Future Generation Computer Systems 79 (2018) 82–94

Contents lists available at ScienceDirect

Future Generation Computer Systems

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

ARMOR: A trust-based privacy-preserving framework for decentralized friend recommendation in online social networks

Xindi Ma [a,*], Jianfeng Ma [a], Hui Li [a], Qi Jiang [a,b], Sheng Gao [c]

[a] School of Cyber Engineering, Xidian University, Xi'an, China
[b] School of Computer & Software, Nanjing University of Information Science & Technology, Nanjing, China
[c] School of Information, Central University of Finance and Economics, Beijing, China

HIGHLIGHTS

- We propose a privacy-preserving framework to achieve the friend recommendation in OSN.
- We adopt a secure protocol to compute the utilities and the recommender results.
- We also theoretically analyze the efficiency and effectiveness of our framework.
- Evaluation results demonstrate that ARMOR is effective and efficient in recommendation.

ARTICLE INFO

Article history:
Received 12 January 2017
Received in revised form 25 May 2017
Accepted 22 September 2017
Available online 28 September 2017

Keywords:
Friend recommendation
Online social network
Privacy preservation
Trust

ABSTRACT

Friend recommendation in online social networks (OSNs) has recently experienced rapid development and received much research attention. Existing recommender systems on the basis of the big social data mostly employ centralized framework, which would cause lots of problems, such as single point failure, communication bottleneck and so on. Some other studies focus on decentralized framework for recommendation, however, most of them concentrate on the improvement of recommendation quality, while underestimating privacy issues, e.g. OSN users' privacy concerns regarding their social relationships, social attributes, and recommendation profiles. In this paper, we propose a novel decentralized framework, namely ARMOR, which utilizes OSN users' social attributes and trust relationships to achieve the friend recommendation in a privacy-preserving manner. In ARMOR, we adopt a light-weight privacy-preserving protocol to aggregate the utilities of multi-hop trust chains and compute the recommender results securely. We also analyze the efficiency of ARMOR in theory and prove that OSN users' privacy can be preserved. Finally, we conduct an experiment to evaluate ARMOR over a real-world dataset and empirical results demonstrate that our ARMOR can effectively and efficiently recommend friends in a privacy-preserving way.

1. Introduction

With the rapid development of information technology and the proliferation of online social interactions, we are witnessing a widespread popularity of Online Social Networks (OSNs). Similar to what people usually do in real life, OSN users always try to extend their social circles in order to satisfy various social demands, e.g., leisure, business, science, and so on [1].

Friend recommendation is essential for users to enlarge their social circles in OSNs. According to the recommendation model, friend recommendation can be classified into two categories: social graphs based, like friends of friends (FoFs); or big social data based, like tags and blog posts [2]. However, the recommender system based on big social data is always centralized, where a service provider is included and may cause the problems of single point failure and communication bottleneck. What is worse, these recommendations also ignore the social influence, like trust, which is a key driver in motivating users to establish friendships [3].

Sharma et al. [4] found that recommendation algorithms based on FoFs method performed no worse than those based on the full network, even though the FoFs-based recommendation required much less data and computational resources. And we consider that it is more probable a person will know a friend of his friends rather than a random person [5]. So the decentralized friend recommender systems based on the FoFs can provide more valuable recommendations while consuming less resources. However,

[*] Corresponding author.
E-mail addresses: xdma1989@stu.xidian.edu.cn (X. Ma), jfma@mail.xidian.edu.cn (J. Ma), hli@xidian.edu.cn (H. Li), jiangqixdu@xidian.edu.cn (Q. Jiang), sgao@cufe.edu.cn (S. Gao).

https://doi.org/10.1016/j.future.2017.09.060
0167-739X/© 2017 Elsevier B.V. All rights reserved.

the decentralized recommender systems also face another common problem, namely, privacy. For instance, if we want to look for a cardiologist over some professional OSNs, such as PatientsLikeMe, [1] for helpful suggestions and preliminary diagnosis. Without a privacy-preserving mechanism, requesting the recommendation from non-close friends or strangers not only reveals our profiles, but also discloses our private information, such as health conditions and medical information. Even worse, the recommendation approaches [6,7] which apply identity to recommend strangers will disclose OSN users' social relationships to the public, which impede users from utilizing it, and also decrease the possibility of establishing the multi-hop trust chain if one of OSN users on the chain returns a negative result. Therefore, it is crucial to protect user privacy when utilizing the friend recommendation in OSNs. Unfortunately, until now there have been limited research efforts or valuable contributions regarding this aspect. State-of-the-art work either suffers from an inaccurate recommendation quality [8] or low efficiency [9,10].

In this paper, we consider the social influence between OSN users and design a trust-based privacy-preserving framework for decentralized friend recommendation in OSNs (ARMOR). For ARMOR, we design it as a decentralized recommendation framework and consider the possibility of using OSN users' social attributes to establish the multi-hop trust chains based on each context-aware 1-hop trust relationship in a privacy-preserving manner, where the trust relationships are formed and strengthened by the shared social attributes. The main contributions of this paper are as follows.

- We propose a novel framework, namely ARMOR, which utilizes OSN users' social attributes and trust relationships to develop the friend recommender mechanism in a decentralized manner while preserving the privacy of OSN users' trust relationships, social attributes, and profiles.
- To protect the privacy information, we adopt a light-weight privacy-preserving protocol [11] to protect the trust relationships and enable the recommender to aggregate the utilities of multi-hop trust chains securely. Similarly, we also employ the same method to protect OSN users' social attributes. In this manner, the users can establish friendships and derive the recommender results by the secure kNN computations.
- We conduct the analysis of ARMOR in terms of both theory and practice. The results indicate that ARMOR can respond to OSN users' requests efficiently and effectively.

The rest of this paper is organized as follows. Section 2 gives some related work. In Section 3, we present the system overview and problem formulation, followed by the details of ARMOR in Section 4. Section 5 presents the theoretical analysis of privacy and efficiency. In Section 6, we empirically test the performance of our framework. Finally, we conclude this paper in Section 7.

2. Related work

In the past few years, friend recommendation has been recognized as an essential application in OSNs [12]. However, as users' privacy becomes increasingly important, traditional friend recommendation in OSNs is facing a significant challenge, namely how to protect the privacy of users while achieving the effective recommendation. In this section, we review the state-of-the-art researches on friend recommendation and privacy issues in friend recommendation.

2.1. Friend recommendation

The friend recommendation in OSNs can be classified into two categories: friend recommendation based on FoFs and friend recommendation based on big social data. In the following, we will review the related works for the two recommendation model, respectively.

2.1.1. Friend recommendation based on FoFs

Social information, which is a unique feature of OSNs, is highly appropriate information for improving the performance of friend recommender systems. For this reason, many research studies on exploiting this information have been conducted. Seo et al. [13] proposed a friendship strength-based personalized recommender system that recommends topics or interests users might have. They introduced an appropriate measure to calculate the closeness between users in a social circle, namely, the friend strength. Xu et al. [14] discovered the preference of users on microblog based on the information of their connected users. They focused on filtering out unnecessary connected users to predict the preference of specific user, as opposed to general approaches finding relevant users. Rodríguez et al. [15] took into account the interaction and the social circle information of users to calculate the tie strength between them. Future, they proposed a personalized model based on the tie strength to enhance social services. Ma et al. [16] proposed user recommendations on social network service(SNS) considering both the relationship in the social circle and the topic similarity between users. To find good friends in social network, Moricz et al. [17] designed the MySpace friend recommendation algorithm, named People You May Know. Sun et al. [18] also proposed a social event recommendation method that exploits user's social and collaborative friendships to recommend events of interest. Daly and Haahr [19] discussed the establishment of friendship chains using user's "betweenness" centrality and user's social "similarity" to the recommended user. However, all the above works failed to consider users' privacy concerns on both their profiles and social information.

2.1.2. Friend recommendation based on big social data

The friend recommender system based on big social data collects the input information from all users. Using these information, it recommends friends that OSN users may wish to establish friendships. Yin et al. [20] proposed a user behavior model, namely temporal context-aware mixture model. They observed rating behaviors of users based on two factors: user implicit preferences and temporal attentions of the whole social circle on the SNS. By investigating the structure of social networks, Huang et al. [6] correlated different "social role" networks, found their relationships and developed an algorithm for network correlation-based social friend recommendation. Chen and Fong [21] used collaborative filtering (CF) algorithm to recommend OSN users on Facebook, where they analyze the similarity based on users' interests and attributes. By taking advantage of sensor-rich smartphones, Wang et al. [22] collected life styles of users from user-centric sensor data and designed a novel semantic-based friend recommendation system for social networks, which recommends friends to users based on their life styles. To overcome the information overload problem, Chen et al. [23] proposed a learning-based recommendation method which suggests informative friends to users, where an informative friend is a friend whose posted updates are liked by the user. As described above, these friend recommender systems require a heavy toll for collecting the big social data and computing the recommender results. What is worse, the above recommendations do not consider the social influence between OSN users. So these recommendations usually have a lower possibility to find the target user. Finally, the most serious is that the above recommender systems also do not consider the privacy of users during the recommendation.

[1] http://www.patientslikeme.com/.

Table 1
Definitions and notations in ARMOR.

