Technical Correspondence

Privacy-Preserving Collaborative Recommender Systems

Justin Zhan, Chia-Lung Hsieh, I-Cheng Wang, Tsan-Sheng Hsu*, Senior Member, IEEE*, Churn-Jung Liau, and Da-Wei Wang*, Member, IEEE*

***Abstract*—Collaborative recommender systems use various types of information to help customers find products of personalized interest. To increase the usefulness of collaborative recommender systems in certain circumstances, it could be desirable to merge recommender system databases between companies, thus expanding the data pool. This can lead to privacy disclosure hazards during the merging process. This paper addresses how to avoid privacy disclosure in collaborative recommender systems by comparing with major cryptology approaches and constructing a more efficient privacy-preserving collaborative recommender system based on the scalar product protocol.**

***Index Terms*—Privacy, security, recommender system.**

I. Introduction

An information filtering system is a system that removes redundant or unwanted information from an information stream using (semi)automated or computerized methods prior to presenting it to a human user. Its main goal is the management of the information overload and increment of the semantic SNR ratio. Recommender systems are active information filtering systems that attempt to present to the user information items (movies, music, books, news, and Web pages) that the user is interested in. Recommender systems [8], [17] typically use collaborative filtering approaches or a combination of the collaborative filtering and content-based filtering approaches, although content-based recommender systems do exist. A recommender system [15] is a Web-based application best known for its usage on e-commerce Web sites, with the aim of helping customers in the decision making and product selection process by providing a list of recommended items. It can be served as one of future directions for cloud computing. The most prominent example is the online bookstore Amazon.com, where collaborative filtering techniques are used to find similarities in users' profiles based on their navigation and buying history. The goal is to identify users who presumably have similar preferences and recommend items that were bought by these related users. Another technical approach is content-based filtering [13], which builds on the hypothesis that the preferred items of a single user can be extrapolated from their preferences in the past. Content-based recommendation systems are systems that recommend an item to a user based upon a description of the item and a profile of the users interests. Content-based recommendation systems may be used in a variety of domains ranging from recommending Web pages, news articles, restaurants, television programs, and items for sale. The third approach is to use domain knowledge to base the recommendations on a thorough understanding of the user's current needs [5], comparable to real-life sales situations. The recommendations are the result of a reasoning process on domain knowledge that also forms the basis for explaining to the user why an item is proposed. Knowledge-based recommender systems explicitly elicit user preferences, i.e., they provide dynamic personalized, and potentially persuasive, sales dialogues.

From the business point of view, recommender systems have the potential to increase sales, because purchasing decisions are often strongly influenced by people who the consumer knows and trusts. In the networked virtual world, consumers also need some word of mouth to support their purchasing decisions; thus, the best source will be recommender systems. Recommender systems can integrate information from product rating matrices made by users and user preference similarity matrices [12] to generate personalized recommendations. It can also help corporations to maximize the precision of targeted marketing.

However, in practical recommender systems, insufficient item rating data are a critical problem, which refers to the lack of prior transactional and feedback data that makes it difficult and unreliable to predict which users are similar to a given user [10]. Im and Hars [11] have claimed that the accuracy of a recommender system increases as the total number of users increases. This implies that accuracy of a recommender system decreases when the total number of users is limited. One way to solve this problem is to combine recommender systems if they have similar product sets. By joining recommender systems, the user sets are enlarged, which means that more accurate recommendation can be made and the precision of targeted marketing is enhanced. In combining recommender systems, consumers and companies may worry about the risk of privacy disclosure. It motivates us to address the problem of providing a better recommendation to customers, while the privacy is preserved. We name our system as a privacy-preserving collaborative recommender system.

There are a number of related works [20], [22], [23], [25]. Schafer *et al.* [16] have come up with a detailed taxonomy of recommender systems by analyzing famous e-commerce Web sites, including Amazon.com, CDNOW, eBay, Levi Strauss & Co., Moviefinder.com, and Reel.com. According to their findings, recommender systems can be categorized into nonpersonalized recommendations, attribute-based recommendations, item-to-item correlation, and people-to-people correlation. Each of these taxonomies has different degrees of automation and persistence. For privacy issues, Canny [6] has proposed some schemes for privacy-preserving collaborative filtering. In his schemes, there is a community of users who can compute the aggregation of their private data without disclosing it. Another approach is the randomized perturbation approach proposed by Polat and Du [14]. They deployed a centralized server to store the perturbed numeric ratings, and then used these disguised ratings to provide predictions to users. Berkovsky *et al.* [3] have proposed an obfuscation scheme about accuracy and privacy to decentralize the rating profiles among multiple repositories. Thus, they can have more users to improve accuracy and to mitigate privacy issues. However, their approach cannot achieve 100% accuracy unless all the private data are disclosed. Aimeur *et al.* had recently proposed a privacy-preserving recommender system; however, their aim is to preserve privacy between customers and merchant [1]. Hsieh *et al.* [9] have proposed a scheme based on homomorphic encryption that provides 100% accuracy. In this paper, we will present a

