mod N^2) = (k * m_1) \mod N \tag{5}$$

ElGamal Encryption as multiplicative PHE

- **KeyGen($1^n, *$):** On input a security parameter 1^n, the algorithm chooses (N, p, q) where $N = p * q$, p and q are n bit primes, considers g as square value and sets $g = 16$. It also chooses a random odd number x, and sets $h = g^x \mod N$. The ElGamal MUL scheme public-private key pair:

$$\langle pk^*, sk^* \rangle = \langle (N, g, h), (N, g, x) \rangle \tag{6}$$

- **Enc(pk^*, m):** The algorithm takes a plaintext m and a public key (N, g, h) as input. It chooses a random $r \in Z_N^*$ and outputs the ciphertext:

$$c^* = \langle c_1^*, c_2^* \rangle = \langle mh^r, g^r \mod N \rangle \tag{7}$$

- **Dec(sk^*, c^*):** The algorithm takes a ciphertext c^* and a private key (N, g, x) as input and outputs the message:

$$m = \frac{c_1^*}{(c_2^*)^x} \mod N \tag{8}$$

- **ElGamal homomorphic properties:**
  1. **Multiplication:** The product of two encrypted ciphertexts results in the multiplication of their corresponding plaintexts:

$$Dec(E^*(m_1) * E^*(m_2) \mod N^2) = (m_1 * m_2) \mod N \tag{9}$$

3.4 Switchable Homomorphic Encryption (SHE)

We work with a variant of ElGamal MUL scheme E^* and Paillier ADD scheme E^+. ElGamal MUL scheme uses a large composite modulus i.e., $N = p * q$, where p and q are large primes. Both the partially homomorphic schemes share the same modulus. We consider two servers, say a server and a proxy. The algorithms used in the SHE scheme are described as follows:

- **KeyGen(1^n):** On input a security parameter 1^n, the algorithm outputs (N, p, q), where $N = p * q$, p and q are n bit primes, and $\phi(N) = (p-1)(q-1)$. The Paillier ADD scheme public-private key pair:

$$\left\langle pk^+, sk^+ \right\rangle = \langle N, (N, \phi(N), p, q) \rangle \tag{10}$$

  It also chooses the generator $g = 16$, two random odd numbers $x_0, x_1 \in Z_N^*$ where $|x_0| \approx |x_1| < 1/2\,|N|$. It sets $x = x_0 x_1$ and $h = g^x$. The ElGamal MUL scheme public-private key pair:

$$\langle pk^*, sk^* \rangle = \langle (N, g, h), (N, g, x_0, x_1) \rangle \tag{11}$$

- **Enc(pk^o, m):** The algorithm runs E^+ encryption scheme if o is '+' else runs E^* encryption scheme.
- **Dec(sk^o, c^o):** The algorithm runs E^+ decryption scheme if o is '+' else runs E^* decryption scheme.
- **KeyShaGen(sk^+):** The algorithm sets both the secret key shares k_0^+ (proxy) and k_1^+ (server) to NULL.
- **KeyShaGen(sk^*):** The algorithm sets both the secret key shares k_0^* (proxy) and k_1^* (server) to x_0 and x_1 respectively.
- **AddToMul(c^+, pk^*):** The algorithm is run locally by the server. Given an ADD ciphertext of the form:

$$E^+(m) = (1+N)^m.(r')^N \mod N^2 \tag{12}$$

  and the MUL public key $pk^* = (N, g, h)$, the algorithm chooses a random $r \in Z_N^*$ and outputs the encrypted MUL ciphertext:

$$E^+(E^*(m)) = \left\langle (1+N)^{mh^r}.(r')^{Nh^r} \mod N^2, g^r \right\rangle \tag{13}$$