Symbol	Definition
A_i, Q_j	OSN user u_i's social attributes vector and querier u_j's queried vector
$\mathcal{F}_{i,j}$	User u_i's j-th 1-hop friend in OSN
R_i	The i-th recommender
$T_{i,j}$	Trust value that user u_i assigns to user u_j
$r_{i,j}$	Social intimacy degree between user u_i and u_j
ρ_i	Role impact factor of user u_i
ω_T, ω_r, ω_ρ	The weights for T, r, and ρ, respectively
$U_{i,j}$	The aggregated utility that measures the trustworthiness of social trust path from u_i to u_j
$\phi(U_{i,j})$	The utility ratio computed for $U_{i,j}$
σ_i	The ID-based signature by user u_i
$\mathcal{T}_{i,j}$, $\mathcal{T}'_{i,j}$	The random trust value between u_i and u_j
$\mathcal{R}_{i,j}$, $\mathcal{R}'_{i,j}$	The random social intimacy degree between u_i and u_j
$\Re_{i,j}[]$, $\Re'_{i,j}[]$	The random two-tuple role impact between u_i and u_j

2.2. Privacy issues in friend recommendation

Privacy has attracted an increasing concern. A number of approaches have been proposed to address identity privacy [24,25], location privacy [26–28], and search privacy [29–32]. In this paper, we mainly concern preserving the privacy of OSN users' social attributes, profiles, as well as their social relationships [33,34] in friend recommendation. Based on the existed privacy-preserving policies, [1,35–37] proposed some privacy management mechanisms to set privacy policies by learning users' habits in releasing resources in OSN. Polatidis et al. [38] proposed a multi-level privacy-preserving method for collaborative filtering systems by perturbing each rating before it was submitted to the recommender server. Li et al. [39] also designed a user group-based privacy-preserving recommender system for users in OSN, which organizes users into groups with diverse interests and allows users interact with recommender server via interest-specific pseudo. Similarly, Puglisi et al. [40] introduced tag forgery technology to hide users' actual preferences in recommendation process. The authors in [41,42] also proposed two novel methods based on cryptography for privacy-preserving social recommendation. In a strict privacy notion, Chen et al. [43] designed a privacy-preserving approach based on differential privacy to protect user identity in social context recommendation. A centralized privacy-preserving friend recommendation scheme was also presented in [9], where OSN users applies their attributes to find matched friends and extends their social circle with strangers. However, all the works above adopt the centralized architecture and always require a third party to maintain the system during the process. Additionally, in [10,44,45], the authors proposed three privacy-preserving solution for user profile matching in social networks. Samanthula et al. [46] proposed a two-phase private friend recommendation protocol for recommending friends to a given target user based on the network structure as well as utilizing the real message interaction between users. Continuously, the authors in [47] proposed two private friend recommendation algorithms based on the social network structure and the users' social tags. To resist the re-identification attacks, Wang et al. [48] also designed two k-anonymization algorithms to protect the users' identities in OSNs.

3. System overview

As discussed above, friend recommendation in OSNs may invoke unexpected privacy issues, which is a key bottleneck for the development and widespread of OSNs. To this end, we design ARMOR based on a light-weight privacy-preserving protocol to help OSN users recommend friends and establish multi-hop trust chains between strangers. In this section, we first present some preliminaries that serve as the basis of ARMOR, and then present the system model, threat model and design goals for ARMOR. For your convenience, the notations used in the sequel are listed in Table 1.

3.1. Preliminaries

3.1.1. Social attributes

The recommender framework, ARMOR, implements privacy-preserving friend recommendation based on personal social attributes. These attributes may have specific meanings, such as disease symptoms [49], interests, locations [50], affiliations, or friends. In OSNs, each user has a unique vector $A \in \{0, 1\}^n$ to represent his social attributes, and n is the length of the vector. In particular, the system defines a public attribute set consisting of d usual attributes, $\{A^1, A^2, \ldots, A^d\}$. In each attribute, we assigns a unique vector to represent the attribute value, e.g., 0001 denotes the user is an internist, while 0110 denotes a surgeon.

To recommend friends for querier u_i, a similarity score $sim_{i,j}$ is introduced, which is computed as following:

$$sim_{i,j} = Q_i \cdot A_j$$

where Q_i is the queried vector of user u_i and A_j is the target user u_j's social attributes vector. To reduce the computation for users, we use the dot-product of the two vectors to represent the similarity score. In addition, we assume that users' social attributes used for comparing the similarity would uniquely identify some particular users. So the recommender will find out the most similar target user to the querier u_i.

3.1.2. Quality of trust attributes aggregation

In Service-Oriented Computing (SOC), QoS (Quality of Service) consists of a set of attributes, used to illustrate the ability of services to guarantee a certain level of performance. Similar to QoS, we present a new concept, Quality of Trust [51].

Definition 1 (*Quality of Trust (QoT)*). [52] is the ability to guarantee a certain level of trustworthiness in trust propagation along a social trust path, taking trust (T), social intimacy degree (r), and role impact factor (ρ) as attributes.

When recommending the friends in OSN, we consider the trustworthiness between users. So we introduce the QoT to evaluate the trustworthiness for the established relationship. In the following, we present the aggregation methods for QoT attributes in a social trust chain.

- **Trust Aggregation:** The trust value between a source querier and the target user in a social chain can be aggregated based on trust transitivity property [53]. Since trust is discounted with the increase of transitivity hops, we adopt the strategy proposed in [54], where if there are k users $u_1, u_2, \ldots, u_k$ in order in a social trust chain (denoted as $p(u_1, \ldots, u_k)$), the aggregated trust value is calculated as following:

$$T_{1,k} = \prod_{(u_i, u_{i+1}) \in p(u_1, \ldots, u_k)} T_{i,i+1}.$$

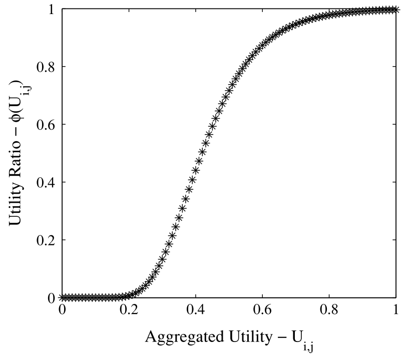


Fig. 1. Gompertz function.

- **Social Intimacy Degree Aggregation:** The social intimacy between users decays with the increasing number of hops between them in a social trust chain. In addition, the decay speed of the social intimacy degree is non-linear in OSN [55]. The aggregated r value in chain $p(u_1, \ldots, u_k)$ can be calculated as following:

$$r_{1,k} = \prod_{(u_i,u_{i+1})\in p(u_1,\ldots,u_k)} r_{i,i+1}.$$

- **Role Impact Factor Aggregation:** In the same society, the role impact factor of a user does not decay with the increase of transitivity hops, which is illustrated in [56]. Thus, the aggregated ρ value of $p(u_1, \ldots, u_k)$ can be calculated as following:

$$\rho_{1,k} = \frac{\sum_{i=2}^{k} \rho_i}{k-1}.$$

- **Utility Function:** In our framework, we introduce the utility function as the measurement of the trustworthiness of social trust chain. Considering the QoT attributes T, r, and ρ, we define the utility function as following:

$$U_{1,k} = \omega_T * T_{1,k} + \omega_r * r_{1,k} + \omega_\rho * \rho_{1,k}$$

where ω_T, ω_r, and ω_ρ are the weights for T, r, and ρ, respectively, $0 < \omega_T, \omega_r, \omega_\rho < 1$ and $\omega_T + \omega_r + \omega_\rho = 1$.

3.1.3. Gompertz function

After computing the utility of the social trust chain, we should use the aggregated utility to reflect the impact of trustworthiness on recommendation. So we introduce the Gompertz function [57] to compute the utility ratio which expresses the trustworthiness of the trust chain for the recommender results. The Gompertz function is usually to construct the reputation model and it has three phases, namely the reputation doubting phase (beginning), the reputation growing phase (middle), and, lastly, the reputation stable phase (end). Recall that, we select Gompertz function to proportion the recommender results, because it is more appropriate to model the trustworthiness during the friend recommendation in OSN. Gompertz function is formally defined as follows and is plotted in Fig. 1.

$$\phi(U_{i,j}) = a \times e^{b\times e^{c\times U_{i,j}}}$$

where a, b, and c are function parameters. In particular, a specifies the upper asymptote, b controls the displacement along the x axis, and c adjusts the growth rate of the function. The output of the function, denoted by $\phi(U_{i,j})$, represents the utility ratio which user u_i assigns to the recommender results with aggregated utility $U_{i,j}$. In this paper, if the querier u_i gets the recommender result with target user u_j, which is represented by the similarity, $sim_{i,j}$, between u_i's social attributes and user u_j's ones, the synthetic recommender result will be computed as $\phi(U_{i,j}) * sim_{i,j}$ when the querier u_i aggregates a utility $U_{i,j}$ for the social trust chain to u_j.

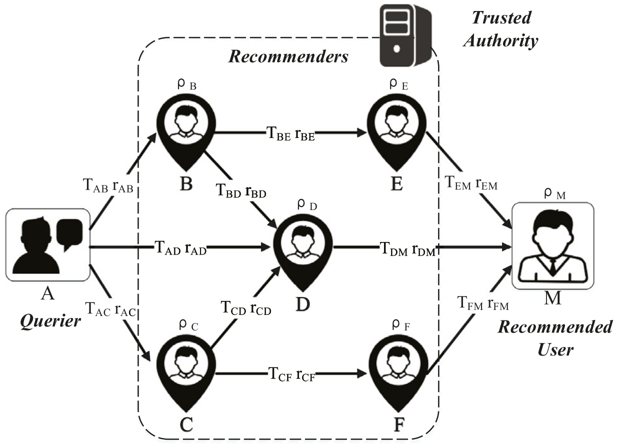


Fig. 2. System framework of ARMOR.