Manuscript received July 14, 2008; revised January 7, 2009 and September 29, 2009; accepted November 28, 2009. Date of publication February 17, 2010; date of current version June 16, 2010. This paper was recommended by Associate Editor M. Last.

J. Zhan is with the National Center for the Protection of Financial Infrastructure, Madison, SD 57042 USA (e-mail: justin.zhan@dsu.edu).

C.-L. Hsieh is with the Grad School of Informatics, Kyoto University, Kyoto 606-8502, Japan.

I-C. Wang, T.-S. Hsu, C.-J. Liau, and D.-W. Wang are with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCC.2010.2040275

more efficient privacy-preserving approach than the existing encryption approaches.

In Section II, we will introduce a recommender system algorithm. In Section III, we will define our problem. In Section IV, we will describe solutions to privacy-preserving collaborative recommender systems. In Section V, we will present the experimental results. We will further discuss the experimental results in Section VI.

II. Recommender Systems

There are two basic entities concerned in a recommender system [16]. The *user* (also referred to as customer) is a person who uses the recommender system to provide his or her opinion and receive recommendation about items. The *item* (also referred to as product) is being rated by users. The inputs of a recommender system are usually arithmetic rating values, which express the users' opinion of items. Ratings are normally provided by the user and follow a specified numerical scale (example: 1: bad to 5: excellent). The outputs of a recommender system can be either predictions or recommendations. The following are the three main processes of recommender systems.

A. Representation

In the original representation, the input data is defined as a collection of numerical ratings of m users on n items, expressed by the mn user–item matrix R in the following example:

user	$item_1$	$item_2$	$item_3$
a_1	−7.82	8.79	−9.66
a_2	4.08	−0.29	6.36
a_3	8.5	4.61	−4.17

We call this user–item matrix of the input dataset, *original representation*. As mentioned earlier, users are not required to provide their opinion on all items. As a result, the user–item matrix is usually sparse, including numerous *no rating* values, making it harder for filtering algorithms to generate satisfactory results. Thus, some technique [2], whose purposes are to reduce the sparsity of the initial user–item matrix, has been proposed in order to improve the results of the recommendation process. The sparsity problem is the problem of having too few ratings, and hence, too few correlations between users. Bergholz addressed [2] this problem by introducing "transitive correlations," a mechanism to increase the number of correlations between existing users and through adding "agents," artificial users that rate in accordance with some predefined preferences. Transitive correlations provide a small help for virtually no price, whereas rating agents improve the coverage of the system significantly, but also have a negative impact on the system performance.

B. Neighborhood Formation

The core step of the recommendation process is determining the similarity between users in the user–item matrix R. Users similar to the active user U_a will form a proximity-based neighborhood with U_a. The active user's neighborhood should then be used in the following step of the recommendation process in order to estimate their possible preferences. Neighborhood formation has been implemented by calculating the similarity between all the users in the user–item matrix R with the help of proximity metrics.

The proximity between two users is usually measured using correlation or cosine measures, which are as follows.

1) *Pearson correlation similarity:* To find the proximity between users U_i and U_k, we can use the Pearson correlation metric as

$$\mathrm{sim}_{ik} = \mathrm{corr}_{ik} = \frac{\sum_{j=1}^{l}(r_{ij} - \overline{r_i})(r_{kj} - \overline{r_k})}{\sqrt{\sum_{j=1}^{l}(r_{ij} - \overline{r_i})^2 \sum_{j=1}^{l}(r_{kj} - \overline{r_k})^2}}.$$

It is important to note that the summations of j are calculated over L items for which both users u_i and u_k have expressed their opinions. Obviously, $L \leq n$, where n represents the number of total items in the user–item matrix R.

