Recommendation in the End-to-End Encrypted Domain

Jyh-Ren Shieh
Graduate Institute of Networking and Multimedia
National Taiwan University
Taipei, 10617, Taiwan, ROC
jerry@cmlab.csie.ntu.edu.tw

Ching-Yung Lin
IBM T. J. Watson Research Center
Hawthorne, New York
NY 10532
USA
chingyung@us.ibm.com

Ja-Ling Wu
Graduate Institute of Networking and Multimedia & Department of CS
National Taiwan University
Taipei, 10617, Taiwan, ROC
wjl@cmlab.csie.ntu.edu.tw

ABSTRACT
In recommendation systems, a central host typically requires access to user profiles in order to generate useful recommendations. This access, however, undermines user privacy; the more information is revealed to the host, the more the user's privacy is compromised. In this paper, we propose a novel end-to-end encrypted recommendation mechanism which encrypts sensitive private data at the user end, without ever exposing plaintext private data to the host server. Unlike previously proposed privacy-preserving recommendation mechanisms, the data in this proposed system are lossless – a pivotal feature to many applications, e.g., in health informatics, business analytics, cyber security, etc. We achieve this goal by developing encrypted-domain polynomial ring homomorphism cryptographic algorithms to compute similarity of encrypted scores on the server, so that collaborative recommendations can be computed in the encryption domain and only an authorized person can decrypt the exact results. We also propose a novel key management system to make sure private information retrieval and recommendation computations can be executed in the encrypted domain in practice. Our experiments show that the proposed scheme offers robust security and lossless accurate recommendation, as well as high efficiency. Our preliminary results show the recommendation accuracy is 21% better than the existing statistical lossy privacy-preserving mechanisms based on random perturbation and user profile distribution. This new approach can potentially be applied to various data mining and cloud computing environments and significantly alleviates the privacy concerns of users.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Information Filtering;

General Terms
Algorithms, Design, Security, Experimentation.

Keywords
Recommender Systems, Personalization, Privacy.

1. INTRODUCTION
Users can be overwhelmed by the huge number of choices available on the Internet. Recommendation systems are widely used to narrow down their choices, for items of any type: books, movies, sports, restaurants, tourist destinations, online news, etc. User profiles such as age range, gender, and location, as well as web activity history such as search keywords and click-through sequences all can be used to find dating prospects over the Internet. The widely-used collaborative recommendation systems rely on the similarity of user tastes: if two users have similar preferences for certain items, one user will probably be interested in the items that the other is interested in. Thus, the more information each user gives to the recommender system about his interests, the more meaningful the recommendations will be.

A recommendation system creates user profiles by collecting information about user preferences for different items. The preferences of a user in the past can facilitate the prediction of other matching information that might also be of interest to the user in the future. In order to recommend items to users, a recommendation host must access the user's profiles, which contain various person-identifiable information. These profiles are usually stored in the host repositories. Currently, users of any recommendation system must allow the host access to the complete contents of their profiles and trust (or, in most cases, just hope) that the host company and the server administrators will keep this information private.

Allowing "plaintext" information, which is readable by the server, leads to many thorny privacy issues. First, if a user's real identity is known to the host, the host can associate the user's personal profile, which contains his private information, to his real identity. This is an obvious privacy breach, considering that although using the service, the user does not want the link between his real identity and his profile to be revealed. This threat becomes more obvious when users reveal sensitive personal data, such as age, height, weight, sexual orientation, and religion. Second, even if the real identity of a user is not known to the server, it can attempt to de-anonymize the user's identity by correlating the information contained in the user's profile and information obtained from other related databases. Third, hosts may collect personal information about their users and resell it to various vendors for sales and marketing to gain significant financial benefits. Today, there is much such user data to be had. However, user data is vulnerable to criminals; it can affect one's reputation, business operations, etc. From a privacy perspective, it is obvious that hiding personal information from the host is a good thing. To do so, an end-to-end encrypted recommendation computing system such as that proposed in this paper is probably the best solution. It leaves private information in each individual's possession, while allowing the server to conduct recommendation computations in the encrypted domain without knowing the original plaintext profiles.

CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.

Several techniques have been proposed to preserve user privacy in recommendation systems. The three main categories for privacy preservation in recommendation systems are user score perturbation, cryptographic approaches, and distributed storage of user profiles. Randomized perturbation [1][2] techniques have been used to preserve privacy in collaborative recommendations. In this approach, user data are modified by adding random noise in order to prevent the central server from deriving the actual user rating. However, the added noise prevents the server from being able to estimate the actual values for similarity measurements; moreover, additional security flaws are caused by small user space sizes. Canny [3][4] proposed a community-based cryptographic privacy-preserving collaborative filtering method. A community of users can compute a public "aggregate" of their data without disclosing any individual user's data. This aggregation allows personalized referrals to be computed by members of the community. A partially homomorphic encryption like RSA [5] is employed to allow sums of encrypted vectors to be computed and decrypted without exposing individual data, but this approach still has accuracy problems. Thus both randomized perturbation and cryptography-based aggregate computation entail a compromise between recommendation accuracy and personal data security. The third option is to store user profiles locally and run the recommendation system in a distributed manner by revealing partial information, without relying on a centralized server [8]–[12]. In this case, although servers only know partial information, eventually, an end-to-end service provider may still be able to assemble the information from each individual server which knows part of the information. For instance, dividing my personal information into 10 pieces and telling 10 different people does not really protect my privacy.

Our contributions are as follows. In this paper, we first propose a novel system solution for privacy-preserving recommendations in which we do not require several customers to be online simultaneously, nor do we introduce random perturbations. Second, we conduct encrypted-domain addition and multiplication operations based on polynomial rings, so that we can generate the same collaborative filtering results as that obtained from the plaintext data. Third, we propose a ring homomorphic encryption algorithm which makes the encrypted-domain recommendation calculations highly computationally efficient, a key step toward making encrypted-domain recommendations feasible. As our approach is based on lossless number theory, recommendation results are the same as recommendations made based on plaintext data. This solves the many drawbacks described above, and is the goal of this paper.

The rest of this paper is organized as follows. We review related work in Section 2. In Section 3, we describe in detail our privacy-preserving system, including how to handle the key management of collaborative recommendation and how to compute the user-to-user similarity entirely in the encryption domain. In Section 4 we analyze the security, efficiency, and accuracy of the proposed system, and in Section 5 we conclude and suggest directions for future work.

2. RELATED WORKS

2.1 Randomized Perturbation

In the randomized perturbation scheme, each user disguises his or her personal data and then sends it to a central place (the data collector) such that the data collector cannot derive the user's actual private information. This scheme should still be able to allow the data collector to conduct collaborative filtering using the disguised data. Polat and Du [1][2] proposed a technique in which user ratings are modified by adding random noise to them in order to prevent the central server from deriving the actual user ratings. The challenge is to find a perturbation algorithm that imposes the smallest error on the recommendation process. Users enjoy a high level of privacy if the server is not able to estimate the actual ratings that they assigned to the items. However, the perturbed list of items rated by the users is revealed, regardless of the perturbation level. This overlooks the fact that keeping the connection between users and items is more crucial (in order to preserve users' privacy) than disguising the ratings assigned to those connections. Also, revealing the places visited by the users to the server enables it to track users over space and time, whether they liked those places or not.

2.2 Homomorphic cryptography

Recommendation schemes based on partially homomorphic cryptography as proposed by Canny et al. [3][4] require user participation in the distributed system, which may not reflect real-world usage such as cloud computing. Recently, in cryptography, Gentry et al. proposed a breakthrough fully homomorphic encryption scheme using lattice-based cryptography that supports computation in the encrypted domain [6][7]. To the best of our knowledge, we are not aware of any prior work applying such a fully homomorphic scheme to recommendation applications (or, perhaps, any data mining application). In practice, their scheme may need substantial improvements in order to be used in practical applications, because the ciphertext size and computation time increase sharply as one increases the security level. Their constructions are limited to evaluating low-degree polynomials over encrypted data. Moreover, the scheme allows only single bit operations, and even that is computationally prohibitive. Also, the scheme does not offer a practical key management solution where all of the users share the same public key for a full homomorphic encryption application.