3.2. System model

In this paper, we design the ARMOR based on FoFs and social influence for users to ask for help directly from their 1-hop friends. Before describing ARMOR, we formally present a definition of the general trust-based friend recommendation in OSN as follows.

Definition 2. Given the queried vector Q_i of user u_i as well as the corresponding topology G of OSN, the **trust-based friend recommendation** returns the recommended user u_j in G, who is regarded as the most suitable candidate, by taking into account u_j's social attributes A_j and the aggregated utility $U_{i,j}$ for the social trust chain to u_j.

To guarantee the privacy of such recommendation in OSN, all the privacy information (i.e., Q_i, A_j, $U_{i,j}$, etc.) should be kept in secret. In this manner, we can derive the basic components for our trust-based privacy-preserving friend recommender system as follows (see in Fig. 2):

- **Trusted Authority:** Trusted authority is an indispensable entity which is trusted by all entities. At the beginning of friend recommendation, it generates and distributes the secure parameters and private keys for users in OSN.
- **Querier:** Querier is the user who initiates the friend recommendation process. For example, he initiates a request to find a cardiologist over the PatientsLikeMe for helpful suggestions and preliminary diagnosis. He will first compute the synthetic recommender results with his 1-hop friends and then select the most matching one as his first recommender.
- **Recommender:** Recommender is a user who is 1-hop friend or stranger to the querier and willing to help the querier extend the social trust chain. During the recommendation process, the recommender will select the next recommender from his 1-hop friends until obtain the recommended user.
- **Recommended User:** Recommended user is the one that the querier is looking for. Through the friend recommendation mechanism, the recommended user can be found that he is the best one to meet the querier's demand in OSN.

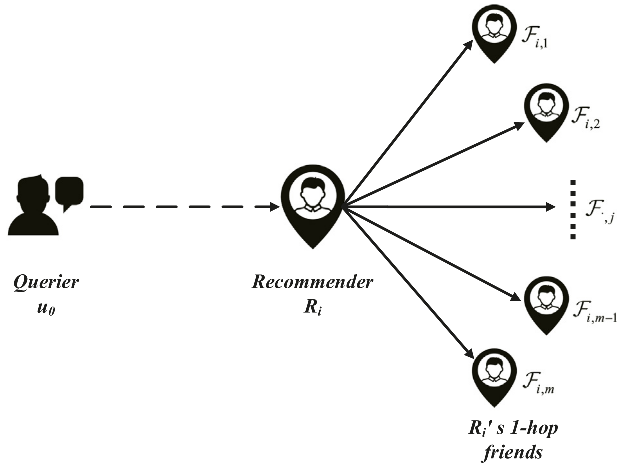


Fig. 3. The process of recommendation.

3.3. Threat model

In ARMOR, malicious users may participate in OSNs and steal information in the phases of recommendation and information delivery. Firstly, we consider the recommenders to be curious-but-honest. During the recommendation process, the recommenders may be curious about users' privacy information, such as social relationships, social attributes, and recommendation profiles. So they may strictly follow the protocols executed in the framework but also violate and disclose users' sensitive and private information. Secondly, external adversaries are also interested in the information transmitted in the recommendation process. These adversaries may forge OSN users' identities and social relationships to capture the delivered information and that will also result in the leakage of privacy.

3.4. Design goals

As a privacy-preserving friend recommender system, ARMOR should fulfill the following requirements.

- **Trust-based Recommendation:** As the main embodiment of social influence in OSNs, the trust relationship is highly appropriate information for improving the performance of friend recommendation. While recommending friends, we should consider the utility of the trust chain to obtain a higher recommendation quality.
- **Recommendation Efficiency:** Due to the timeliness of recommendation services, our goal is to design an efficient friend recommender system to obtain recommender results as soon as possible. The proposed framework should efficiently recommend friends and cost few extra communication and computational overheads.
- **Privacy Preservation:** The proposed mechanism should achieve the privacy requirements in terms of the following aspects [9].
    1. *QoT attributes privacy*. We treat the QoT attributes as private data since it potentially reveals information on friendships and personal social circles. It requires that the QoT attributes between two 1-hop friends cannot be revealed to others during the utility aggregation.
    2. *Social attributes privacy*. Since the social attributes represent users' profiles and preferences, directly revealing one's social attributes would leak his social privacy. It requires that users' social attributes cannot be revealed during the friend recommendation process.

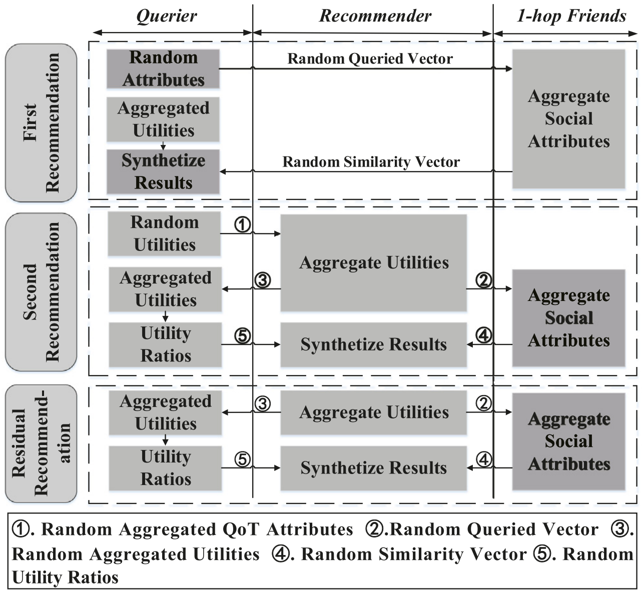


Fig. 4. The overall procedure of ARMOR.

4. Construction of ARMOR framework

In this section, we present the details of ARMOR. As described in Section 3.2, the first recommender is found out by the querier himself. Then the first recommender will look for the next recommender from his 1-hop friends, and the residue will continue, which is shown in Fig. 3. Since the search processes for first and second recommenders are different from others, we divide the recommendation process into the following three phases: first privacy-preserving recommendation, second privacy-preserving recommendation, and residual privacy-preserving recommendation. The overall procedure is shown in Fig. 4.

4.1. First privacy-preserving recommendation

In this phase, the querier will start the friend recommendation and look for the first recommender. During the process, we use u_0 to represent the querier, Q to represent u_0's queried vector, A to represent user's social attribute vector, R_1 to represent the first recommender, and $\mathcal{F}_{0,j}$ to represent u_0's j-th 1-hop friend. Firstly, the querier u_0 aggregates the utilities for his 1-hop friends. And then, the similarities between u_0 and his 1-hop friends will be computed in a privacy-preserving manner. Generally, we assume that each OSN user has m 1-hop friends. Before the utility aggregation, the querier u_0 should set the aggregate weights ω_T, ω_r, and ω_ρ, $\omega_T + \omega_r + \omega_\rho = 1$.

Step-I: While aggregating the utilities for his 1-hop friends, the querier u_0 owns all the QoT attributes. So he can aggregate the utilities directly as following:

$$U_{0,\mathcal{F}_{0,j}} = \omega_T * T_{0,\mathcal{F}_{0,j}} + \omega_r * r_{0,\mathcal{F}_{0,j}} + \omega_\rho * \rho_{\mathcal{F}_{0,j}}$$

where $j \in [1, m]$. After that, the querier u_0 gets a utility vector for his 1-hop friends and we define it as $U_1 \in \mathbb{R}^m$.

Step-II: Then, we compute the similarities with u_0's 1-hop friends through a light-weight privacy-preserving protocol [11]. Given security parameters k_1, k_2, k_3 and k_4, two large primes p and α_2, such that $|p| = k_1$, $|\alpha_2| = k_2$, a large random number $s_2 \in Z_p$, and $n+2$ random numbers e_v, $v = 1, 2, \ldots, n+2$, with $|e_v| = k_3$. To preserve the privacy, we also set $q_{n+1} = q_{n+2} = 0$. For each $q_v \in Q$, $v \in [1, n+2]$, the querier u_0 calculates it as follows:

$$f_v = \begin{cases} s_2(\alpha_2 q_v + e_v) \bmod p, & q_v \neq 0 \\ s_2 e_v \bmod p, & q_v = 0 \end{cases}.$$

All f_v, $v \in [1, n+2]$, form a new vector $F \in \mathbb{R}^{n+2}$. Afterward, the querier u_0 sends F and parameter α_2 to each of his 1-hop friend.

Step-III: When u_0's j-th 1-hop friend $\mathcal{F}_{0,j}$, $j \in [1, m]$, receives the random queried vector F, he will extend his social attributes and set $a_{j,n+1} = a_{j,n+2} = 0$. Then, for each $a_{j,v} \in A$, $v \in [1, n+2]$, $\mathcal{F}_{0,j}$ aggregates the social attributes as follows:

$$d_{j,v} = \begin{cases} a_{j,v}\alpha_2 f_v \bmod p, & a_{j,v} \neq 0 \\ w_{j,v} f_v \bmod p, & a_{j,v} = 0 \end{cases}$$

where $w_{j,v}$ is a random number, with $|w_{j,v}| = k_4$. After that, $\mathcal{F}_{0,j}$ calculates the random similarity as $D_j = \sum_{v=1}^{n+2} d_{j,v} \bmod p$ and sends D_j back to u_0.