2) *Cosine Similarity:* In the n-dimensional item space, users are represented by feature vectors. A user vector consists of n feature slots, one for each available item. The values used to fill these slots can either be the rating r_{ij} that a user u_i provided for the corresponding item, ij, or 0, if no such rating exists. Now, we can compute the proximity between two users u_i and u_k by calculating the similarity between their vectors as the cosine of the angle is formed between them.

Based on the results of Breese *et al.* [4], Pearson Correlation is considered a better metric for similarity calculations in recommender systems. Thus, we will use Pearson correlation similarity.

C. Recommendation Generation

The final step in the recommendation process is to produce either a prediction, which will be a numerical value representing the predicted opinion of the active user, or a recommendation that will be expressed as a list of the top-N items that the active user will appreciate. In both cases, the result should be based on the neighborhood of users. With provision of recommendations, users can have a reliable list of what they want. Thus, a personalized target marketing can be made efficiently.

III. Problems

Let us assume that there are two e-commerce entities, for example, online bookstores, both of which have similar product sets, but with different customer sets. Also, both of them already have their own recommender systems with some data records. In the sense of practical sparsity problem, these two entities want to cooperate with each other to strengthen their recommender system databases and improve the precision of their recommendations for their own customers. There are couple ways to merge recommender systems: vertical, horizontal, and integration. Horizontal collaboration assume that the collaborative parties data contain the same set of items with different records. Vertical collaboration assume that the collaborative parties contain the different set of items. In this paper, we focus on the horizontal collaboration. It is meaningful since new recommender system databases with more ratings of every item can be achieved. For merging the recommender databases, while not disclosing the actual commercial data, we have to check the recommender system algorithm to find the vulnerability of potential privacy disclosure.

Among the three steps in the recommender system algorithm, representation and recommendation generation are related only to the accuracy of recommendations provided to customers. The neighborhood formation step is the source of possible privacy disclosure. In the neighborhood formation step, we measure the proximity between two customers by calculating the Pearson correlation similarity. For example, let us assume that, for item(product) j, the ratings of r_{ij} and r_{kj} are made by users u_i and u_k from different e-commerce entities. While joining these two recommender system databases, we have to share the values $(r_{ij} - \overline{r_i})$ and $(r_{kj} - \overline{r_k})$ with each other. But, with value $(r_{ij} - \overline{r_i})$, others can see how much user i prefers item j compared to their average rating r_i.

IV. Privacy-Preserving Collaborative Recommender System

The database is separated into two parts, Alice and Bob, which both need each other's data to compute the correlation coefficient without disclosing their own data. The privacy issue is on the numerator of the Pearson correlation coefficient, which is a scalar product. Alice has $\overrightarrow{X_a}$ and Bob has $\overrightarrow{X_b}$. We want to know the scalar product of $\overrightarrow{X_a}$ and $\overrightarrow{X_b}$ without disclosing $\overrightarrow{X_a}$ or $\overrightarrow{X_b}$.

A very intuitive way to deal with the problem is only to send anonymous vector of ratings of certain user [19]. However, the problem of this straightforward approach is vulnerable to background knowledge attack where an aggregation of user preference will be useful to reversely identify the identity of a user. In this paper, we will present two approaches: the homomorphic-encryption-based approach and the scalar-product-based approach.

A. Homomorphic Encryption Approach

ElGamal encryption provides the multiplicative homomorphism, where the multiplication of two cypher texts equals the encryption of the multiplication of the plain texts. To compute the Pearson correlation similarity, we need the operators of $(r_{ij} - \overline{r_i})$ from one party and $(r_{kj} - \overline{r_k})$ from the other one. Let us assume that no party wants to take the risk of disclosing customer preferences. Because the computations are multiplications, we can use the homomorphic property of the ElGamal encryption. Two parties can encrypt their data on their own with a public key, and then, after the multiplication computation, the multiplication of two encrypted pieces of data will become the encryption of the multiplication of data. Thus, the private data of two parties can be preserved during the similarity computation. Assume Alice wants to join recommender system databases with entity Bob. In other words, Alice wants to get the statistical information from entity Bob. Then, the algorithm is presented as follows.