2.3 User Profile Distribution

Another option is to store user profiles locally and to run the recommender system in a distributed manner, without relying on any server. Miller et al. [8] proposed transmitting only the similarity measures over the network and keeping users' profiles secret on their sides to preserve privacy. Berkovsky et al. [9] proposed a distributed P2P system to avoid storing users' profiles on a single server. Although these methods eliminate the main source of threat against users' privacy, they require high cooperation among users to generate meaningful recommendations. Every user pays the price of using this method, regardless of his interest in protecting privacy. Lathia et al. [10] proposed a concordance measure to estimate the similarity between two users in a distributed system without revealing their actual profiles to each other. Lathia's method demands that users trust each other to estimate their similarity is hardly feasible in a recommendation system based on thousands of users [11][12].

2.4 Collaborative Recommendation

The most common form of collaborative recommendation is the neighborhood-based approach, which works by computing the similarity between all pairs of users; predictions are made by aggregating ratings of the target item by the user's neighbors or the user's ratings of items that are neighbors of the target item. Algorithms within these families differ in the definition of similarity and the formulation of neighborhoods. Finding a particular consumer's neighborhood with similar tastes or interests and quantifying the strength of similarity are the most crucial

steps for making a collaborative recommendation. Recently, algorithms based on a weighted user-item bipartite-graph soft clustering [15] [16] have been proposed which yield higher accuracies than classical collaborative filtering approaches. This serves as the kernel of our collaborative recommendation system.

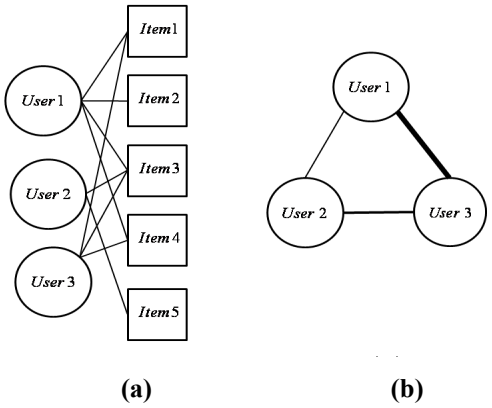

Figure 1. Illustration of soft communities: (a) the bipartite graph representing user-to-item relations, and (b) the induced user cluster affinities.

We summarize the relationship among users, items, and the corresponding item scores, $\mathbb{E}_{\mathrm{ui}}$, using the bipartite graph $\mathbb{G}(\mathbb{U}_{\mathrm{usr}}, \mathbb{I}_{\mathrm{item}}, \mathbb{E}_{\mathrm{ui}})$ shown in Figure 1 (a). In the graph, the nodes at the left denote the groups of n users, the nodes at the right denote the m items, and user groups and item objects are connected by edges associated with corresponding weighting scores. The corresponding user affinities are shown in Figure 1 (b). As pre-scribed, let $\mathbb{G}(\mathbb{U}_{\mathrm{usr}}, \mathbb{I}_{\mathrm{item}}, \mathbb{E}_{\mathrm{ui}})$ denote a bipartite graph, where $\mathbb{U}_{\mathrm{usr}} = \{\mathrm{u}_{\mathrm{usr}}^{i}\}_{i=1}^{n}$ and $\mathbb{I}_{\mathrm{item}} = \{\mathrm{i}_{\mathrm{item}}^{k}\}_{k=1}^{m}$ are two disjoint vertex sets and $\mathbb{E}_{\mathrm{ui}}$ denote all the edges that connect $\mathbb{U}_{\mathrm{usr}}$ and $\mathbb{I}_{\mathrm{item}}$. Let $\mathbb{E}_{\mathrm{ui}} = \{\mathrm{e}_{\mathrm{ui}}^{ik}\}$ denote the $n \times m$ adjacency matrix with $\mathrm{e}_{\mathrm{ui}}^{ik} \geq 0$ being the the corresponding item score between $\mathrm{u}_{\mathrm{usr}}^{i}$ and $\mathrm{i}_{\mathrm{item}}^{i}$. The bipartite graph induces a similarity between $\mathrm{u}_{\mathrm{usr}}^{a}$ and $\mathrm{u}_{\mathrm{usr}}^{j}$, that is,

$$\mathrm{SIM}(\mathrm{u}_{\mathrm{usr}}^{a}, \mathrm{u}_{\mathrm{usr}}^{k}) = \sum_{i=1}^{m} \frac{\mathrm{e}_{\mathrm{ui}}^{ai}\mathrm{e}_{\mathrm{ui}}^{ki}}{\lambda_i} = \left(\mathrm{E}_{\mathrm{UI}}\Lambda^{-1}\mathrm{E}_{\mathrm{UI}}{}^{\mathrm{T}}\right)_{ak}, \quad (1)$$

where $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_m)$, and $\lambda_k = \sum_{i=1}^{m} \mathrm{e}_{\mathrm{ui}}^{ik}$ denotes the degree of the vertex $\mathrm{u}_{\mathrm{usr}}^{i} \in \mathbb{U}_{\mathrm{usr}}$. Equation (1) can be interpreted from the perspective of Markov random walks on graphs. $\mathrm{SIM}(\mathrm{u}_{\mathrm{usr}}^{a}, \mathrm{u}_{\mathrm{usr}}^{k})$ is essentially a quantity proportional to the stationary probability of the direct transition between $\mathrm{u}_{\mathrm{usr}}^{a}$ and $\mathrm{u}_{\mathrm{usr}}^{k}$. However, this shows only the similarity between users. To measure the similarity of the whole group, we have the following matrix formulation:

$$\mathrm{SIM} = \mathrm{W}\Lambda\mathrm{W}^{\mathrm{T}}, \quad (2)$$

where $\mathrm{W} \in \mathrm{Z}_{n\times m}^{+}$ is a non-negative $n \times m$ matrix. For the components of this matrix, one column corresponds to one user and one row to one item, i.e., $\mathrm{w}_{\mathrm{ui}}^{ai}$ is the score of item $\mathrm{i}_{\mathrm{item}}^{i}$ rated by user $\mathrm{u}_{\mathrm{usr}}^{a}$. Λ is an $m \times n$ non-negative diagonal matrix within which λ_k indicates the user's level of participation.

It is well-known that such collaborative filtering can be realized by the calculation of user-to-user similarity so as to recommend items to users on the basis of the past history of the most similar users.

3. PRIVACY-PRESERVING RECOMMENDATION SYSTEM

3.1 Encrypted-Domain Recommendation System

3.1.1 One possible use case

We use the following example to describe our privacy-preserving encrypted-domain recommendation system. In health informatics, patient privacy must be carefully protected. Therefore, after seeing the doctor, the patient encrypts the more sensitive parts of his or her own medical record – for instance, psychological treatments, infertility treatments, or cosmetic procedures – and then transmits them to the server at the hospital or insurance company via a secure channel. Thus sensitive data stored in the central database is totally encrypted. When another doctor diagnoses a new patient, in order to find the most appropriate treatment, he or she looks for similar patients with similar symptoms in the encrypted database to see how other doctors diagnosed them, and then takes this into account to give their own patient the best treatment. Figure 2 illustrates our privacy-preserving recommendation scheme for this particular case.

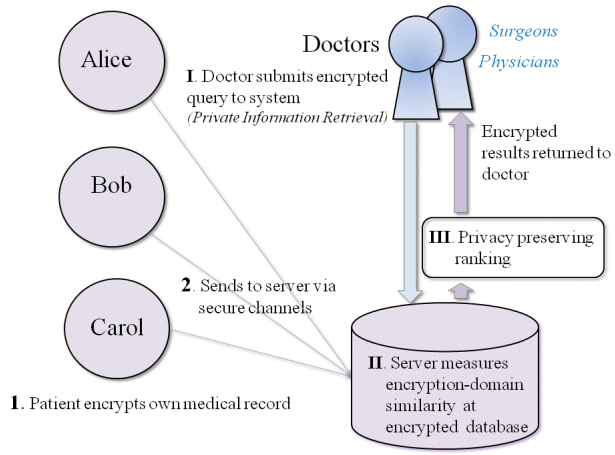

Figure 2. Privacy-preserving recommendation scheme.