Step-IV: When receiving the random similarities from his 1-hop friends, u_0 will conduct the following for $j \in [1, m]$:

$$sim_{0,\mathcal{F}_{0,j}} = \frac{{s_2}^{-1} D_j - {s_2}^{-1} D_j \bmod \alpha_2^2}{{\alpha_2}^2} = \sum_{v=1}^{n} q_v a_{j,v}.$$

Step-V: Because the utility for each 1-hop friend has been aggregated, the querier u_0 will directly compute the utility ratios in accordance with the *Gompertz Function* [57] and we represent the ratio for j-th 1-hop friend as $\phi(U_{0,\mathcal{F}_{0,j}})$, $j \in [1, m]$. Then the querier u_0 calculates the synthetic recommender results as $P_{0,j} = \phi(U_{0,\mathcal{F}_{0,j}}) * sim_{0,\mathcal{F}_{0,j}}$, $j \in [1, m]$. According to the synthetic results, u_0 will select the most matching 1-hop friend as the first recommender, represented as R_1, and set the corresponding synthetic result as $\mathcal{P}_1$.

4.2. Second privacy-preserving recommendation

When the querier u_0 finds out the first recommender, he will look for the next recommender from R_1's 1-hop friends. In the following phases, the utilities and similarities must be aggregated in a privacy-preserving manner. We define the second recommender as R_2 and look for him in the following way:

Step-I: Given a large prime α_1, such that $|\alpha_1| = k_2$, a large random number $s_1 \in Z_p$, and four random numbers c_1, c_2, c_3, c_4, with $|c_1| = |c_2| = |c_3| = |c_4| = k_3$. For the QoT attributes which are assigned to first recommender, the querier u_0 calculates them as follows:

$$\begin{cases} \mathcal{T}_{0,1} = s_1(\alpha_1 \omega_T T_{0,1} + c_1) \bmod p \\ \mathcal{R}_{0,1} = s_1(\alpha_1 \omega_r r_{0,1} + c_2) \bmod p \\ \mathfrak{R}_{0,1} = s_1(\alpha_1 \omega_\rho \rho_1 + c_3) \bmod p \end{cases}$$

where $T_{0,1}$, $r_{0,1}$, ρ_1 represent the QoT attributes which are assigned to R_1 by querier u_0.

Step-II: To aggregate the role impact correctly, we also introduce a parameter 1 to form a two-tuple with ρ_1 and random it as following:

$$\mathfrak{R}_{0,1}[1] = s_1(\alpha_1 \omega_\rho + c_4) \bmod p.$$

So the random two-tuple role impact can be rewritten as $\mathfrak{R}_{0,1} = \{s_1(\alpha_1 \omega_\rho \rho_1 + c_3) \bmod p, s_1(\alpha_1 \omega_\rho + c_4) \bmod p\}$. After that, the querier u_0 keeps $s_1^{-1} \bmod p$ secret and sends the random QoT attributes $\{\mathcal{T}_{0,1}, \mathcal{R}_{0,1}, \mathfrak{R}_{0,1}\}$ and α_1 to recommender R_1 to help to aggregate the utilities for R_1's 1-hop friends.

Step-III: In ARMOR, the recommender R_1 owns all the QoT attributes of his 1-hop friends, $T_{1,j}, r_{1,j}, \rho_{\mathcal{F}_{1,j}}$, $j \in [1, m]$. To guarantee the aggregation correctly, we also extend the role impact to $\{1, \rho_{\mathcal{F}_{1,j}}\}$, $j \in [1, m]$. While receiving the random attributes from u_0, the recommender R_1 will aggregate the QoT attributes as follows, $j \in [1, m]$:

$$\begin{cases} \mathcal{T}'_{2,j} & = \mathcal{T}_{0,1} * T_{1,j} = s_1 T_{1,j}(\alpha_1 \omega_T T_{0,1} + c_1) \bmod p \\ \mathcal{T}_{2,j} & = \mathcal{T}_{0,1} * \alpha_1 T_{1,j} = s_1 \alpha_1 T_{1,j}(\alpha_1 \omega_T T_{0,1} + c_1) \bmod p \\ \mathcal{R}'_{2,j} & = \mathcal{R}_{0,1} * r_{1,j} = s_1 r_{1,j}(\alpha_1 \omega_r r_{0,1} + c_2) \bmod p \\ \mathcal{R}_{2,j} & = \mathcal{R}_{0,1} * \alpha_1 r_{1,j} = s_1 \alpha_1 r_{1,j}(\alpha_1 \omega_r r_{0,1} + c_2) \bmod p \\ \mathfrak{R}'_{2,j}[0] & = \mathfrak{R}_{0,1} \cdot \{1, \rho_{\mathcal{F}_{1,j}}\} = (s_1(\alpha_1 \omega_\rho \rho_1 + c_3) \\ & \quad + s_1 \rho_{\mathcal{F}_{1,j}}(\alpha_1 \omega_\rho + c_4)) \bmod p \\ \mathfrak{R}'_{2,j}[1] & = s_1(\alpha_1 \omega_\rho + c_4) \bmod p \\ \mathfrak{R}_{2,j} & = \dfrac{\mathfrak{R}_{0,1} \cdot \alpha_1 \{1, \rho_{\mathcal{F}_{1,j}}\} \bmod p}{2} \\ & = \dfrac{(s_1 \alpha_1(\alpha_1 \omega_\rho \rho_1 + c_3) + s_1 \alpha_1 \rho_{\mathcal{F}_{1,j}}(\alpha_1 \omega_\rho + c_4)) \bmod p}{2} \end{cases}.$$

Then the random utility can be aggregated as following, $j \in [1, m]$:

$$\begin{aligned} U^R_{0,\mathcal{F}_{1,j}} &= (\mathcal{T}_{2,j} + \mathcal{R}_{2,j} + \mathfrak{R}_{2,j}) \bmod p \\ &= s_1 \left(\alpha_1^2 \left(\omega_T T_{0,1} T_{1,j} + \omega_r r_{0,1} r_{1,j} + \omega_\rho \frac{\rho_1 + \rho_{\mathcal{F}_{1,j}}}{2} \right) \right. \\ &\quad \left. + \alpha_1 \left(c_1 T_{1,j} + c_2 r_{1,j} + \frac{c_3 + c_4 \rho_{\mathcal{F}_{1,j}}}{2} \right) \right) \bmod p. \end{aligned}$$

Afterward, we define the random aggregated utilities as $U^R_2 \in \mathbb{R}^m$.

Step-IV: During the last recommendation, the querier u_0 has sent the random queried vector F and parameter α_2 to R_1. In addition, when it is determined that R_1 is the first recommender, u_0 will also send the synthetic result $\mathcal{P}_1$ to him. When computing the similarity scores with his 1-hop friends, R_1 sends F and α_2 to each of his 1-hop friend. And simultaneously, R_1 will also send U^R_2 to querier u_0. After that, all the 1-hop friends $\mathcal{F}_{1,*}$ will extend their social attributes and aggregate them as follows, $j \in [1, m]$, $a_{j,v} \in A$, $v \in [1, n+2]$:

$$d_{j,v} = \begin{cases} a_{j,v}\alpha_2 f_v \bmod p, & a_{j,v} \neq 0 \\ w_{j,v} f_v \bmod p, & a_{j,v} = 0 \end{cases}$$

where $w_{j,v}$ is a random number generated by $\mathcal{F}_{1,j}$ with $|w_{j,v}| = k_4$. $\mathcal{F}_{1,j}$ calculates $D_j = \sum_{v=1}^{n+2} d_{j,v} \bmod p$ and sends D_j back to R_1.

Step-V: After receiving the random aggregated utilities U^R_2, the querier u_0 will conduct the following for $j \in [1, m]$:

$$\begin{aligned} U_{0,\mathcal{F}_{1,j}} &= \frac{s_1^{-1} * U^R_{0,\mathcal{F}_{1,j}} - s_1^{-1} * U^R_{0,\mathcal{F}_{1,j}} \bmod \alpha_1^2}{\alpha_1^2} \\ &= \omega_T T_{0,1} T_{1,j} + \omega_r r_{0,1} r_{1,j} + \omega_\rho \frac{\rho_1 + \rho_{\mathcal{F}_{1,j}}}{2}. \end{aligned}$$

We represent the above computed utilities as $U_2 \in \mathbb{R}^m$. And after that, u_0 will also compute the utility ratio and send ${s_2^{-1}}\phi(U_2)$ to R_1.

Step-VI: When receiving the information from 1-hop friends and u_0, R_1 will synthesize the recommender results as following, $j \in [1, m]$:

$$\begin{aligned} P_{1,j} &= \frac{s_2^{-1}\phi(U_{0,\mathcal{F}_{1,j}}) * D_j - (s_2^{-1}\phi(U_{0,\mathcal{F}_{1,j}}) * D_j) \bmod \alpha_2^2}{\alpha_2^2} \\ &= \sum_{v=1}^{n} q_v a_{j,v}. \end{aligned}$$

According to the above synthetic results, R_1 will select the most matching 1-hop friend and compare his synthetic result with $\mathcal{P}_1$. If $\mathcal{P}_1$ is less, we will set this user as the second recommender and set the corresponding synthetic result as $\mathcal{P}_2$. Otherwise, we believe that R_1 is the recommended user for u_0 in OSN.