1) Alice generates (p, g, x, y) of ElGamal scheme, then publishes (p, g, y) as the public key and keeps x as the secret key.
2) Alice randomly generates a number α, and then calculates (g^α mod$p$) as c_1.
3) Alice encrypts the ratings of $(r_{ij} - \overline{r}_i)$ for user i on item j with k by calculating $((r_{ij} - \overline{r}_i) * y^\alpha \; modp)$ as c_2, where j is from the first item to the last item of Alice.
4) Alice sends the ciphertext $C = (c_1, c_2)$ to Bob.
5) Bob generates another random number β, and then calculates $c'_1 = (g^\beta \text{mod}p), c'_2 = ((r_{kj} - \overline{r}_k) * y^\beta \; \text{mod}p)$ of its own sequence of $(r_{kj} - \overline{r}_k)$ for user k and item j, where j belongs to the same item set of Alice.
6) Bob multiplies c_2 and c'_2, then got $((r_{ij} - \overline{r}_i)(r_{kj} - \overline{r}_k) * y^{\alpha+\beta}$ mod$p$) as $c_2c'_2$.
7) Bob randomly swaps the order of the encrypted data and sends the ciphertext $C' = (c'_1, c_2c'_2)$ and the value of $\Sigma(r_{kj} - \overline{r}_k)^2$ to Alice.
8) Alice decrypts the encryption $c_2c'_2$ with c_1 and c'_1, then retrieves the statistical value of $(r_{ij} - \overline{r}_i)(r_{kj} - \overline{r}_k)$.
9) Finally, Alice uses $(r_{ij} - \overline{r}_i)(r_{kj} - \overline{r}_k)$ and $\Sigma(r_{kj} - \overline{r}_k)^2$ to calculate the Pearson correlation similarity.

Through the aforementioned approaches, the privacy of user preference can be preserved throughout the computation of similarities.

B. Scalar Product Approach

The problem of secure multiparty computation (SMC) was first addressed by Yao in his seminal paper [21]. The solutions are generic and elegant, but their prohibitive cost in protecting privacy makes them unsuitable for large-scale applications. Therefore, practical solutions need to be developed. In this paper, commodity-based scalar product approach, which is proposed by Du and Zhan [7], is adapted in our scenario. The basic secure two-party product protocol proposed in [24] is described as follows.

Protocol π. $f_2(\overrightarrow{x_a}, \overrightarrow{x_b}) \mapsto (y_a, y_b)$, where $\overline{x_a x_b} = y_a + y_b$.

1) The commodity server generates random vectors $\overrightarrow{R_a}$, $\overrightarrow{R_b}$, and random number r_a, and lets $r_b = \overrightarrow{R_a}\overrightarrow{R_b} - r_a$. It then sends $(\overrightarrow{R_a}, r_a)$ to $Ailce$ and $(\overrightarrow{R_b}, r_b)$ to Bob.
2) $Alice$ sends $\overrightarrow{X_a}' = \overrightarrow{X_a} + \overrightarrow{R_a}$ to Bob.
3) Bob sends $\overrightarrow{X_b}' = \overrightarrow{X_b} + \overrightarrow{R_b}$ to $Alice$.
4) Bob computes $t = \overrightarrow{X_a}'\overrightarrow{X_b} + r_b - y_b$ and sends it to $Alice$, where y_b is a randomly generated number.
5) $Alice$ computes $y_a = t - \overrightarrow{X_b}'\overrightarrow{R_a} + r_a$.

The aforementioned protocol has been proven to be information-theoretically secure [18].

The result of the scalar product equals the sum of y_a and y_b. Since the summation and multiplication computation in the numerator of the Pearson correlation formula is a scalar product computation. It is straightforward to adapt the commodity-based scalar product approach, where the raw data from two parties can be preserved and the computation of similarities can be correctly computed. Furthermore, a revised commodity-based scalar product is developed for our scenario. The difference between the commodity-based scalar product and the revised commodity-based scalar product is that, in the revised commodity-based scalar product, the random numbers needed are predistributed in the computation, instead of deploying a commodity server.

V. Performance Evaluation

We implemented both the homomorphic encryption approach and the scalar product approach. Then, we compare ElGamal approach with both the original and revised commodity-based scalar product approaches.