- **MulToAdd(c^+, k_0^*, k_1^*):** The algorithm is jointly run by a server and a proxy. On input an encrypted MUL ciphertext of the form (13) the server chooses a random $s \in Z_N^*$, computes

$$c' = (g^{r+s})^{k_1^*}, R = g^s \tag{14}$$

  and forwards (13) and (14) to the proxy.
  The proxy then using its key shares k_0^*, computes $(c')^{k_0^*} = h^{r+s}$ and finds its inverse $(h^{r+s})^{-1}$. It also computes and returns

$$\begin{aligned} c'' &= ((1+N)^{mh^r}.(r')^{Nh^r})^{h^{r+s-1}} \mod N^2 \\ &= (1+N)^{mh^{-s}}.(r')^{Nh^{-s}} \mod N^2 \\ &= E^+(mh^{-s}) \end{aligned} \tag{15}$$

and $R' := R^{k_0^*}$ to the server.
Finally, the server computes $(R')^{k_1^*} = h^s$ and recovers the corresponding ADD ciphertext $E^+(m)$ by homomorphically removing h^{-s} from c''.

4 Lyu et al.'s Protocol

In this section, we give an overview of Lyu et al. protocol which was meant to recommend services based on the current location of a user and his past behavior without compromising his privacy [9]. It solved the problems existing in the state-of-the-art privacy-preserving algorithms that were based on k-NN technique for searching POI's [6]. The major bottleneck was that the recommendation service didn't consider the user's past behavior while recommending any services that ultimately resulted in failing to attract the user's usage of the recommendation system. Another major demerit was it lacked any benefits for the service providers facilitating such services as the private keys were available only to the user and hence, the service providers could not extract any information from the user's data towards making their profits.

Lyu et al. resolved the aforementioned issues using collaborative filtering technique that works on top of database encrypted using FHE, besides encrypting the user's location and preferences. Also, it allowed the service providers to extract some aggregate information based on the user's data via an introduction of a Privacy Service Provider (PSP) that generates and holds the private keys. This increased the commercial value of the recommendation service but still the need of heavy computational resources required for FHE restricts the usage.

4.1 System Model

An overview of the protocol is shown in Fig. 3. It involves three main components:

- **Privacy-Preserving Recommendation Server (PPRS):** PPRS is a semi-trusted i.e. honest but curious entity that is responsible for finding the nearest POIs to the user, by considering the similarity between the POIs near the user and his current location. It performs two main tasks: The first task is to calculate the recommendation list based on preference vector PV and co-occurrence matrix CM. Here, PV provides a rating of a user for a particular item and CM describes the similarities between the items. The second task is to calculate the aggregated user behavior over the encrypted database (ED).
- **Privacy Service Provider (PSP):** PSP is a trusted third party which makes a profit from user's behavior statistics. It holds and generates the private and public key pairs. It is responsible for providing public keys to users, ED and PPRS whenever the requests arrive. It generates partial recommendation list based on user's location information.
- **Encrypted Database (ED):** It stores CM encrypted by FHE.