4.3. Residual privacy-preserving recommendation

If R_1 is not the recommended user, R_2 has to continue the recommendation process and judges whether he is the recommended one. Because the remainder recommendations are conducted in the same way, we assume that recommender R_{i-1}, $i \geq 2$, sets his h-th 1-hop friend as the i-th recommender in last recommendation, represented as R_i. Then, R_{i-1} sends the aggregated QoT attributes $\{\mathcal{T}'_{i,h}, \mathcal{R}'_{i,h}, \mathfrak{R}'_{i,h}\}$, $\mathcal{P}_i$, and parameter α_1 to R_i. After that, R_i look for the next recommender as follows:

Step-I: When receiving the information from R_{i-1}, the recommender R_i will aggregate the QoT attributes for each of his 1-hop friend as follows, $j \in [1, m]$:

$$\begin{cases} \mathcal{T}'_{i+1,j} & = \mathcal{T}'_{i,h} * T_{i,j} \\ \mathcal{T}_{i+1,j} & = \mathcal{T}'_{i,h} * \alpha_1 T_{i,j} \\ \mathcal{R}'_{i+1,j} & = \mathcal{R}'_{i,h} * r_{i,j} \\ \mathcal{R}_{i+1,j} & = \mathcal{R}'_{i,h} * \alpha_1 r_{i,j} \\ \mathfrak{R}'_{i+1,j}[0] & = \mathfrak{R}'_{i,h} \cdot \{1, \rho_{\mathcal{F}_{i,j}}\} \\ \mathfrak{R}'_{i+1,j}[1] & = s_1(\alpha_1 \omega_\rho + c_4) \bmod p \\ \mathfrak{R}_{i+1,j} & = \mathfrak{R}'_{i,h} \cdot \alpha_1 \{1, \rho_{\mathcal{F}_{i,j}}\}/(i+1) \end{cases}.$$

Then, the above equations can be conducted as follows, $j \in [1, m]$:

$$\begin{cases} \mathcal{T}'_{i+1,j} & = s_1 T_{i,j} \prod_{l=1}^{i-1} T_{l,l+1}(\alpha_1 \omega_T T_{0,1} + c_1) \bmod p \\ \mathcal{T}_{i+1,j} & = s_1 \alpha_1 T_{i,j} \prod_{l=1}^{i-1} T_{l,l+1}(\alpha_1 \omega_T T_{0,1} + c_1) \bmod p \\ \mathcal{R}'_{i+1,j} & = s_1 r_{i,j} \prod_{l=1}^{i-1} r_{l,l+1}(\alpha_1 \omega_r r_{0,1} + c_2) \bmod p \\ \mathcal{R}_{i+1,j} & = s_1 \alpha_1 r_{i,j} \prod_{l=1}^{i-1} r_{l,l+1}(\alpha_1 \omega_r r_{0,1} + c_2) \bmod p \\ \mathfrak{R}'_{i+1,j}[0] & = \left(s_1 \left(\alpha_1 \omega_\rho \left(\sum_{l=1}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}\right) + c_3\right)\right. \\ & \quad \left. + s_1 c_4 \left(\sum_{l=2}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}\right)\right) \bmod p \\ \mathfrak{R}'_{i+1,j}[1] & = s_1(\alpha_1 \omega_\rho + c_4) \bmod p \\ \mathfrak{R}_{i+1,j} & = \left(s_1 \left({\alpha_1}^2 \omega_\rho \frac{\sum_{l=1}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}}{i+1}\right.\right. \\ & \quad \left.\left. + \frac{\alpha_1(c_3 + c_4(\sum_{l=2}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}))}{i+1}\right)\right) \bmod p \end{cases}.$$

Step-II: After that, the random utilities from querier u_0 to R_i's 1-hop friends can be aggregated as following, $j \in [1, m]$:

$$\begin{aligned} U^R_{0,\mathcal{F}_{i,j}} &= (\mathcal{T}_{i+1,j} + \mathcal{R}_{i+1,j} + \mathfrak{R}_{i+1,j}) \bmod p \\ &= s_1 \left(\alpha_1^2 \left(\omega_T T_{i,j} \prod_{l=0}^{i-1} T_{l,l+1} + \omega_r r_{i,j} \prod_{l=0}^{i-1} r_{l,l+1}\right.\right. \\ &\quad \left. + \omega_\rho \frac{\sum_{l=1}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}}{i+1}\right) + \alpha_1 \left(c_1 T_{i,j} \prod_{l=1}^{i-1} T_{l,l+1}\right. \\ &\quad \left.\left. + c_2 r_{i,j} \prod_{l=1}^{i-1} r_{l,l+1} + \frac{c_3 + c_4(\sum_{l=2}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}})}{i+1}\right)\right) \bmod p. \end{aligned}$$

The aggregated results are denoted as $U^R_{i+1} \in \mathbb{R}^m$.

In the following, the recommender R_i will compute the similarities and synthetic results in the privacy-preserving manner. Because the processing procedure is similar with the **Steps-IV–VI** in Section 4.2, so we will not repeat the description here.

Since the above calculations contain *mod* operation, so we should define the following constraints for $j \in [1, m]$ to guarantee the correct result. If not, we will lose the quotients in *mod* operation, resulting in incorrect result.

$$\begin{cases} p > s_1 \alpha_1 T_{i,j} \prod_{l=1}^{i-1} T_{l,l+1}(\alpha_1 \omega_T T_{0,1} + c_1) \\ p > s_1 \alpha_1 r_{i,j} \prod_{l=1}^{i-1} r_{l,l+1}(\alpha_1 \omega_r r_{0,1} + c_2) \\ p > s_1 \left({\alpha_1}^2 \omega_\rho \frac{\sum_{l=1}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}}{i+1} + \frac{\alpha_1(c_3 + c_4(\sum_{l=2}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}}))}{i+1}\right) \\ p > s_2^{-1} \phi(U_{0,\mathcal{F}_{i,j}}) * D_j \\ \alpha_1 > c_1 T_{i,j} \prod_{l=1}^{i-1} T_{l,l+1} + c_2 r_{i,j} \prod_{l=1}^{i-1} r_{l,l+1} + \frac{c_3 + c_4(\sum_{l=2}^{i} \rho_l + \rho_{\mathcal{F}_{i,j}})}{i+1} \\ {\alpha_2}^2 > \sum_{a_{j,v} \neq 0} \phi(U_{0,\mathcal{F}_{i,j}}) e_v a_{j,v} \alpha_2 + \sum_{q_v \neq 0, a_{j,v} = 0} \phi(U_{0,\mathcal{F}_{i,j}}) w_{j,v} \\ \quad (q_v \alpha_2 + e_v) + \sum_{q_v = 0, a_{j,v} = 0} \phi(U_{0,\mathcal{F}_{i,j}}) e_v w_{j,v} \end{cases}.$$

To resist the outside forgery attack, we also introduce the ID-based signature to verify the truthfulness of the information. During the process of the interaction, we sign the information to ensure that the interacting users are legitimate people within the OSN rather than external attackers. For example, the querier u_0 can verify that the signatured information $\sigma_{R_i}(U^R_{i+1})$ was not sent by the real recommender R_i. Similarly, the adjacent recommenders can also guarantee the authenticity of interaction in the same way. In summary, we show the detailed process of utility aggregation and friend recommendation in Fig. 5, where u_0 and R_{i-1} in the lower right corner of the sent messages represent that these messages are sent by u_0 or R_{i-1}.

5. Theoretical analysis

In this section, we theoretically show that ARMOR fulfills the privacy and efficiency requirements illustrated in Section 3.4.

5.1. Privacy preservation

To study the security of ARMOR, we adopt a simulation model [58,59] that is defined in secure two-party protocols for semi-honest adversaries and widely used to prove the security of multi-party protocols. Intuitively, we say a protocol is secure if each party participating in it can be computed based on its input and output only. We require that a party's view in a protocol execution is simulated only when the input and output are given. This implies that the parties learn nothing from the execution of the protocol itself.

Theorem 1. *The friend recommendation in ARMOR is secure in curious-but-honest model.*

Proof. The proof is given in Appendix.