We choose the same database as Hsieh *et al.* [9], which is the "Jester Joke Recommender System," released by K. Goldberg from the University of California at Berkley (http://shadow.ieor.berkeley.edu/humor/). It has 4.1 million continuous ratings (-10.00 to $+10.00$) of 100 jokes from 73 421 users, collected between April 1999 and May 2003. We take the densest subdataset of ratings from 23 500 users who have rated 36 or more jokes, which is a matrix with dimensions of $23\,500 \times 101$. For the representation process of recommendation generation, we add the default value 0 for the items not rated. The first column of every row stores how many items are rated by the user, which is not necessary for computing the Pearson correlation coefficient. Therefore, we simply ignore the first column of the database.

Consider that Alice has one user rating ($\overrightarrow{A}$), while Bob has 23 499 ($\overrightarrow{B_1}$ to $\overrightarrow{B_{23499}}$). Without disclosing their original data, they want to know the Pearson correlation coefficient sim_{AB_1}, sim_{AB_2}, ..., sim_{AB_n} $(1 \leq n \leq 23\,499)$. We implemented these approaches in Ruby(1.8.6 patchlevel 111). Alice's code runs on a server with two AMD Opteron 2220 SE 2.8 GHz processors, and 20-GB DDR2 667 RAM, while Bob's code runs on a server with an Intel Xeon 5160 3.0-GHz processors and 10-GB DDR2 667 RAM. Both the operating systems are FreeBSD 7.0-STABLE. In addition, we have another machine for the commodity-based approach as the commodity server. The specification of which is

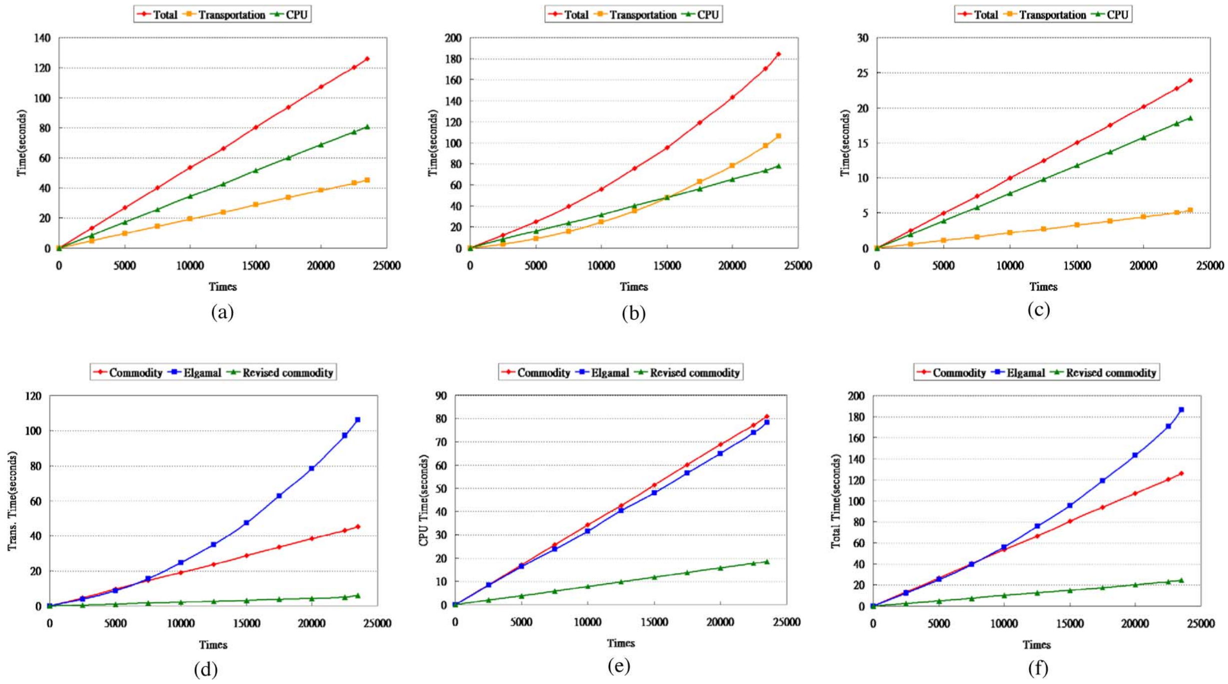

Fig. 1. Experimental results. (a) Commodity approach. (b) ElGamal approach. (c) Revised commodity approach. (d) Transportation time comparison. (e) CPU time comparison. (f) Total execution time comparison.

an Intel(R) Pentium(R) 4 3.40-GHz processor with 3-GB DDR2 667 RAM, and the operating system is Ubuntu 7.10.