3.1.2 System Structure

In order to achieve the above functions, our system requires three encryption mechanisms. The first one is the ring homomorphic encryption. To ensure privacy, the patient should encrypt his or her own data. The goal is that every patient's sensitive medical records are encrypted and that similarity measurements on this data are evaluated in the encryption domain. Thus, ring homomorphic encryption provides the first tier of data protection, allowing for computing multiplication and addition in the encryption domain and inducing these two operations for the underlying plaintext. Homomorphic key distribution is conducted at the key manager server. Thus, when the inspection data is obtained at the clinic, the key management server delivers the homomorphic key to the patients, who encrypt their own private medical information and send it to the hospital database server via a secure channel.

The channel between the parties, such as from the user to the server, may be vulnerable to eavesdropping and malicious attacks, including man-meet-in-the-middle and replay. Also, to ensure the confidentiality of medical records, we must hide the identity of the patients. Thus, another important issue is how to send these medical records in a way that preserves the patient's privacy and anonymity. Ring homomorphic encryption calculation alone cannot ensure information security. Thus, we here apply the second tier of communication security mechanisms using the improved Diffie-Hellman-based protocols, which enable us to securely send the patient's encrypted data to the server. Also, instead of using digital signatures, our scheme hides patient identities by using a hash function. Effective key generation and management is critical for a complete, end-to-end encrypted domain recommendation system. The key for the secure channel by our improved Diffie-Hellman-based protocol is based on the secret sharing principle. Hence the session key is obtained by the two communication parties.

When the doctor submits an encrypted request to the server to calculate the similarity of private information, the similarity measure is calculated in the encryption domain. Therefore, the server needs not be trusted since it knows nothing about the real medical information. Our homomorphic encryption and private information retrieval procedures are illustrated in Figure 3.

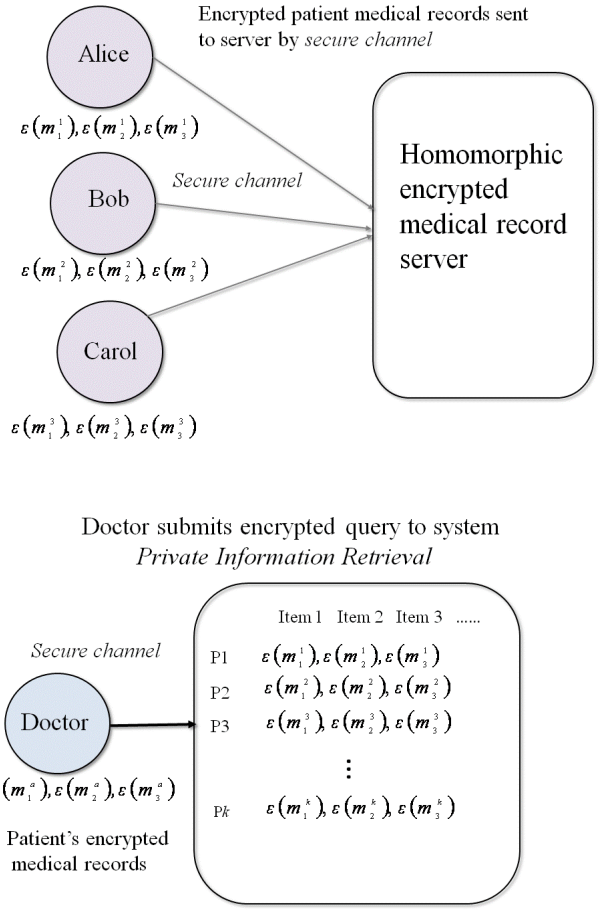

Figure 3. Homomorphic encryption and private information retrieval via end-to-end secure channel.

After the encryption-domain similarity measurement is completed, the doctor provides order-preserving encryption keys to the server database. The third-tier order-preserving encryption is applied to transform these encryption similarity results, which are then sent to the proxy for ranking. Thus, even if the proxy has the homomorphic private key to decrypt the transformed similarity results, it still is limited to calculating rankings, and has no way of knowing the real similarity value. The case ranked most similar to the information is then returned in encrypted form to the doctor, who can use his or her private key to participate the order-preserving data encryption. In this way the doctor takes into account the treatment for similar patients when deciding on the best treatment for the current patient, as shown in Figure 4.

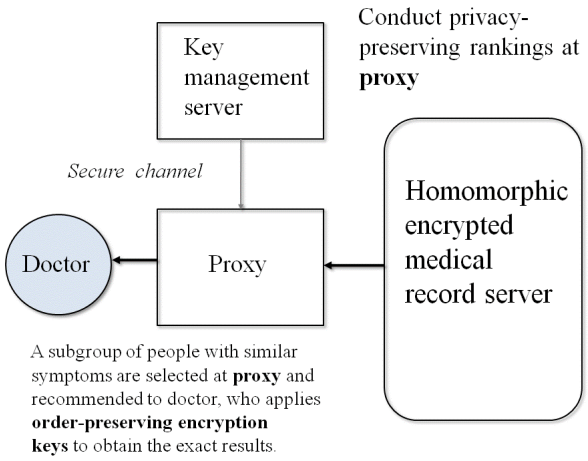

Figure 4. Order preserving encryption and encrypted similarity ranking.

3.2 Ring Homomorphic Encryption

Homomorphic encryption is a form of encryption where a specific algebraic operation performed on the plaintext is equivalent to another (possibly different) algebraic operation performed on the ciphertext. The commonly used RSA code is only homomorphic with respect to multiplication. Given public key (N, k), x is encrypted as $x^k \bmod N$. Since $(x^k \ mod \ N) \cdot (y^k \ mod \ N) = (xy)^k mod \ N$, it follows that the encrypted form of xy is the product of the codes for x and y. Hence we can correctly multiply two encrypted numbers without ever knowing what the numbers were, and obtain a lossless result.

Multiplication-only homomorphic cryptosystems such as RSA are not enough for end-to-end encrypted-domain recommendation. For this task fully homomorphic encryption is required. The utility of fully homomorphic encryption has been long recognized. However, it was not considered feasible until 2009, when Gentry et al. [6] proposed the first fully homomorphic encryption scheme based on lattice-based cryptography. The scheme is limited to evaluating low-degree polynomials over encrypted data, and thus a modified bootstrable scheme was applied to improve its efficiency [7]. One obstacle of these methods is that the key operations are based on one binary bit [6][7].

We propose an efficient ring homomorphic encryption system based on polynomial rings which directly calculates integers in the encrypted domain, and is thus more efficient for tasks such as collaborative filtering based recommendations. A ring homomorphism is a function between two rings which respects the operations of addition and multiplication. With ring homomorphic encryption, you submit an encrypted copy of all

your private data and get back an encrypted copy of the actual scalar matching result you are looking for, without revealing any of your private data. A ring homomorphism function, in our scheme, is a function between two polynomial rings with respect to the operations of addition and multiplication. More precisely, if R and S are polynomial rings, then a ring homomorphism is a function $\phi: R \to S$ such that

$$\phi(x+y) = \phi(x) + \phi(y) \text{ and} \quad (3)$$

$$\phi(x \times y) = \phi(x) \times \phi(y) \quad (4)$$

3.3 Encrypted Domain Operations

We need to derive encrypted-domain calculations, including both multiplications and additions. In (2), we have $\mathrm{SIM} = \mathrm{W}\Lambda\mathrm{W}^{\mathrm{T}}$, where $\mathrm{W} \in \mathrm{Z}^{+}_{\mathrm{n}\times\mathrm{m}}$ is a non-negative matrix and its entry component represents the weighting of the relationship between each user and each item. To preserve user privacy we must define a method for measuring similarity in the encryption domain. Based on equations (1) and (2), we have

$$\mathcal{E}\left(\mathrm{SIM}(\mathrm{u}^{a}_{\mathrm{usr}}, \mathrm{u}^{k}_{\mathrm{usr}})\right) = \sum_{i} \frac{\varepsilon(m^{a}_{i}) \cdot \varepsilon(m^{k}_{i})}{\sum_{k}\sum_{a} \varepsilon(m^{a}_{i}) \cdot \varepsilon(m^{k}_{i})}$$

$$\approx \textstyle\sum_{i} \varepsilon(m^{a}_{i}) \cdot \varepsilon(m^{k}_{i}), \quad (5)$$

where $\varepsilon(.)$ stands for the ring homomorphic encryption algorithms and m^{a}_{i} denotes the weight of the relationship between user a and item i, which is the simplified notation for $\mathrm{w}^{ai}_{\mathrm{ui}}$ used in Equations (2) for describing the graph model.