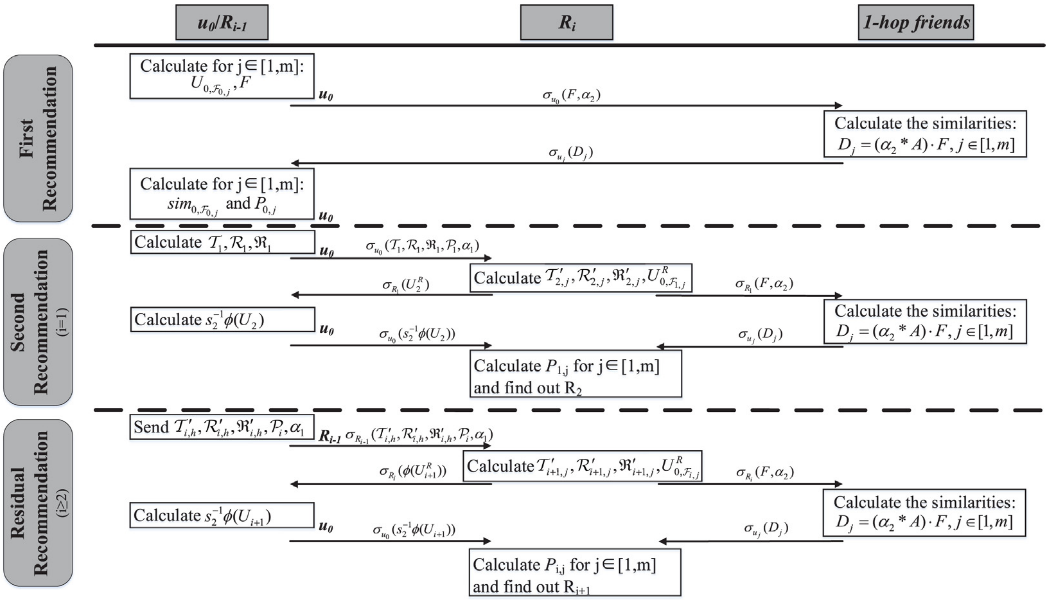

u_0/R_{i-1}
R_i
1-hop friends
First Recommendation
Calculate for j∈[1,m]: $U_{0,\mathcal{F}_{0,j}}, F$
u_0
$\sigma_{u_0}(F,\alpha_2)$
Calculate the similarities: $D_j=(\alpha_2 * A)\cdot F, j\in[1,m]$
$\sigma_{u_j}(D_j)$
Calculate for j∈[1,m]: $sim_{0,\mathcal{F}_{0,j}}$ and $P_{0,j}$
u_0
Second Recommendation (i=1)
Calculate $\mathcal{T}_1,\mathcal{R}_1,\Re_1$
u_0
$\sigma_{u_0}(\mathcal{T}_1,\mathcal{R}_1,\Re_1,\mathcal{P}_1,\alpha_1)$
Calculate $\mathcal{T}'_{2,j},\mathcal{R}'_{2,j},\Re'_{2,j},U^R_{0,\mathcal{F}_{1,j}}$
$\sigma_{R_1}(U^R_2)$
$\sigma_{R_1}(F,\alpha_2)$
Calculate $s_2^{-1}\phi(U_2)$
u_0
$\sigma_{u_0}(s_2^{-1}\phi(U_2))$
Calculate the similarities: $D_j=(\alpha_2 * A)\cdot F, j\in[1,m]$
$\sigma_{u_j}(D_j)$
Calculate $P_{1,j}$ for j∈[1,m] and find out R_2
Residual Recommendation (i≥2)
Send $\mathcal{T}'_{i,h},\mathcal{R}'_{i,h},\Re'_{i,h},\mathcal{P}_i,\alpha_1$
R_{i-1} $\sigma_{R_{i-1}}(\mathcal{T}'_{i,h},\mathcal{R}'_{i,h},\Re'_{i,h},\mathcal{P}_i,\alpha_1)$
Calculate $\mathcal{T}'_{i+1,j},\mathcal{R}'_{i+1,j},\Re'_{i+1,j},U^R_{0,\mathcal{F}_{i,j}}$
$\sigma_{R_i}(\phi(U^R_{i+1}))$
$\sigma_{R_i}(F,\alpha_2)$
Calculate $s_2^{-1}\phi(U_{i+1})$
u_0
$\sigma_{u_0}(s_2^{-1}\phi(U_{i+1}))$
Calculate the similarities: $D_j=(\alpha_2 * A)\cdot F, j\in[1,m]$
$\sigma_{u_j}(D_j)$
Calculate $P_{i,j}$ for j∈[1,m] and find out R_{i+1}

Fig. 5. Utility aggregation and friend recommendation.

5.2. Efficiency analysis

In this section, we study the communication costs in ARMOR. During the recommendation, we assume that each user has m 1-hop friends. At the beginning of utility aggregation, u_0 can finish the first aggregation himself, which costs 0 to transmit. In the second aggregation, u_0 processes R_1's QoT attributes and spends $O(1)$ to transmit the processed QoT attributes to R_1. Afterward, R_1 aggregates the utilities for his 1-hop friends and sends the aggregated results to u_0, which costs $O(m)$. Then, in the residual aggregations, R_{i-1} sends the previous aggregated QoT attributes to R_i, which spends $O(1)$, and R_i costs $O(m)$ to transmit the aggregated utilities to u_0. We assume that there are k utility aggregations in the social trust chain to recommended user. So the total communication cost for utility aggregation is $O((k-1)*m)$. Notably, a traditional scheme without any privacy-preserving technique also requires $O((k-1)*m)$ communication overhead in utility aggregation.

When u_0 aggregates the social attributes, he will send his random queried vector to his 1-hop friends $\mathcal{F}_{0,*}$ in first recommendation, which costs $O((n+2)*m)$ to transmit. Then, $\mathcal{F}_{0,*}$ aggregate their social attributes and spend $O(m)$ to transmit the random similarities to u_0. In the residual recommendation, R_i will compute the similarities with his 1-hop friends, which spends $O((n+2)*m+m)$ to transmit information interactively. After that, u_0 will also spend $O(m)$ to send the computed utility ratio to R_i. If the social trust chain still owns k recommenders, the total communication cost for social attributes aggregation is $O(k(n+4)*m-m)$. Notably, the total communication cost of a traditional scheme without any privacy-preserving technique for social attributes aggregation is also $O(k(n+4)*m-m)$.

6. Performance evaluation

In this section, we present a series of empirical results of ARMOR conducted over a real-world dataset, which indicate that ARMOR can effectively and efficiently fulfill the design goals described in Section 3.4. The experiments were conducted on a machine with a 3.2 GHz quad-core processor and 8GB RAM.

Dataset. We adopt a real dataset which contains the Facebook networks for 100 colleges and universities [60]. Based on that, we select three university OSNs to evaluate the performance of our ARMOR and highlight the social contents of OSNs in Table 2.

Table 2
Facebook dataset.

University name	Reed	Caltech	Haverford
Number of users	962	769	1446
Number of existing friendships	37 624	33 312	119 178
Number of possible friendships	924 482	590 592	2 089 470
Social attributes used/total	7/7	7/7	7/7

6.1. Recommendation quality

In the experimental evaluation, we mainly focus on analyzing the recommendation quality and reachability between two arbitrary users in the Facebook dataset. To measure the quality, we first randomly select 200 users from the above three university OSNs and extract their 50% friendships which are viewed as the ground-truth, respectively. Then we compute their average accuracy and reachability after 20 000 simulated recommendation processes. The following measurement metrics are used for accuracy evaluation:

- *Recommendation precision R_p*. The average of precisions for the recommendations over 200 randomly selected users.

  $$R_p = \frac{\sum_i |F_i \cap G_i| / |F_i|}{200}$$

  where $|\cdot|$ denotes the number of elements in the set, F_i denotes the established friendships by the recommendation, G_i denotes the true friendships of selected user u_i. In another word, the numerator is in fact the sum of precisions (i.e., $\sum_i |F_i \cap G_i| / |F_i|$) over all selected users, and the dominator is 200 because R_p is the average of 200 users in one experiment.
- *Recommendation recall R_r*. The average of recalls for the recommendations over 200 randomly selected users.

  $$R_r = \frac{\sum_i |F_i \cap G_i| / |G_i|}{200}.$$

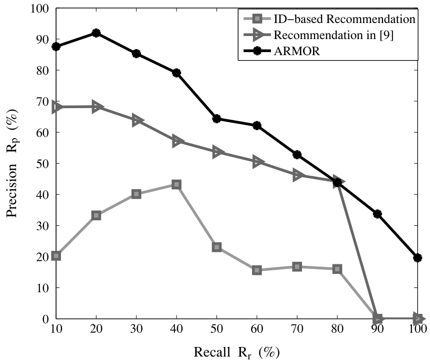


Fig. 6. Comparison of recommendation quality.

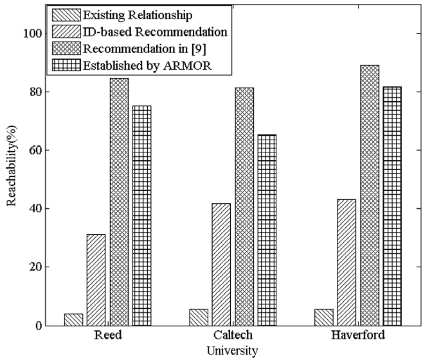


Fig. 7. Comparison of reachability.

We also use the following metric to evaluate the reachability between users:

$$\Gamma = \frac{\sum_{i\in[1,num],j\in[1,num],i\neq j} \varepsilon_{ij}}{num * (num - 1)}$$

where *num* is the *Number of users*, and ε_{ij} is the 1-hop relationship between user u_i and u_j in the OSN. If there is a 1-hop relationship between u_i and u_j, $\varepsilon_{i,j}$ will be set to 1, otherwise set to 0. For the existing relationships and possible relationships, we consider them as asymmetric pairwise relationships.

First of all, we carry out the analysis on the recommendation quality of ARMOR based on the collected dataset and compare it with other recommendation mechanisms. We mainly compare our framework with the ID-based recommendation approach [7] and a privacy-preserving recommendation approach in [9]. As shown in Fig. 6, the precision–recall curves represent the recommendation quality of three recommendation mechanisms. From the evaluation results, we find that our ARMOR owns the highest recommendation quality, as it exhibits the largest AUC(area under the curve). Since the traditional ID-based recommendation mechanism only considers the matching of identity, it has the worst recommendation quality. Similarly, the recommendation approach in [9] neglects the influence of trust-chain utility on the recommender results, it also has a poorer recommendation quality. In addition, with the varying of recall R_r, the precision R_p will decrease on the whole. The reason for this is that the querier establishes many friendships outside the recommendation chains during the recommendation process, which will increase the reachability between users and make it easier for users to recommend. So the established friendships in true set G will account for a less proportion with the processing of recommendation.

In the following, we also compare the reachability of ARMOR with the non-recommendation performance (as the baseline) and the above referred two recommendation mechanisms. We analyze the reachability based on the collected datasets from three universities: Reed College, California Institute of Technology, and Haverford College. As shown in Fig. 7, the non-recommendation only owns the reachability as 4.07 percent, 5.64 percent, and 5.7 percent in three datasets. In addition, since traditional ID-based recommendation mechanism lacks of the ability of extending recommendation chains,it has the lowest reachability as 41.65 percent, 31.15 percent, and 43.21 percent, respectively. The recommendation scheme in [9] has the best performance and our ARMOR is a little bit inferior. The reason for that is our framework considers the aggregated utility into the synthetic recommender results and that will filter out some "unqualified" recommenders. So some friendships will not be established and the reachability is lower.