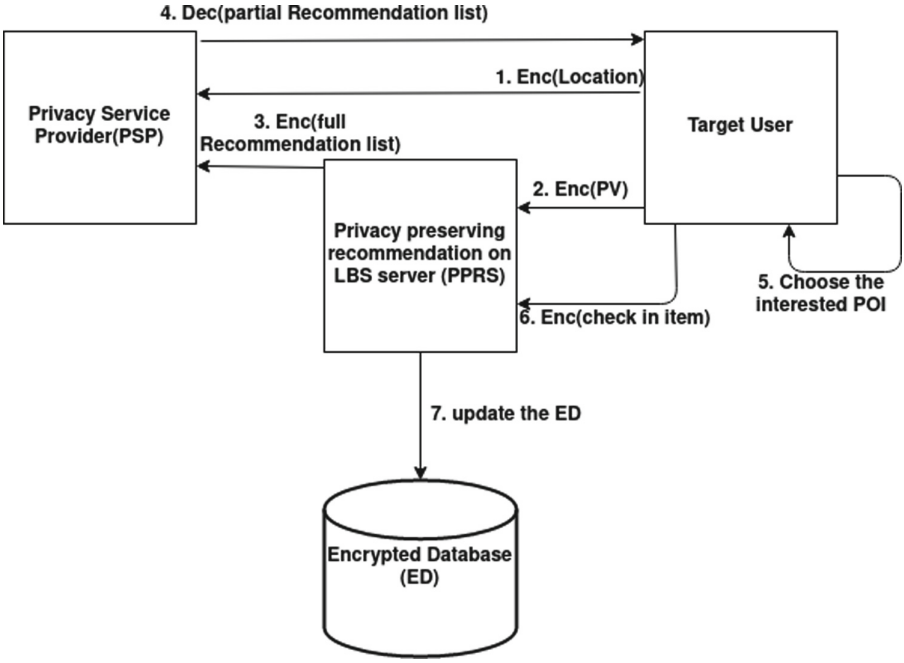

Fig. 3. An overview of Lyu et al.'s protocol

4.2 Description of Lyu et al.'s Protocol

This section describes Lyu et al.'s protocol. First, we describe briefly the three main phases of the protocol and then go for a detailed step-by-step description. The three main phases are as follows:

Initialization Phase: Personal co-occurrence matrix (CM_u), which contains the information of visited POIs, is generated by the user on the basis of his/her preferences. Each user sends his/her encrypted CM_u to the PPRS. PPRS constructs the final CM by combining these $CM'_u s$. The operation performed here is exploiting the homomorphic addition property of FHE operation. This combined matrix CM is stored in the ED.

Recommendation Phase: For computing full recommendation list for the user, each item's prediction value is computed by performing homomorphic addition and multiplication property of FHE. Prediction value is derived as

$$P_{u.i} = \sum_{j \in N(u)} (w_{ij} * r_{uj}) \tag{16}$$

where,

$N(u)$ denotes all the items, w_{ij} is the similarity item i and j, and r_{uj} is the rating of user u for item j.

ED Updation Phase: ED stores the CM and it needs to be updated according to the user's new behaviors. The updation occurs as follows:

- After receiving recommendation results from PSP in plain text, the target user selects any POI of his/her choice.
- He/She then sends the encrypted results to the ED for further updated recommendation.
- Instead of sending the whole CM, each user sends only the difference from its original CM to update the matrix with the latest information. It protects the privacy of the user from PPRS by sending the matrix in an encrypted FHE domain.

The detailed step-by-step description of the Lyu et al.'s protocol is as follows:

Step 1: Initially, the public key (p_k) is distributed by the PSP to a user and PPRS. The target user sends her/his encrypted location to PSP.
Step 2: The target user also sends encrypted preference vector to PPRS at the same time.
Step 3: PPRS generates and sends the encrypted full recommendation list to PSP.
Step 4: After PSP decrypts both the target user's location and the full recommendation list, it scans the whole list and generates the partial recommendation list according to the user's location information. It ensures user's privacy by sending it through an encrypted channel to the target user.
Step 5: The target user selects the POI from the partial recommendation list.
Step 6: She/he sends the encrypted result to the PPRS.
Step 7: PPRS then updates the ED for providing the best recommendation.

5 Proposed SHELBRS Protocol

In our proposed framework, we replace the FHE component with SHE to minimize the overall computational complexity and also, speed up the entire process so that it could be better suited for real-life scenarios. The security of the proposed protocol is kept at par with the corresponding FHE based protocol. Our protocol does not make use of any trusted third party server to store crypto keys. It simply sends the secret key shares between the servers so that an individual server cannot leak any user's sensitive information.

The details of each stage of SHELBRS protocol is discussed as follows:

5.1 Setup Stage

The setup of SHELBRS is based on a client-server architecture. We consider two servers, say server X and server Y, and the interaction between the servers or a server and a client is shown in Fig. 4. Security holds as long as at least one of the servers is honest i.e. they do not collude by sharing cryptographic keys.