For all users and items we also have, in matrix form, the soft clustering formulation for a group of people in the encryption domain as follows:

$$\mathcal{E}(\mathrm{SIM}) = \mathcal{E}(\mathrm{W}\Lambda\mathrm{W}^{\mathrm{T}}) = \mathcal{E}(\mathrm{W})\mathcal{E}(\Lambda)\mathcal{E}(\mathrm{W}^{\mathrm{T}}), \quad (6)$$

where Λ is an $m \times n$ non-negative diagonal matrix with λ_k indicating how often a user participates in a given activity. We propose polynomial ring homomorphism, which is described in this section also with details in Section 4, to realize equation (4) in practice. After the server-side encrypted-domain computation, the encrypted recommendation results are returned to each user, again via a secure channel, with which the authorized user can decrypt the results using his or her private key. Thus from the user side to the server, and then back to the user side, all the user's information is encrypted, transferred, and computed in the encrypted domain. The encrypted-domain computation is based on the polynomial ring homomorphism. In this section, we derive a new ring homomorphism based on the concept of lattice-based cryptosystems [14].

Lattice-based cryptographic schemes hold great promise even against quantum computers, as they boast very strong security proofs based on worst-case hardness, relatively efficient implementations, and great simplicity. Compared to modular exponentiation (RSA) as well as repeated squaring and doubling (ECC), lattice-based polynomial ring homomorphic encryption provides not only strong encryption-domain summation and multiplication, but also boasts $O(N^2)$ performance or even $O(NlogN)$ when applying FFT. Note that both RSA and ECC are of complexity $O(N^3)$. This is because polynomial ring convolution operations are faster than the other approaches. Moreover, the cryptosystem uses the polynomial ring $P[N] = \mathrm{Z}[x] / (x^N - 1)$ and $\mathrm{P}_q[N] = \mathrm{Z}_q[N]/(x^N - 1)$, where N is positive prime, Z denotes the ring of integers, and Z_q is the integer modulo q. Therefore the encryption capability associated with the polynomial ring $\mathrm{P}_q[N]$ is Z_q^N. For example, if modulo q is 17 and the degree N is 7, the total representation plaintext number can reach 17^7. Thus the throughput of the proposed encryption scheme is far greater than Gentry et al's single-bit representation with only 2^1 options. This shows that our system lends itself to real-world implementations.

As shown in Figure 5, a lattice is a regular array of points in space: note that we can connect the dots to form parallelograms. The lattice may be described by giving basis vectors that span such a parallelogram. The closest vector problem (CVP) is that given point P, it is difficult to find the point Q in the lattice that is closest to P without knowing L, the basis of the lattice.

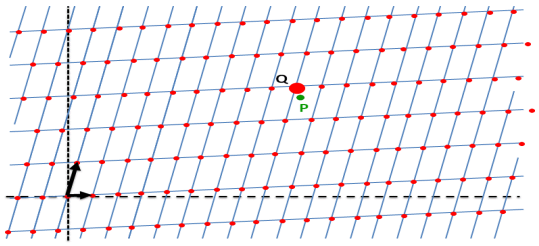

Figure 5. Example of lattice-based cryptography, where Q is the encrypted point of P, and the basis of the lattice

3.4 Lattice-Based Polynomial Ring Homomorphic Encryptions

In this section we describe the encryption and decryption scheme. User-side encryption is conducted as follows.

Score mapping to polynomial rings. Alice transfers the integer score for a item to a polynomial. For example, 3 is mapped to $x+1$, 5 is mapped to x^2+1, and 11 to x^3+x+1. This approach in effect spreads an integer over several coefficients of a polynomial. This idea is the basis for the secrecy, efficiency, and encryption-domain operation of our proposed approaches.

Parameter settings. Three positive integral parameters (N, p, q) are chosen for the key generation which satisfy the following conditions: (1) N is prime; (2) p and q are chosen such that they are relatively prime to each other, i.e., $gcd(N,q) = gcd(p,q) = 1$. The values of (N, p, q) can be known to the public.

Polynomials chosen. There are four sets of polynomials $\mathcal{L}_f$, $\mathcal{L}_g$, $\mathcal{L}_r$, and $\mathcal{L}_m$ respectively defined in the polynomial ring $R[x] \triangleq \mathrm{Z}[x] / (x^N - 1)$. The coefficients of these polynomials are constrained by the choice of parameter p, i.e. all the operations are defined in the polynomial ring $\mathrm{R}_p[x] \triangleq \mathrm{Z}_p[x]/(x^N - 1)$, which is equivalent to the operations defined by taking modulo $(x^N - 1)$ on polynomials and modulo p on polynomial coefficients simultaneously.

Key generation. Homomorphic key generation is a crucial part of an encryption scheme. In our scheme, the key is embedded in the coefficient of the polynomials as follows.

We first define

$$T(d_1, d_2) = \left\{ f(x): \text{the set of trinary polynomials, in which } d_1 \text{ coefficients of } f(x) = 1 \text{ and } d_2 \text{ coefficients of } f(x) = -1 \right\}.$$

Since $f(x)$ is a trinary polynomial, the system chooses a $f(x) \in T(d+1, d)$ as his or her private key which is used to decrypt the

encrypted-domain recommendation result. Alice then randomly selects a polynomial $f(x) \in \mathcal{L}_f$ such that $f(x)$ has an inverse both in modulo p and modulo q, i.e., the inverses exist in both $\mathrm{R}_p[x] = \mathrm{Z}_p[x]/(x^N - 1)$ and $\mathrm{R}_q[x] = \mathrm{Z}_q[x]/(x^N - 1)$.

That is, the system can compute the two inverses of $f(x)$, $F_p(x)$, and $F_q(x)$, respectively in $\mathrm{R}_p[x]$ and $\mathrm{R}_q[x]$ as

$$f(x)F_p(x) \equiv 1 \; mod \; \mathrm{R}_p[x] \quad \text{and} \qquad (7-\text{a})$$

$$f(x)F_q(x) \equiv 1 \; mod \; \mathrm{R}_q[x]. \qquad (7-\text{b})$$

Then Alice randomly selects a trinary polynomial $g(x) \in \mathcal{L}_g$ in which $g(x) \in T(d,d)$ and then computes

$$h(x) = F_q(x)g(x) \; mod \; \mathrm{R}_q[x] \qquad (8)$$

Alice uses the polynomial $h(x)$ as her public key, and $f(x)$ and $F_q(x)$ becomes her private key for decryption [15].