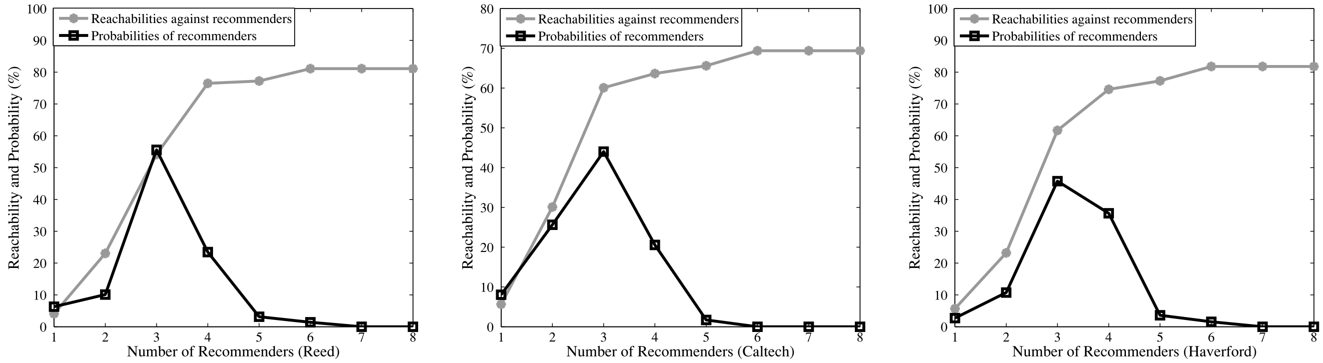
Among all the trust chains established between OSN users, we also investigate the reachability with the increasing of recommenders and the distribution of the number of recommenders on each trust chain. As shown in Fig. 8, with the increasing of recommenders in trust chain, the reachability will also increase quickly and reach the stabilization when the trust chain ends. While a new recommender is found, many new friendships will be established between the querier and users. However, rare recommendations can reach the 5 or 6 hops, so the reachabilities will increase slowly in the end. This phenomenon is strictly consistent with the well-known theory, namely 6-degree of separation [61]. Additionally, from the evaluation results in Fig. 8, we also find that most of the recommendations only need three or four hops and most of the newly established trust chains require less than four hops to complete the recommendation process, which are 95.45 percent, 98.29 percent, and 94.85 percent for Reed, Caltech, and Haverford, respectively. That just coincides with the realistic situation of friendships in OSNs [62].

6.2. Recommendation efficiency

To test the efficiency of our ARMOR, we first discuss the computation costs for the three datasets by varying with the number of recommenders. Then we take the Haverford dataset as an example to discuss the computation cost of ARMOR in different stages. Varying with the number of users in OSN, we evaluate the run time of four stages in ARMOR for once recommendation process, including the run time for computing the random aggregated utility (@R_i), the run time for computing utility ratio (@u_0), the run time for aggregating the social attributes (@1-hop friends), and the run time for synthesizing the recommender results (@R_i). It should be noted that all the timing reported are averaged over 200 randomized runs. The evaluation results are shown in Fig. 9.

In Fig. 9(a), we plot the run time of the recommendation by varying with the number of recommenders. The simulation results show that the computation cost clearly increase with the number of recommenders. As the number of recommenders increases, more and more users will join the recommendation and ARMOR will take more time to compute the utilities and the recommender results. Additionally, we also find that ARMOR always takes more time on Haverford dataset than the remaining two datasets. The reason for that is Haverford has more users in the dataset and each user may have more 1-hop friends to aggregate the utilities and the social attributes. So our ARMOR will spend more time on recommendation for the users in Haverford dataset.

In Fig. 9(b), we plot the run time of the stages for once recommendation by varying with the number of users. The simulation results show that the run time for aggregating utility by R_i and for

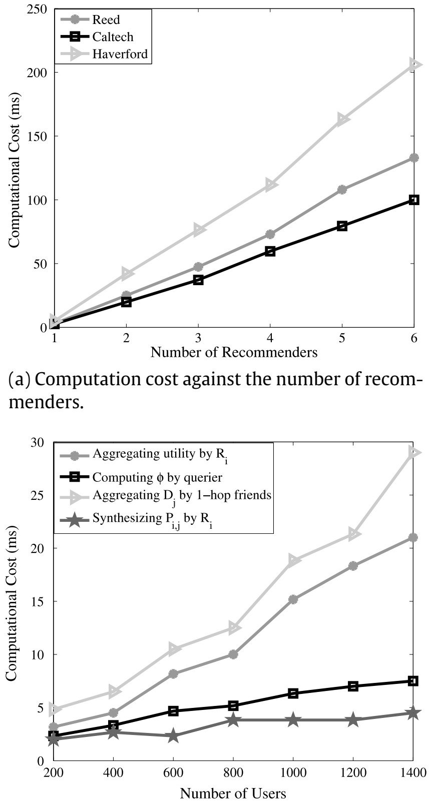


(a) Reachability and probability for reed.

(b) Reachability and probability for Caltech.

(c) Reachability and probability for Haverford.

Fig. 8. Reachability and distribution against the number of recommender.



(a) Computation cost against the number of recommenders.

(b) Computation cost for the four stages.

Fig. 9. The efficiency evaluation of ARMOR.

aggregating the social attributes by 1-hop friends clearly increase with the number of users in dataset. The reason for this is that the change in the number of users will lead to that of R_i's 1-hop friends. Therefore, R_i has to spend more time on aggregating the utilities when he has more 1-hop friends. Similarly, more 1-hop friends means that our ARMOR needs more time to aggregate the social attributes. However, the run time for computing utilities ratio and for synthesizing recommender results are less effected by the increasing of users and just increase a little slowly. The reason for that is querier u_0 and recommender R_i only need to perform a simple calculation to get the results. Hence, although more 1-hop friends join the recommendation, the impact on them is minimal.

Finally, we also discuss the efficiency of ARMOR along with the privacy-preserving recommendation approach in [9]. Our ARMOR only requires a maximum of 206 ms to recommend friends in the privacy-preserving manner when using the Haverford dataset. However, the mechanism in [9] required more than 700 ms to establish the friendships at the beginning and they needed more to complete the total process. Therefore, based on the above analysis, we are convinced that our ARMOR is sufficiently efficient to answer the requests of queriers.

7. Conclusion

The disclosure of user profiles and social attributes in social recommendation seriously threats user's personal privacy. In this paper, we presented a novel solution, namely ARMOR, to address the grand challenges in privacy-preserving friend recommendation in OSNs. In ARMOR, we utilized the OSN users' social attributes and trust relationships to achieve the friend recommendation in a decentralized manner. Then, to protect the privacy of users, we adopted a light-weight privacy-preserving protocol to random the trust relationships and social attributes and derived the recommender results by the secure kNN computations. Moreover, we evaluated the effectiveness and performance of ARMOR, the results of which indicate that ARMOR is an effective and efficient solution.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant Nos. U1405255, 61672413, 61672408, 61602537, 61602357, 61303221, U1509214), National High Technology Research and Development Program (863 Program) (Grant Nos. 2015AA016007), China Postdoctoral Science Foundation Funded Project (Grant No. 2016M592762), Shaanxi Science & Technology Coordination & Innovation Project (Grant No. 2016TZC-G-6-3), Shaanxi Provincial Natural Science Foundation (Grant Nos. 2015JQ6227, 2016JM6005), China 111 Project (Grant No. B16037), Beijing Municipal Social Science Foundation (Grant No. 16XCC023).

Appendix. Proof of Theorem 1

Before the proof, we divide the friend recommendation process into two phases: privacy-preserving utility aggregation (**Phase A**) and privacy-preserving similarity computation (**Phase B**). So we will proof the security of these two phases respectively.

Phase A Construct a simulator $S_{u_0}^1$ that can simulate a view indistinguishable from the real view of u_0, $V_{u_0}^{\pi 1}(T_{0,1}, r_{0,1}, \rho_1, \omega_T, \omega_r, \omega_\rho, k_{1,2,3,4}, p; s_1, \alpha_1, c_{1,2,3,4}; U_{0,\mathcal{F}_{i,j}}^R)$. Here, $S_{u_0}^1$ conducts the following:

1. Pick random numbers $\widetilde{s_1}, \widetilde{\alpha_1}, \widetilde{c_{1,2,3,4}}$.
2. Generate m random numbers: $U_{0,\mathcal{F}_{i,j}}^R, j \in [1, m]$.
3. Output: $(T_{0,1}, r_{0,1}, \rho_1, \omega_T, \omega_r, \omega_\rho, k_{1,2,3,4}, p; \widetilde{s_1}, \widetilde{\alpha_1}, \widetilde{c_{1,2,3,4}}; \widetilde{U_{0,\mathcal{F}_{i,j}}^R})$.