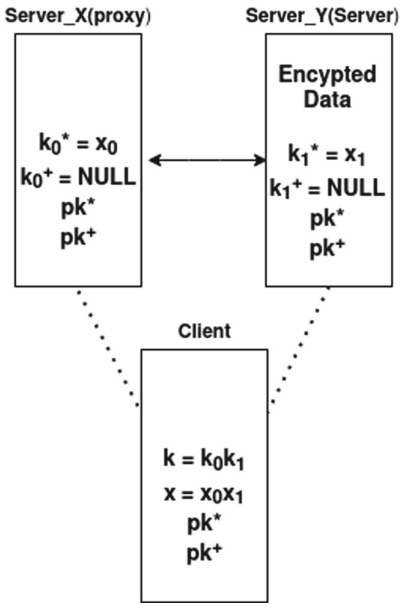

Fig. 4. Secure computation via two servers

Let us assume that the landscape I is mapped on a Hilbert curve and divided into indices $I_1, I_2, \ldots I_n$. Based on a client-server model, a client is located on one of the indices and has its own preference vector. The encrypted database of CM is already stored at server Y. A client generates a public-private key pairs using (10) (11) and sends public keys pk^+ and pk^* to both the servers. Clients uses KeyShaGen(sk^*) and KeyShaGen(sk^+) algorithms as described in Sect. 3.4 and sends k_0^* and k_0^+ to server X and k_1^* and k_1^+ to server Y. The POIs nearby user's location is recommended by the computations which are being performed on the servers.

5.2 Initialization Stage

This stage describes the steps which needs to be computed before the client starts executing his/her role.

- Personal co-occurrence matrix (CM_u) contains the information of visited POIs. During the initialization stage, each user generates his/her personal CM_u based on initial users' preference.
- An user u sends CM_u, which is encrypted by the public key pk^+, to the server Y.
- The end task is to merge all CM_u to generate the final CM. This requires paillier homomorphic Addition property as described in (4) and AddToMul algorithm as discussed in Sect. 3.4.

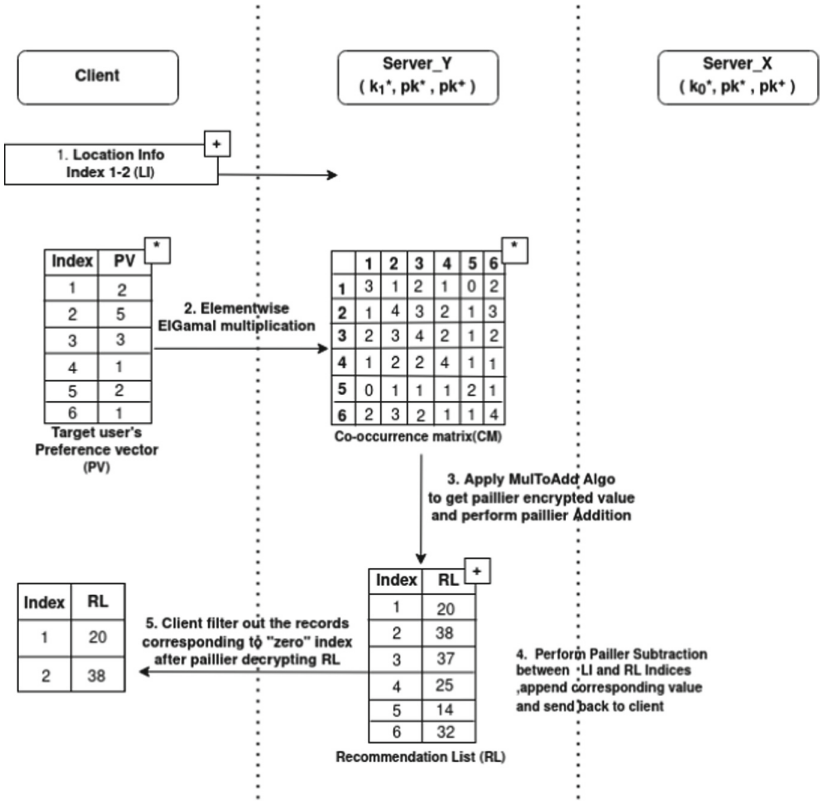

Fig. 5. SHELBRS: proposed recommendation

– The encrypted CM is stored at server Y.

The computation of CM is considered as the initial setup required before client's experiment. So, the computation cost of CM is not considered in the total computation cost taken by the client.

5.3 Protocol Operation Stage

The detailed process of how the recommendation is being generated is shown in Fig. 5. The client and the servers interact in the following manner:

Step 1: The client encrypts history preference vector PV using $Enc(pk^*, PV)$ and location info using $Enc(pk^*, Location\_info)$ and sends it to server Y.
Step 2: The client computes each item's prediction value P using (16) to generate a recommendation list. The prediction is generated using Algorithm 1.