Centered lift coefficients. Alice keeps each coefficient of the plaintext score polynomial $m_i^a(x)$ within the range $\left(-\frac{p}{2n}, \frac{p}{2n}\right]$ to ensure correct encryption and decryption, maintaining the one-to-one and onto property between the plaintext and the cipher text, where n is the number of plaintexts we are going to encrypt and p is the modulo we have chosen. It is well known that if we take integers 2 modulo 7, we can express 2 as $-12, -5, 2, 9, 16$ and so on. Now, we define the centered lift operation (CLO) to restrict the coefficients to be centered around 0. That is, every coefficient $m_i^a(j)$ of $m_i^a(x)$ satisfies $-\frac{p}{2n} < m_i^a(j) \leq \frac{p}{2n}$, where $0 \leq j \leq n$. After applying CLO, the representation for each of the coefficient is unique. As an example, if we choose p=4 and n=1, then $-\frac{4}{2} < m_i^a(j) \leq \frac{4}{2}$. That is, the centered lifted version of the plaintext score polynomial $m_i^a(x) = 3x^5 + 6x^4 + 7x^2 - 3x - 15$ is $CLO_{4,1}\{m_i^a(x)\} = -x^5 + 2x^4 - x^2 + x + 1$.

Encryption. Alice encrypts her score for item i (denoted as $m_i^a(x)$) using the public key $h(x)$ and the randomly blinding key $r_i^a(x)$ as

$$\boldsymbol{e_i^a(x)} = \left(pr_i^a(x)h(x) + m_i^a(x)\right) mod \; \mathrm{R}_q[x] \qquad (9)$$

to make each cipher as unique as if a one-time pad were applied. Likewise, she encrypts her score for item l (denote as $m_l^a(x)$) using the public key $h(x)$ and the blinding key $r_l^a(x)$ as

$$e_l^a(x) = \left(pr_l^a(x)h(x) + m_l^a(x)\right) mod \; \mathrm{R}_q[x]. \qquad (10)$$

As each group of attendees encrypts their own scores for each item using different randomly chosen blinding keys $r(x) \in \mathcal{L}_r$ and their corresponding public keys $h(x)$, in general, the encryption of participant k's score polynomial $m_i^k(x)$ to item i is

$$\boldsymbol{e_i^k(x)} = (pr_i^k(x)h(x) + m_i^k(x)) mod \; \mathrm{R}_q[x] \qquad (11)$$

When the server receives the encrypted scores $\mathrm{e}_i^a(x)$ and $\mathrm{e}_i^k(x)$, it can only multiply the encrypted scores, yielding

$$\mathrm{e}_i^a(x)\mathrm{e}_i^k(x) \; mod \; \mathrm{R}_q[x] =$$

$$\left(\left(pr_i^a(x)h(x) + m_i^a(x)\right)\left(pr_i^k(x)h(x) + \mathrm{m}_i^k(x)\right)\right) mod \; \mathrm{R}_q[x] =$$

$$(p^2 r_i^a(x)r_i^k(x)h(x)^2 + ph(x)\left(r_i^a(x)m_i^k(x) + r_i^k(x)m_i^a(x)\right) +$$

$$\boldsymbol{m_i^a(x)m_i^k(x)}) \; mod \; \mathrm{R}_q[x]. \qquad (12)$$

As we can see in the formula, although the encrypted $\mathrm{m}_i^a(x)$ and $\mathrm{m}_i^k(x)$ corresponding to $\mathrm{e}_i^a(x)$ and $\mathrm{e}_i^k(x)$ are multiplied, the plaintext score multiplication $\boldsymbol{m_i^a(x)m_i^k(x)}$ still remains a separate term in Equation (12). Therefore, although the multiplication of $\mathrm{e}_i^a(x)$ and $\mathrm{e}_i^k(x)$ is performed in the encryption domain, it still induces multiplication of the underlying plaintext $\boldsymbol{m_i^a(x)m_i^k(x)}$, which is the exact, lossless multiplication score of the plaintext, to which is no perturbed noise or distribution. The homomorphic private key owner decrypts the $e_i^a(x)e_i^k(x)$ at the owner side. When the owner receives the encrypted product $e_i^a(x)e_i^k(x)$ from the server, he or she applies the first private key $f(x)^2$ and proceeds as follows:

$$f(x)^2\mathrm{e}_i^a(x)\mathrm{e}_i^k(x) \; mod \; \mathrm{R}_q[x] = f(x)^2\left(p^2 r_i^a(x)r_i^k(x)\boldsymbol{h(x)^2} + ph(x)(r_i^a(x)m_i^k(x) + r_i^k(x)m_i^a(\mathrm{x})) + m_i^a(x)m_i^k(x)\right) mod \; \mathrm{R}_q[x] \text{it which is addedwere} \qquad (13)$$

Since $h(x) = F_q(x)g(x) \; mod \; \mathrm{R}_q[x]$, we have

$$h(x)^2 = F_q(x)^2 g(x)^2 \; mod \; \mathrm{R}_q[x] \qquad (14)$$

Since $f(x)^2 F_q(x)^2 \equiv 1 \; mod \; \mathrm{R}_q[x]$, $f(x)^2$ deduces $\boldsymbol{F_q(x)^2}$ but keeps $\boldsymbol{g(x)^2}$ intact.

After the private key owner applies $f(x)^2$ over $\mathrm{R}_q[x]$, we have

$$\alpha \equiv f(x)^2\mathrm{e}_i^a(x)\mathrm{e}_i^k(x) \; mod \; \mathrm{R}_q[x] = (p^2 r_i^a(x)r_i^k(x) \; \boldsymbol{g(x)^2} + pf(x)g(x)\left(r_i^a(x)m_i^k(x) + r_i^k(x)m_i^a(x)\right) + f(x)^2 m_i^a(x)m_i^k(x)) \; mod \; \mathrm{R}_q[x]. \qquad (15)$$

Correct encryption and decryption needs to maintain the isomorphism, i.e., one-to-one and onto, between plaintexts and ciphertexts, where the coefficients of α should fall within the interval $-\frac{q}{2}$ to $\frac{q}{2}$. The first criterion is that the range of modulo q is bounded from modulo p, and the second is the number of coefficients of $f(x)$ equals to 1 and -1, respectively. If we select appropriate values for multiplication modulo p as well as d, the number of the coefficients equals 1 in the trinary polynomial $f(x)$, $g(x)$ and the homomorphic private key, and then applies the second private key $F_p(x)^2$ and takes operations modulo p as

$$(F_p(x)^2(\boldsymbol{p^2 r_i^a(x)r_i^k(x)} \; \boldsymbol{g(x)^2} + \boldsymbol{pf(x)g(x)}\left(\boldsymbol{r_i^a(x)m_i^k(x)} + \boldsymbol{r_i^k(x)m_i^a(x)}\right) + (\boldsymbol{F_p(x)^2 f(x)^2} m_i^a(x)m_i^k(x)) \; mod \; \mathrm{R}_q[x](\mathrm{mod}\; p).$$

As both terms $(\boldsymbol{p^2 r_i^a(x)r_i^k(x)} \; \boldsymbol{g(x)^2})$ and $\boldsymbol{pf(x)g(x)}\left(\boldsymbol{r_i^a(x)m_i^k(x)} + \boldsymbol{r_i^k(x)m_i^a(x)}\right)$ are divisible by p, these two terms are zeroed out. The remaining term becomes $\boldsymbol{m_i^a(x)m_i^k(x)}$ since $\boldsymbol{f(x)^2 F_q(x)^2 \equiv 1}$. We show that $\mathrm{e}_i^a(x)\mathrm{e}_i^k(x)$ can be decrypted at the user side to get the real weight $\boldsymbol{m_i^a(x)m_i^k(x)}$. Let us use the encryption-domain scalar product $\boldsymbol{m_1^a m_1^k + m_2^a m_2^k + m_3^a m_3^k = 3 \times 7 + 5 \times 6 + 10 \times 1 = 61}$ as an example.

Taking N= 7, $p = 29$, and $q = 491531$,

$f(x) = x^5 - x^4 - x^2 + x + 1$, $g(x) = -x^6 + x^3 + x^2 - 1$, the inverse $F_p(x)$ of $f(x)$ modulo p is $F_p(x) = 7x^6 + 6x^5 + 22x^4 + 10x^3 + 24x^2 + 4x + 15$ and the inverse $F_q(x)$ of $f(x)$ modulo q is $F_q(x) = 359995x^6 + 214612x^5 + 477685x^4 + 48461x^3 + 76153x^2 + 470762x + 318457$.