To minimize the probabilistic variation, our experimental results are the average of 100 effective executions. The experiments focus on different numbers of user data for secure two-party computation: execution time, transportation time, and CPU time. The amount of CPU time is the aggregation result by adding Alice and Bob's CPU time. The transportation time is equal to the difference of the execution time and the CPU time.

Since the scalar-product-based approach is integer-based, the inputs must be positive integers. Let $X, \min$, and dig represent the original data, the effective minimum rating, and the effective decimal digits of the database, respectively. Let

$$Y = \begin{cases} (X - \min) \times 10^{\mathrm{dig}}, & \text{if } \min < 0 \\ X \times 10^{\mathrm{dig}}, & \text{otherwise} \end{cases}$$

be our integer input data. It is trivial to prove that, after replacing X with Y, they will still have the same Pearson correlation coefficient. The following section introduces each approach that we implemented and show the experimental results, which are as follows.

1) *Commodity approach:* We use *commodity approach* to stand for the original commodity-server-based scalar product approach. Each round can compute a scalar product. In other words, if there are N scalar products needed to be computed, simply run this protocol N times. Fig. 1(a) shows the total execution, transportation, and CPU time of this approach. As expected, all of them are linear.
2) *ElGamal approach:* We implemented the same algorithm as that of Hsieh *et al.* [9] to compare the performance of privacy-preserving recommender systems with that of others. A neutral third party is unnecessary in this approach since only two random numbers are needed for Alice and Bob's private keys. Fig. 1(b) shows the total execution, transportation, and CPU time of this approach.
3) *Revised commodity approach:* It is much more reasonable to make sure that all the needed pieces of data are ready before executing any multiparty computation. As a result, we preproduce and transfer the random numbers to Alice and Bob before executing the tasks. In other words, the commodity server, which is the neutral third party producing only random numbers, is not necessarily included in computing transportation, CPU, and total execution time. Both Alice and Bob know that they are going to do N scalar products, and therefore, they exchange all the needed information in one round, rather than in N rounds. However, additional storage is needed. In order to see what the difference of the performance is between the original commodity approach and the revised one, we decided to keep the original one. Fig. 1(c) shows the total execution, transportation, and CPU time of this approach. As expected, all of them are linear.

VI. Discussion

Fig. 1(d)–(f) compare the results of transportation, CPU, and total execution time among the aforementioned approaches. The revised commodity-based approach has the lowest time cost and needs the least computing resources among the three approaches. Therefore, we can conclude that the revised commodity-based approach has the best performance. However, additional storage is needed for random numbers. At the same time, the stored random numbers must not be disclosed to each party.

Through Fig. 1(a)–(c), we can see that with less numbers of user data, it costs more time on computing than transporting data. However, in ElGamal approach, it costs less time on computing than transporting with larger numbers of user data, which is different from the other two approaches. In Fig. 1(d), the ElGamal's transportation time line is not linear. A possible reason for this is that Ruby uses a fixed-size buffer for input/output (IO). It may become nonlinear once the transportation data becomes too large. To show the total time cost for a

100-D scalar product, we list the experimental results for each approach in the following table.

Approach	ElGamal	Comm.	Revised Comm.
Time(s)	0.10301	0.00515	0.00178

VII. Conclusion and Future Works

Because of the development of e-commerce, recommender systems have become more and more popular. Customers may not trust imprecise recommendations from a recommender system that has a limited database. It is then desirable for e-commerce entities with limited databases to merge their recommender system databases to enhance the reliability of recommendations for customers and maximize the precision of targeted marketing while preserving the privacy of customer preferences. With the algorithms introduced in this paper, e-commerce entities can merge their recommender system databases without disclosing customers' private data. As future works, we will design and implement a prototype of this privacy-preserving collaborative recommender system with the proposed approaches.

References

[1] E. Aimeur, G. Brassard, J. M. Fernandez, and F. S. M. Onana, "Alambic: A privacy-preserving recommender system for electronic commerce," *Int. J. Inf. Secur.*, vol. 7, no. 5, pp. 307–334, 2008.

[2] A. Bergholz, "Coping with sparsity in a recommender system," *Lecture Notes Comput. Sci.*, vol. 2703, pp. 86–99, 2003.

[3] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci, "Enhancing privacy and preserving accuracy of a distributed collaborative filtering," in *Proc. ACM Conf. Recommender Syst. (Proc. RecSys 2007)*, pp. 9–16.