Algorithm 1. Recommendation

INPUT: CM^*, PV^*, Item Index set I^*
OUTPUT: Recommendation list RL for all items
Define: 1. CM^* is ElGamal encrypted co_occurrence matrix
2. PV^* is ElGamal encrypted history preference vector
3. RL[i] is item $i's$ recommendation score

procedure RECOMMEND(CM^* , PV^* , I^*, size)
  Assign $RL^+[1 \cdots size] = 0$
  for $i = 1;\ i <= size;\ i = i + 1$ **do**
    for $j = 1;\ j <= size;\ j = j + 1$ **do**

$$ctotal = \textbf{ElGamalMultiplication}(CM[i][j]^*, PV[j]^*)$$
$$c1 = \textbf{PallierEncryption}(pkadd, ctotal[0])$$
$$temp = \textbf{MulToAdd}([c1, ctotal[1]], k_1^*, k_0^*)$$
$$RL[i]^+ = \textbf{PaillierAddition}(RL[i]^+, temp)$$

    end for
  end for
  return RL^+
end procedure

To calculate each index's recommendation score, ElGamal encrypted CM^* and PV^* are elementwise multiplied using the multiplicative property of **ElGamal Encryption**. The corresponding result is transformed into encrypted ElGamal ciphertext using **Paillier Encryption** scheme in Sect. 3.3.

Step 3: It is further converted into Paillier encrypted ciphertext using **MulToAdd** algorithm in Sect. 3.4. The corresponding result is finally added using additive property of **Paillier Encryption** to generate the corresponding recommendation score.

Likewise, the above steps 2 and 3 are executed for each index to generate the final recommendation list.

Step 4: According to the target user's location, the final recommendation list is filtered out.

The server Y performs Paillier homomorphic subtraction property corresponding to the indices stored in the recommendation list and user's location. It appends the result to the recommendation list and the updated list is sent to the client.

Step 5: The client then decrypts it using Paillier decryption algorithm and filters out the records corresponding to the value '0'. The client chooses one of the locations (indices) from the recommendation list according to his/her choice and update his/her behavior in the inversion list. Each client sends only his/her new paillier encrypted CM_u to server Y which is the difference between the current CM_u and the client's original CM_u before the recommendations.

6 Experimental Results

To validate the proposed protocol based on SHE, the experiment is executed on Ubuntu 20.04.2 LTS powered by Intel® Core™ i5-6200U CPU @ 2.30 GHz × 4 processor and RAM 8 GB. We have considered a client and two servers on the same machine. In this experiment, we considered the artificial dataset with POIs in range {10, 20, 40, 80, 100, 1000}.

The CM is already stored on the server Y and then, the client starts executing his/her behavior. So, the execution time taken for the computations during initialization phase to generate CM is not considered in the total computation cost taken by the client. The time taken to generate public-private keypair is constant. The updation of ED can be performed even when a user is offline, so it does not affect the efficiency of the system.

Table 1. Computation cost

Total elements	Encryption time [s]	Recommendation time [s]	Decryption time [s]
10	0.001	0.022	0.001
20	0.001	0.081	0.001
40	0.001	0.299	0.002
80	0.003	1.142	0.004
100	0.004	1.923	0.006
1000	0.04	197.99	0.058

The total computation cost involves encryption of a client's preference vector, computation of recommendation list and decryption of recommendation list. In Table 1, we measured encryption time, recommendation time,and decryption time for the indices in an encrypted domain. We have run the experiment five times for each index and taken an average of it.

The experiment uses encryption which adds extra computation cost over plaintext. So, we calculated and compared the total computation cost for the plaintext and encrypted domain up to 1000 indices as shown in Table 2.

We also plotted the graph comparing the total execution time taken by SHELBRS scheme and Lyu et al. protocol in Fig. 6 and Table 3.

6.1 Security Analysis

Our protocol aims to provide data confidentiality. It does not leak any meaningful information throughout the protocol. The security of the scheme is based on the following assumptions:

– The ElGamal and Paillier schemes are secure.