The public key $h(x)$ is $h(x) = F_q(x)g(x) \; mod \; \mathrm{R}_q[x] = 339225 + 41538x^5 + 346149x^4 + 263073x^3 + 62307x^2 + 27692x + 394609$.

Plaintext scores given from* Users *A and K

$m_1^a = \mathbf{3}$, $m_1^a(x) = x + 1$, $\qquad m_1^k = \mathbf{7}$, $m_1^k(x) = x^2 + x + 1$

$m_2^a = \mathbf{5},\ m_2^a(x) = x^2 + 1,$ $m_2^k = \mathbf{6},\ m_2^k(x) = x^2 + x$

$m_3^a = \mathbf{10},\ m_3^k(x) = x^3 + x,$ $m_3^k = \mathbf{1},\ m_3^k(x) = 1$

The random blinding keys for the encryption of each score are as follows:

$r_1^a(x) = x^3 + x^2 - x - 1,\quad r_1^k(x) = -x^4 - x^2 + x + 1$

$r_2^a(x) = -x^6 + x^3 - x + 1,\ r_2^k(x) = x^6 + x^5 - x^3 - 1$

$r_3^a(x) = x^6 + x^5 - x^4 - x,\ r_3^k(x) = x^5 - x^4 - x^3 + 1$

Encryption

$$\mathrm{e}_1^a(x) = 235408x^6 + 159221x^5 + 179994x^4 + 353054x^3 + 484568x^2 + 269992x + 283889$$

$$\mathrm{e}_1^k(x) = 179965x^6 + 353083x^5 + 484626x^4 + 270020x^3 + 283860x^2 + 235380x + 159193$$

$$e_2^a(x) = 394600x^6 + 339244x^5 + 41546x^4 + 346120x^3 + 263058x^2 + 62319x + 27708$$

$$e_2^k(x) = 186928x^6 + 83034x^5 + 200738x^4 + 34641x^3 + 124639x^2 + 55386x + 297698$$

$$e_3^a(x) = 387666x^6 + 117646x^5 + 325376x^4 + 90027x^3 + 422307x^2 + 242314x + 380790$$

$$e_3^k(x) = 221511x^6 + 207672x^5 + 256152x^4 + 332339x^3 + 311566x^2 + 138448x + 6906$$

Similarity measurement in the encryption domain

$$(e_1^a(x)e_1^k(x) + e_2^a(x)e_2^k(x) + e_3^a(x)e_3^a(x))\ mod\ R_q[x] = 464241x^6 + 345580x^5 + 79051x^4 + 9314x^3 + 130865x^2 + 131419x + 314135$$

Decryption

$$\alpha \equiv (\boldsymbol{f}(\boldsymbol{x})^2(e_1^a(x)e_1^k(x) + e_2^a(x)e_2^k(x) + e_3^a(x)e_3^a(x)))mod\ R_q[x] = 468645x^6 + 475249x^5 + 1531x^4 + 18383x^3 + 23015x^2 + 8623x + 479159$$

$Centered\ lift\ coefficients$

$$\alpha = -22886x^6 - 16282x^5 + 1531x^4 + 18383x^3 + 23015x^2 + 8623x - 12372$$

$$\boldsymbol{F_p(x)}^2((-22886x^6 - 16282x^5 + 1531x^4 + 18383x^3 + 23015x^2 + 8623x - 12372)\ mod\ R_q[x])\ \boldsymbol{mod\ p} = x^4 + 3x^3 + 3x^2 + 4x + 1$$

We then recover the **exact** plaintext similarity measurement results.

$$\boldsymbol{x^4 + 3x^3 + 3x^2 + 4x + 1 = 1 \times 2^4 + 3 \times 2^3 + 3 \times 2^2 + 4 \times 2^1 + 1 = 61}$$

In the general case $\sum_{i=1}^{n}(\mathrm{e}_i^a(x)\mathrm{e}_i^k(x))\ mod\ \mathrm{R}_q[x]$, the criteria of $\frac{q}{2n} > Q_n$, $Q_n = 8p^2d^3 + \frac{4}{n}p^2d^2(2d+1) + \frac{p}{2n}(2d+1)^2$ must be held.

We have thus demonstrated the complete procedure of ring homomorphic encryption-domain similarity measurement. Note that in this paper, our focus is to fundamentally solve the system, key management, and algorithmic challenges of recommendations in the end-to-end encrypted domain. Therefore, we omit the detailed cryptographic proofs of the ring polynomial homomorphism, which is outside the scope of this paper.

3.5 Order-Preserving Encryption Ranking

3.5.1 Concept and scheme

We here present an order-preserving encryption scheme for data which allows any comparison operation to be directly applied on the encrypted data. Figure 4 shows the use of this order-preserving encryption ranking proxy in the overall system. In the original proxyless scenario, when the encryption-domain computation is completed, the server sends back the encrypted weights to the proxy, which decrypts the results using a homomorphic private key. In the original scenario, only the doctor can decrypt the results. Therefore, in order to identify the most similar patient, the server must conduct pairwise encrypted-domain similarity measures between the patient and all other people in the database and then return all the encrypted measurements to the doctor. We derive an order-preserving encryption ranking that makes use of a proxy. As this result is protected by the doctor with order-preserving encryption, the proxy does not know the actual similarity measurement results, as this would allow access to private patient data. Thus the proxy is allowed only to see rankings and not the similarity measurements. This is an important aspect, because the proxy needs to know the rankings of the patients and thus only return the most relevant ones to the doctor. Without this order-preserving encryption ranking procedure, the server would be required to return to the doctor all pairwise similarity measurements, which are not known by the server.

It sends the ranking results to the doctor via a secure channel, after which the doctor uses his or her order-preserving private key to decipher the real measurement results. We have thus shown that a scalar product-based algorithm can be implemented in the encryption domain, which in turn shows that a privacy-preserving collaborative recommendation system can be realized accordingly.

3.5.2 Implementation

Order-preserving encryption ranking is accomplished as follows: after the server implemented the encryption-domain similarity measurement stated in Section 3.4, if it were to directly carry out ranking, the proxy might be able to obtain the actual similarity measure values, which would be a breach of the patient's privacy. Thus we apply order-preserving encryption at the server to rescale the actual similarity measurement value with the order-preserving encryption secret key provided by the doctor. The encrypted similarity measurement is then

$$\mathrm{e}\left(\mathrm{W}_{op}^{ak}\right) = \mathrm{e}\left(pk_{op}^1\right).\mathrm{e}(\mathrm{W}^{ak}) + \mathrm{e}\left(pk_{op}^2\right), \tag{17}$$

where $\mathrm{e}\left(pk_{op}^1\right)$ and $\mathrm{e}\left(pk_{op}^2\right)$ are the order-preserving encryption keys provided by the doctor. $\mathrm{e}(\mathrm{W}^{ak})$ is the original encryption-domain similarity measurement results, and $e\left(\mathrm{W}_{op}^{ak}\right)$ is the similarity measurement results after applying order-preserving encryption. This idea is derived from the affine transform function

$$y = \mathrm{A}x + b. \tag{18}$$

In the finite-dimensional case, each affine transformation is given by a matrix A and a vector b. This transformation satisfies the following two properties: (1) the collinearity relation between points; i.e., points which lie on a line continue to be collinear after the transformation. (2) The ratios of distances along a line; i.e., for distinct collinear points p_1, p_2, and p_3 the ratio $|p_2 - p_1|/|p_3 - p_2|$ is preserved. With this property we can encrypt the real similarity result but still preserve the order of the underlying similarity scores for comparison. After proxy ranking, the group of patients the doctor is looking for is sent to the doctor via a secure channel. The doctor then decrypts this data to gain access to the reference information that he or she needs.