[4] J. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 14th Conf. Uncertainty Artif. Intell. (UAI 1998)*, pp. 43–52.

[5] R. Burke, "Knowledge-based recommender systems," in *Encyclopedia of Library and Information Systems*, A. Kent, Ed. vol. 69, supp. 32, New York: Marcel Dekker, 2000. Available: citeseer.ist.psu.edu/article/burke00knowledgebased.html

[6] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Dev. Inf. Retrieval*, pp. 238–245, 2002.

[7] W. Du and Z. Zhan, "A practical approach to solve secure multi-party computation problems," presented at the ACM New Security Paradigms Workshop, Virginia Beach, VA, Sep. 23–26, 2002.

[8] B. Shapira, P. Shoval, and U. Hanani, "Information filtering: Overview of issues, research and systems," *User Model and User-Adapted Interaction*, vol. 11, no. 3, pp. 203–259, 2001.

[9] C. Hsieh, J. Zhan, D. Zeng, and F. Wang, "Preserving privacy in joining recommender systems," in *Proc. Int. Conf. Inf. Security Assur. (ISA 2008)*, pp. 561–566.

[10] Z. Huang, H. Chen, and D. Zeng, "Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering," *ACM Trans. Inf. Syst.*, vol. 22, pp. 116–142, 2004.

[11] I. Im and A. Hars, "Does a one-size recommendation system fit all? The effectiveness of collaborative filtering based recommendation systems across different domains and search modes," *ACM Trans. Inf. Syst.*, vol. 26, no. 1, 4 pp., 2007.

[12] H. Suwa, H. Yamamoto, I. Okada, Y. Ogawa, and T. Ohta, "Development of recommender systems using user preference tendencies: An algorithm for diversifying recommendation," *Towards Sustainable Soc. Ubiquitous Netw.*, vol. 286, pp. 61–73, 2008.

[13] M. Pazzani and D. Billsus, "Content-based recommendation systems," *Lect. Notes Comput. Sci.*, vol. 4321, pp. 325–341, 2007.

[14] H. Polat and W. Du, "Privacy-preserving collaborative filtering using randomized perturbation techniques," in *Proc. 3rd IEEE Int. Conf. Data Mining (ICDM 2003)*, pp. 625–628.

[15] P. Resnick and H. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, 1997.

[16] J. Schafer, J. Konstan, and J. Riedi, "Recommender systems in e-commerce," in *Proc. 1st ACM Conf. Electron. Commerce*, pp. 158–166.

[17] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating "word of mouth"," in *Proc. ACM 1995 Conf. Human Factors Comput. Syst.*, New York: ACM Press, 1995, pp. 210–217.

[18] C. Shen, J. Zhan, D. Wang, T. Hsu, and C. Liau, "Information theoretically secure number product protocol," in *Proc. Int. Conf. Mach. Learning Cybern.*, Aug. 19–22, 2007, pp. 3006–3011.

[19] L. Sweeney, "k-anonymity: A model for protecting privacy," in *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, 2002.

[20] I. Wang, C. Shen, Z. Zhan, T. Hsu, C. Liau, and D. Wang, "Empirical evaluations of secure scalar product," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 4, pp. 410–419, Jul. 2009.

[21] A. C. Yao, "Protocols for secure computations," in *Proc. 23rd Annu. IEEE Symp. Found. Comput. Sci.*, 1982, pp. 1–5.

[22] J. Zhan, "Privacy-preserving collaborative data mining," *IEEE Comput. Intell. Mag.*, vol. 3, no. 2, pp. 31–41, 2008.

[23] J. Zhan, S. Matwin, and L. Chang, "Privacy-preserving collaborative association rule mining," *J. Netw. Comput. Appl.*, vol. 30, pp. 1216–1227, 2007.

[24] Z. Zhan and L. Chang, "Privacy-preserving collaborative data mining," in *Proc. IEEE Int. Workshop Found. New Directions Data Mining*, Melbourne, FL, Nov. 19–22, 2003, p. 208.

[25] Z. Zhan, I. Wang, C. Hsieh, T. Hsu, C. Liau, and D. Wang, "Towards efficient privacy-preserving recommender systems," in *Proc. IEEE Int. Conf. Granular Comput.*, Hangzhou, China, Aug. 2008, pp. 778–783.