Table 2. Comparison of total execution time in plaintext domain and encrypted domain

Total elements	Plaintext domain	Encrypted domain
10	0.001	0.024
20	0.001	0.083
40	0.002	0.302
80	0.005	1.149
100	0.007	1.933
1000	1.32	198.088

Table 3. Comparison of total execution time taken by proposed SHELBRS scheme and Lyu et al. [9]

Total elements	Proposed SHELBRS scheme	Lyu et al. [9]
10	0.024	2.79
20	0.083	5.48
40	0.302	11.13
80	1.149	22.32
100	1.933	28.03
1000	198.088	269.47

- At least one of the servers is honest i.e. if one of the servers is malicious, the other server remains honest.
- None of the servers collude.

Let us assume an Adversary A plays the role of either a malicious server Y or a malicious server X. Initially, A is given the public keys and private key shares to perform Paillier encryption E^+, ElGamal encryption E^*, AddToMul and MulToAdd algorithms. A is also given access to choose any arbitrary plaintext and can perform encryption to get the corresponding ciphertext. A user sends $E^*(m_0)$ to the server Y where m_0 is an integer except a value 0. Now, A's goal is to find a challenge m'_0 such that $m'_0 = m_0$. If A's goal is achieved, the security is broken. To prove $m'_0 s$ security, below mentioned lemmas are as follows.

Lemma 1: If Server Y is a malicious server and A chooses to attack AddToMul algorithm, it has access to $E^*(m_0)$, $E^+(m_0)$ and $E^+(E^*(m_0))$. Paillier encrypted $E^+(m_0)$ and $E^+(E^*(m_0))$ terms are semantically secure and no decryption key sk^+ is associated with any server, so, no information regarding m_0 is leaked through these terms. Server Y has key share $k_1^* = x_1$ and $E^*(m_0) = (m_0 h^r, g^r)$ where r is randomly chosen from the group Z_N^*. The security of $E^*(m_0)$ based

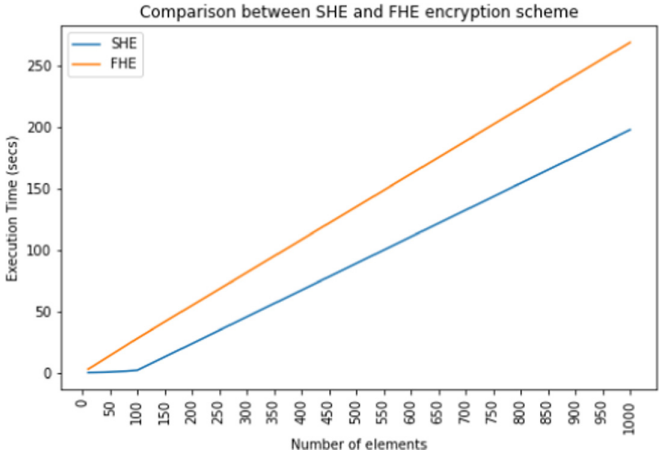

Fig. 6. Comparison of total execution time taken by proposed SHELBRS scheme and Lyu et al. [9]

on SHE security proof [7] is perfectly secure. So, A cannot learn anything about m_0 in the protocol.

Lemma 2: If Server Y is a malicious server and A chooses to attack MulToAdd algorithm, it has access to $k_1^* = x_1$. $E^*(m_0)$, $E^+(E^*(m_0))$, $R' = g^{sx_0}$ and $c'' = E^+(m_0 h^{-s})$. $E^*(m_0)$, $E^+(E^*(m_0))$ and c'' are secure according to Lemma 1. A cannot learn about secret key share x_0 from R' as s is randomly chosen from the group Z_N^*. Therefore, the proposed scheme is secure against the malicious activity performed by server Y itself.

Likewise, we can prove the data confidentiality using Lemma 1 and Lemma 2 when Server X acts as an adversary A.