3.6 Secure Channel between Parties

3.6.1 Secure channel

The spirit of our end-to-end encryption domain recommendation is that after getting the information from the patient in the clinic, e.g., the client, the information is encrypted and sent to the server to measure similarity in the encryption domain and then sent to the proxy for ranking, also in the encryption domain. The most

correlated patient's clinical information is sent to the doctor so as to assist the doctor in deciding on an effective treatment. Secure channels are required for every connections involved, e.g., client to server, server to proxy, and proxy to doctor.

3.6.2 Diffie-Hellman-based Protocols

The Diffie–Hellman key exchange protocols allow two parties that have no prior knowledge of each other to establish a shared secret key jointly over an insecure communications channel to provide end-to-end security. This key can then be used to encrypt subsequent communications using a symmetric key cipher. As shown in figure 6, we adopted Diffie-Hellman-based protocols because it offers the following advantages: first, it provides anonymity: our proposed protocols use hash functions instead of digital signatures to confirm identities. In our scheme, a user does not need to sign for his or her private data as this would compromise the user's identity. Second, due to the nature of this protocol, our scheme is protected from man-meet-in-the-middle and replay attacks. Moreover, our scheme is effective and simple, since we adopt a uniform data format and because the hash function is more efficient than the use of digital signatures.

3.6.3 Protocol implementation

In the Diffie-Hellman key protocol, Alice and Bob agree on a public g that generates multiplicative group G. They then select random values r_a and r_b, respectably, in the range between 1 and the order of G, and Alice calculates $t_a = g^{r_a}$ and Bob calculates $t_b = g^{r_b}$, after which they exchange these values. Due to the group homomorphic property, the shared secret key S_{AB} can be generated as $S_{AB} = {t_b}^{r_a} = {t_a}^{r_b}$. In thus doing, Alice and Bob agree on, but do not confirm, the key. We use the hash function $H(.)$ and message authentication code with key k denoted as $MAC_k(.)$ with challenge with response nounces N_A and N_B encrypted using public keys $E_A(.)$ and $E_B(.)$ to prevent compromising the key agreement information.

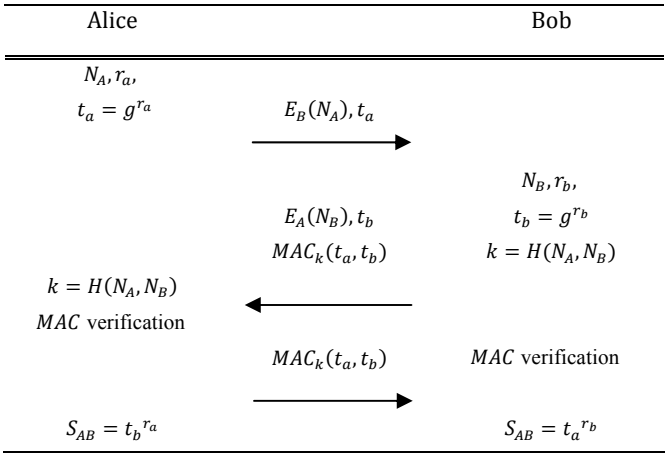

Figure 6. The protocol uses hashes and the MAC function to provide anonymity and identify confirmation.

4. EXPERIMENTS

4.1 Experiment Setting

To demonstrate end-to-end encrypted domain recommendation, we implemented the proposed algorithms as a prototype system using Microsoft Windows 7 (32-bit) operation system, an Intel Core 2 Quad CPU, and 2 GB RAM. We also used this system to evaluate our system security, recommendation accuracy, and system performance.

4.2 Data Sets

In our experiments, we use the Jester and Book-Crossing datasets to evaluate the accuracy of the encrypted-domain recommender. We used the recommendation algorithm described in Section 2, and then compared the results with those from the proposed privacy-preserving collaborative recommendation scheme. We also compared our approaches with randomized perturbation [2] based on the same traditional recommendation method. The Jester dataset is a web-based joke recommendation system, developed at the University of California, Berkeley, collected between April 1999 and May 2003, this database has 4.1 million continuous ratings (from +10 to −10) of 100 jokes from 73,421 users. The Book-Crossing dataset has been mined by Cai-Nicolas Ziegler, DBIS Freiburg, and includes 1,149,780 integer ratings (from 0 to 10) of 271,379 books from 278,858 users.

4.3 Efficiency Comparison

Compared with the standard recommendation system, our system requires a longer running time. This is because we execute end-to-end encryption to secure user information and calculate similarities and ranks in the encryption domain. This is reflected in figure 7, which illustrates the roles that more users and stronger encryption play in increasing runtimes. However, as shown in Section 4, in our privacy information retrieval system, encryption can be done in advance; decryption is a process separate from encryption. Hence in the figure we isolate the running times for the encryption-domain and non-privacy-preserving similarity measurements. These indicate that the difference is limited.

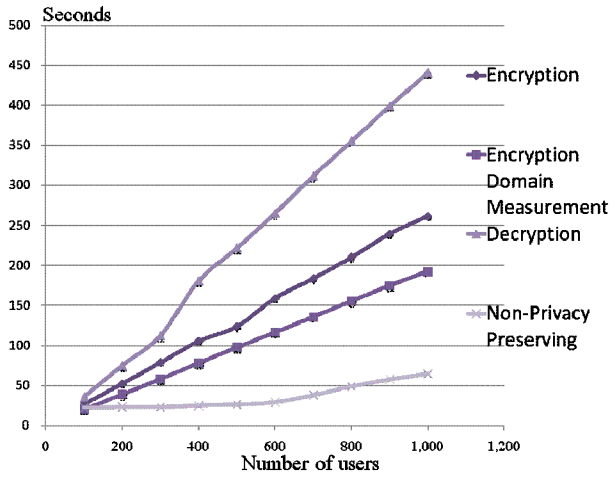

Figure 7. Efficiency comparison between a standard non-privacy preserving recommendation system and the proposed system.

4.4 Accuracy Comparison

Since our approaches are based on number theory with ring homomorphic encryption, our lossless privacy-preserving recommendation scheme provides exactly the same results as the standard non-privacy-preserving recommendation. This is in notable contrast to statistics-based approaches to privacy-preserving recommendation: such approaches yield only approximate results.

4.4.1 Evaluation criteria

We evaluate the system accuracy using the most common criterion: mean absolute error (MAE). This quantity is used to

measure how close forecasts or predictions are to the actual outcomes. The mean absolute error is given by

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n} e_i. \quad (19)$$

We compare MAE as the accuracy for our privacy-preserving approach with that for the non-privacy preserving approach.

4.4.2 Evaluation results

We also compare the results of the random perturbation and user profile distribution approaches. In terms of accuracy, that is, MAE, we compare the results of clustering on the original data with the clustering results for the privacy-preserving approach. Since our approaches are based on ring homomorphism number theory, we obtain exact the same operation values as the unencrypted values, in contrast to the random perturbation and user profile distribution approaches, both of which modify the statististical characteristics of the original values to protect the user's privacy. As shown in Figure 8, in contrast to our approaches, these two methods do not result in higher accuracy.

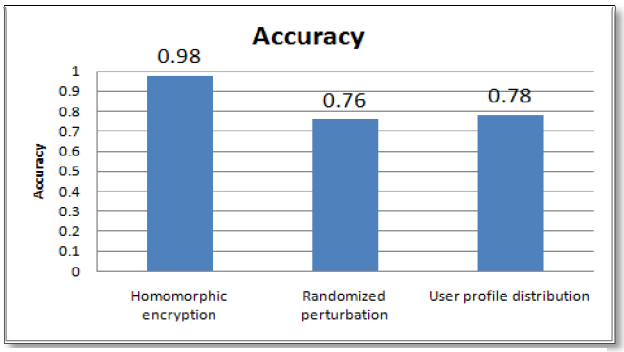

Figure 8. Comparison with the average accuracy of other approaches of privacy-preserving recommendation.

4.4.3 Accuracy effective by user data

In prior work on statistics-based privacy-preserving recommendation systems [2], if there is little user data, or if the data is randomly distributed, recommendation results suffer dramatically. For our system, since we calculate the Euclidean metric in the encryption domain, we have no such limitation.

4.5 Security Comparisons

4.5.1 End-to-end encryption principle for private information retrieval

One serious problem with other privacy-preserving recommendation systems is that users do not encrypt their own data but instead must trust other potentially un-trusted parties to encrypt the data for them. Thus, besides lacking encryption during some parts of the procedure, user privacy is left in the hands of a third party. In comparison, our system provides a level of security that other systems do not provide. Since we implement an end-to-end encryption security environment, it creates a security domain amongst the user's terminal, the service provider, and the database. In Table 1, we compare the recommendation schemes for the other two kinds of privacy-preserving systems.

4.5.2 Encryption algorithm security

As one of post-quantum cryptosystems, our proposed lattice-based encrypted algorithm is stronger than any other lightweight based encryption [1][2][3]. The measurement of the hardness of the system is similar to that of a public key cryptosystem which is usually judged by the most effective known attacks against it. The corresponding most effective known attack is the meet-in-the-middle and the lattice reduction algorithms. When compared to our approach, the randomized perturbation privacy-preserving scheme has the following security problems. First, the privacy is not sufficiently protected, because in order to obtain more accurate recommendation results their systems must fit the perturbation random numbers to a Gaussian distribution with zero mean. This limits their encryption strength. Whereas big plaintext scores and relatively small amounts of random noise render such privacy preservation useless, larger amplitude noise vectors to mask the original data lower prediction accuracy. The second problem is that the system is vulnerable to chosen plaintext attacks. A malicious user who submits scores of zero may be able to compromise the system's random perturbation number. The third security concern is that this approach requires the revealing of extra information to maintain the accuracy of the random perturbation scalar product.

For privacy preservation via factor analysis for user profile distribution, which involves a secure computation technique using partially homomorphic encryptions, user scores are stored as encrypted vectors and aggregates of the data are provided in the public domain. This approach requires users to seek out recommendations explicitly. The partially homomorphic encryptions of this approach, such as RSA and ECC, allow for homomorphic computation of only one operation (either addition or multiplication) on plaintext. Therefore, the similarity measurements it yields are not as accurate as those yielded with our ring homomorphic algorithm, which supports both addition and multiplication in the encryption domain. We compare the security of various privacy-preserving recommendation approaches in table 1.

Table 1. Comparison with the security other approaches

	Our approach	Randomized perturbation	User profile distribution
Vulnerable to attack	No	Yes	Yes
Trusted third party	No	Yes	Yes
Accuracy	High	Medium	Medium
Security	High	Fair	Fair
End-to-end encryption	Yes	No	No

4.4 Accuracy Comparison

Since our approaches are based on number theory with ring homomorphic encryption, our lossless privacy-preserving recommendation scheme provides exactly the same results as the standard non-privacy-preserving recommendation. This is in notable contrast to statistics-based approaches to privacy-preserving recommendation: such approaches yield only approximate results.

4.4.1 Evaluation criteria

We evaluate the system accuracy using the most common criterion: mean absolute error (MAE). This quantity is used to $d_2 \approx N/3$. This will yield stronger security.

4.6 Computation Cost and Complexity

4.6.1 Running time

Experiments indicate that our decryption process requires more computation time, due to our squaring of the private key. However, as our system is based on polynomial rings, the main speed improvements can be made by optimizing polynomial ring operations. The most time-consuming operation is ring inversion. As shown in the experiments, the key part of arithmetic is the ring multiplication operation. Our implementation uses a naive multiplication algorithm. The encryption and decryption performance of our implementation could be improved by implementing a better multiplication algorithm and optimizing it. In a newer version of our system, we pre-calculate the square operation and use a table lookup to retrieve the private key to reduce decryption key generation time.

4.6.2 Real system implementation

We ran a series of tests and extrapolated the data in a conservative manner. A system that works well in practice must get a lot of details right, including security and scalability, respect to the operations on plaintext data. Since the criteria and parameters are more complicated than those for the standard NTRU systems [17], the performance of our system differs from theirs. To make our message space the same as that of the standard NTRU, our system would have to use a larger p and q, which would require more computation time. However, since the operation is done on $R[x] = \mathrm{Z}[x]/(x^N - 1)$, which is limited to evaluating low-degree polynomials over indecipherable encrypted data. Our simulation using the data in Section 4.2 shows each recommendation takes, in average, 1.7 min for 200 users, which is fast enough for some applications e.g. the healthcare informatics scenario mentioned in Section 3.1.

5. CONCLUSION

To the best of our knowledge, we are the first team to propose an end-to-end encrypted domain recommender system which fully protects privacy without the drawback of adding noise or partially revealing information. Our key management protocol and the ring homomorphism mechanism are the key to make a real recommendation work. Our solution makes it possible for servers to collect private data from users without compromising privacy. Our experiments have shown that our solution is accurate, effective, and highly secure. We see a potential advantage in using such an approach for various privacy-preserving mining tasks in people's daily activities. In contrast to the randomized perturbation scheme which adds randomized perturbation to real data, thus yielding only approximate results, our system provides exact measurements. Instead of storing user profiles in a distributed manner, our approach meets the criteria of a centralized scheme, which would work well for cloud computing. Moreover, we provide an effective cryptosystem with ring homomorphism which we have proven not only secure but also practical and implementable as compared to ideal lattice-based approaches, which only allow one bit operation. In the future, we will study how these kinds of operations can be extended to techniques such as singular value decomposition in the encryption domain for other privacy-preserving data mining tasks.

6. REFERENCES

[1] Polat, H. and Du, W. 2006. Privacy-Preserving Collaborative Filtering on Vertically Partitioned Data. PAKDD 2005: 651-658.

[2] Polat, H. and Du, W. 2006. Achieving private recommendations using randomized response techniques. PAKDD 2006. LNCS (LNAI), vol. 3918, 9-12. Springer, Heidelberg.

[3] Canny, J. 2002. Collaborative filtering with privacy. In *proceedings of the IEEE Symposium on Security and Privacy* (Oakland, CA, SA, May, 2002) 45-57.

[4] Canny, J. 2002. Collaborative filtering with privacy via factor analysis. In *proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Tampere, Finland, August, 2002) 238-245.

[5] Rivest, R. L., Shamir, A., and Adleman, L. 1978. A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM 21, 120-126.

[6] Gentry, C. 2009. Fully Homomorphic Encryption Using Ideal Lattices. ACM 41, 169-178.

[7] van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. 2010. Fully homomorphic encryption over integer. In *proceedings of* 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques.

[8] Miller, B. N., Konstan, J. A., and Riedl, J. 2004. Pocketlens: Toward a personal recommender system. *ACM Trans. Inf. Syst.* 22, 3 (2004).

[9] Berkovsky, S., Eytani, Y., Kuflik, T., and Ricci, F. 2007. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *proceedings of ACM RecSys* (2007).

[10] Lathia, N., Hailes, S., and Capra, L. 2007. Private distributed collaborative filtering using estimated concordance measures. In *proceedings of ACM RecSys* (2007).

[11] Miller, B. N., Konstan, J. A., and Riedl, J. 2004. Pocketlens: Toward a personal recommender system. *ACM Trans. Inf. Syst.* 22, 3 (2004).

[12] Luo, B. and Hancock, E. R. 2001. Structural graph matching using the em algorithm and singular value decomposition. *IEEE Trans. Pattern Anal. Mach.* (2001).

[13] NIST, 2006. Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography *Special Publication 800-56A* (2006).

[14] Regev, O. 2006. Lattice-based cryptography. In *Advances in cryptology* (CRYPTO), 131-141, (2006).

[15] Yu, K., Yu, S., and Tresp, V. 2005. Soft Clustering on Graphs. In *Advances in Neural Information Processing Systems*.

[16] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *proceedings of the 10th International World Wide Web Conference* (2001).

[17] Hoffstein, J., Pipher, J., and Silverman, J. H. 1998. NTRU, A Ring-Based Public Key Cryptosystem (ANTS III), LNCS 1423, Springer-Verlag, Berlin, 267-288.