Now, we will handle the case when the client performs elGamal encryption on a message m_0 where $m_0 = 0$. The ElGamal Encryption of m_0 plaintext results into one of the ciphertexts as "zero". This is not secure as it leaks information regarding plaintext data. To handle such problems, we can represent "zero" in the form

$$\mathbf{MulToAdd}(E^+(E^*(n_1))) * \mathbf{MulToAdd}(E^+(E^*(n_1)))^{-1} = E^+(0) \tag{17}$$

7 Conclusions and Future Work

A lightweight privacy-preserving recommendation protocol for LBS was proposed in this paper. It incorporated Hilbert curve, collaborative filtering recommender based on co-occurrence matrix and SHE to recommend the services. Based on the simulation and experiments, we found that the computation cost for 1000

POIs is 198.088 s. Compared with the state-of-the-art protocol, the proposed protocol takes less computation time and reduces complexity, providing at par security. As the future direction of the work, we would like to extend our protocol for the larger geographical area as we focused herein only on item-based filtering on a single geographical area.

References

1. Badsha, S., Yi, X., Khalil, I.: A practical privacy-preserving recommender system. Data Sci. Eng. **1**(3), 161–177 (2016)
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. ACM Trans. Comput. Theory (TOCT) **6**(3), 1–36 (2014)
3. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing, pp. 169–178 (2009)
4. Gruteser, M., Grunwald, D.: Anonymous usage of location-based services through spatial and temporal cloaking. In: Proceedings of the 1st International Conference on Mobile Systems, Applications and Services, pp. 31–42 (2003)
5. Horowitz, D., Contreras, D., Salamó, M.: EventAware: a mobile recommender system for events. Pattern Recogn. Lett. **105**, 121–134 (2018)
6. Lien, I.-T., Lin, Y.-H., Shieh, J.-R., Wu, J.-L.: A novel privacy preserving location-based service protocol with secret circular shift for k-NN search. IEEE Trans. Inf. Forensics Secur. **8**(6), 863–873 (2013)
7. Lim, H.W., Tople, S., Saxena, P., Chang, E.-C.: Faster secure arithmetic computation using switchable homomorphic encryption. IACR Cryptol. ePrint Arch., 2014:539 (2014)
8. Lü, L., Medo, M., Yeung, C.H., Zhang, Y.-C., Zhang, Z.-K., Zhou, T.: Recommender systems. Phys. Rep. **519**(1), 1–49 (2012)
9. Lyu, Q., Ishimaki, Y., Yamana, H.: Privacy-preserving recommendation for location-based services. In: 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA), pp. 98–105. IEEE (2019)
10. Melchor, C.A., Gaborit, P.: A fast private information retrieval protocol. In: 2008 IEEE International Symposium on Information Theory, pp. 1848–1852. IEEE (2008)
11. Moon, B., Jagadish, H.V., Faloutsos, C., Saltz, J.H.: Analysis of the clustering properties of the Hilbert space-filling curve. IEEE Trans. Knowl. Data Eng. **13**(1), 124–141 (2001)
12. Papakyriakopoulos, O., Serrano, J.C.M., Hegelich, S.: Political communication on social media: a tale of hyperactive users and bias in recommender systems. Online Soc. Netw. Media **15**, 100058 (2020)
13. Pingley, A., Wei, Yu., Zhang, N., Xinwen, F., Zhao, W.: A context-aware scheme for privacy-preserving location-based services. Comput. Netw. **56**(11), 2551–2568 (2012)
14. Qi, L., Wang, R., Chunhua, H., Li, S., He, Q., Xiaolong, X.: Time-aware distributed service recommendation with privacy-preservation. Inf. Sci. **480**, 354–364 (2019)
15. Fenwick, M.I.R., Hittle, M., White, O.: Fitness app strava lights up staff at military bases. BBC Journal Archive (2018). https://www.bbc.com/news/technology-42853072

16. Sagan, H.: Space-Filling Curves. Springer, Heidelberg (2012)
17. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295 (2001)
18. Sun, G., Song, L., Liao, D., Hongfang, Yu., Chang, V.: Towards privacy preservation for "check-in" services in location-based social networks. Inf. Sci. **481**, 616–634 (2019)
19. Utsunomiya, Y., Toyoda, K., Sasase, I.: LPCQP: lightweight private circular query protocol for privacy-preserving k-NN search. In 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), pp. 59–64. IEEE (2015)
20. Zhang, X., Yu, L., Wang, M., Gao, W.: FM-based: algorithm research on rural tourism recommendation combining seasonal and distribution features. Pattern Recogn. Lett. (2018)