We define the following hybrids:

- $H_0 = V_{u_0}^{\pi 1}(T_{0,1}, r_{0,1}, \rho_1, \omega_T, \omega_r, \omega_\rho, k_{1,2,3,4}, p)$
- $H_1 = (T_{0,1}, r_{0,1}, \rho_1, \omega_T, \omega_r, \omega_\rho, k_{1,2,3,4}, p; \widetilde{s_1}, \widetilde{\alpha_1}, \widetilde{c_{1,2,3,4}}; U_{0,\mathcal{F}_{i,j}}^R)$
- $H_2 = S_{u_0}^1(T_{0,1}, r_{0,1}, \rho_1, \omega_T, \omega_r, \omega_\rho, k_{1,2,3,4}, p; U_{0,\mathcal{F}_{i,j}}^R)$

Given that $\{\widetilde{s_1}, \widetilde{\alpha_1}, \widetilde{c_{1,2,3,4}}\}$ are generated according to the same distribution as $\{s_1, \alpha_1, c_{1,2,3,4}\}$, the hybrids $H_0 \overset{c}{\equiv} H_1$. Similarly, the distribution of $U_{0,\mathcal{F}_{i,j}}^R$ and $U_{0,\mathcal{F}_{i,j}}^R$ are exactly the same, and the light-weight privacy-preserving protocol is secure, thus $H_1 \overset{c}{\equiv} H_2$. Hence, we show that $V_{u_0}^{\pi 1} \overset{c}{\equiv} S_{u_0}^1$.

Then, a simulator $S_{R_i}^1$ is constructed that can simulate a view indistinguishable from R_i's real view $V_{R_i}^{\pi 1}(T_{i,j}, r_{i,j}, \rho_{\mathcal{F}_{i,j}}; \mathcal{T}'_{i,h}, \mathcal{R}'_{i,h}, \Re'_{i,h}, \alpha_1)$. Here, $S_{R_i}^1$ conducts the following:

1. Generate three random numbers: $\widetilde{\mathcal{T}'_{i,h}}, \widetilde{\mathcal{R}'_{i,h}}, \widetilde{\Re'_{i,h}}$.
2. Generate a random parameter $\widetilde{\alpha_1}$.
3. Output:$(T_{i,j}, r_{i,j}, \rho_{\mathcal{F}_{i,j}}; \widetilde{\mathcal{T}'_{i,h}}, \widetilde{\mathcal{R}'_{i,h}}, \widetilde{\Re'_{i,h}}, \widetilde{\alpha_1}), j \in [1, m]$.

We define the hybrids $H_0 = V_{R_i}^{\pi 1}(T_{i,j}, r_{i,j}, \rho_{\mathcal{F}_{i,j}}; \mathcal{T}'_{i,h}, \mathcal{R}'_{i,h}, \Re'_{i,h}, \alpha_1)$ and $H_1 = (T_{i,j}, r_{i,j}, \rho_{\mathcal{F}_{i,j}}; \widetilde{\mathcal{T}'_{i,h}}, \widetilde{\mathcal{R}'_{i,h}}, \widetilde{\Re'_{i,h}}, \widetilde{\alpha_1})$.

Given that $\{\widetilde{\mathcal{T}'_{i,h}}, \widetilde{\mathcal{R}'_{i,h}}, \widetilde{\Re'_{i,h}}, \widetilde{\alpha_1}\}$ are generated according to the same distribution as$\{\mathcal{T}'_{i,h}, \mathcal{R}'_{i,h}, \Re'_{i,h}, \alpha_1\}$, and that the light-weight privacy-preserving protocol is secure, the hybrids $H_0 \overset{c}{\equiv} H_1$. Hence, $V_{R_i}^{\pi 1} \overset{c}{\equiv} S_{R_i}^1$.

Phase B Construct a simulator $S_{u_0}^2$ that simulate a view indistinguishable from u_0's real view $V_{u_0}^{\pi 2}(Q, \phi(U_i), k_{1,2,3,4}, p; s_2, \alpha_2, e_v)$, $v \in [1, n+2]$. Here, $S_{u_0}^2$ conducts the following:

1. Pick random numbers $\widetilde{s_2}$ and $\widetilde{\alpha_2}$.
2. Generate $n+2$ random numbers $\widetilde{e_v}, v \in [1, n+2]$.
3. Output: $(Q, \phi(U_i), k_{1,2,3,4}, p; \widetilde{s_2}, \widetilde{\alpha_2}, \widetilde{e_v}), v \in [1, n+2]$.

We define the following hybrids:

- $H_0 = V_{u_0}^{\pi 2}(Q, \phi(U_i), k_{1,2,3,4}, p)$
- $H_1 = (Q, \phi(U_i), k_{1,2,3,4}, p; \widetilde{s_2}, \widetilde{\alpha_2}, \widetilde{e_v}), v \in [1, n+2]$
- $H_2 = S_{u_0}^2(Q, \phi(U_i), k_{1,2,3,4}, p; F)$

Given that $\{\widetilde{s_2}, \widetilde{\alpha_2}, \widetilde{e_v}\}, v \in [1, n+2]$, are generated with the same distribution as $\{s_2, \alpha_2, e_v\}, v \in [1, n+2]$, the hybrids $H_0 \overset{c}{\equiv} H_1$. Because the security of the light-weight privacy-preserving protocol, we also have $H_1 \overset{c}{\equiv} H_2$. Thus, $V_{u_0}^{\pi 2} \overset{c}{\equiv} S_{u_0}^2$.

Next, we construct a simulator $S_{R_i}^2$ that simulate a view indistinguishable from R_i's real view $V_{R_i}^{\pi 2}(F, \alpha_2, \mathcal{P}_i; s_2^{-1}\phi(U_i), D_j), j \in [1, m]$. Here, $S_{R_i}^2$ conducts the following:

1. Generate a random number $\widetilde{s_2^{-1}\phi(U_i)}$.
2. Generate m random number $\widetilde{D_j}, j \in [1, m]$.
3. Run the protocol and output: $(F, \alpha_2, \mathcal{P}_i; \widetilde{s_2^{-1}\phi(U_i)}, \widetilde{D_j}), j \in [1, m]$.

We define the following hybrids:

- $H_0 = V_{R_i}^{\pi 2}(F, \alpha_2, \mathcal{P}_i)$
- $H_1 = S_{R_i}^2(F, \alpha_2, \mathcal{P}_i; \widetilde{s_2^{-1}\phi(U_i)}, \widetilde{D_j}), j \in [1, m]$

Given that $\widetilde{s_2^{-1}\phi(U_i)}$ and $\widetilde{D_j}, j \in [1, m]$ are generated as the distribution of $s_2^{-1}\phi(U_i)$ and $D_j, j \in [1, m]$, and that the light-weight privacy-preserving protocol is secure, we have $H_0 \overset{c}{\equiv} H_1$. Thus, $V_{R_i}^{\pi 2} \overset{c}{\equiv} S_{R_i}^2$.

Finally, we construct a simulator $S_{\mathcal{F}_{i,j}}^2$ that simulate a view indistinguishable from $\mathcal{F}_{i,j}$'s real view $V_{\mathcal{F}_{i,j}}^{\pi 2}(A; w_{j,v}; F, \alpha_2), v \in [1, n+2]$. Here, $S_{\mathcal{F}_{i,j}}^2$ conducts the following:

1. Generate $n+2$ random numbers $\widetilde{w_{j,v}}, v \in [1, n+2]$.
2. Generate a $n+2$ random vector $\widetilde{F}$ and a random number $\widetilde{\alpha_2}$.
3. Run the protocol with $\widetilde{w_{j,v}}, \widetilde{F}$ and $\widetilde{\alpha_2}, v \in [1, n+2]$.
4. Output: $(A; \widetilde{w_{j,v}}; \widetilde{F}, \widetilde{\alpha_2}), v \in [1, n+2]$.

Then, the hybrids are defined as follows:

- $H_0 = V_{\mathcal{F}_{i,j}}^{\pi 2}(A)$
- $H_1 = (A; \widetilde{w_{j,v}}), v \in [1, n+2]$
- $H_2 = S_{\mathcal{F}_{i,j}}^2(A; \widetilde{w_{j,v}}; \widetilde{F}, \widetilde{\alpha_2}), v \in [1, n+2]$

Given $\widetilde{w_{j,v}}, v \in [1, n+2]$, generated as the distribution of $w_{j,v}, v \in 1, n+2$, we have $H_0 \overset{c}{\equiv} H_1$. Then, through the security of light-weight privacy-preserving protocol and the same distribution of $\{\widetilde{F}, \widetilde{\alpha_2}\}$ with $\{F, \alpha_2\}$, we have $H_1 \overset{c}{\equiv} H_2$. Hence, $V_{\mathcal{F}_{i,j}}^{\pi 2} \overset{c}{\equiv} S_{\mathcal{F}_{i,j}}^2$.

To summarize, no adversary can distinguish the simulators' views from their own real ones. So we can ensure that the QoT attributes between two 1-hop friends cannot be revealed to others in Phase A and the social attributes are not exposed in Phase B. Therefore, we claim that no adversary can obtain the QoT attributes and social attributes in the friend recommendation and our ARMOR is secure. □

References

[1] N. Kökciyan, P. Yolum, PriGuard: A semantic approach to detect privacy violations in online social networks, IEEE Transactions on Knowledge and Data Engineering 28 (10) (2016) 2724–2737.

[2] Q. Tang, J. Wang, Privacy-preserving friendship-based recommender systems, IEEE Transactions on Dependable and Secure Computing PP (2016) 1–16. http://dx.doi.org/10.1109/TDSC.2016.2631533.

[3] L. Backstrom, D.P. Huttenlocher, J.M. Kleinberg, X. Lan, Group formation in large social networks: membership, growth, and evolution, in: Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 2006, pp. 44–54.

[4] A. Sharma, M. Gemici, D. Cosley, Friends, strangers, and the value of ego networks for recommendation, in: Proceedings of the Seventh International Conference on Weblogs and Social Media, Cambridge, Massachusetts, USA, 2013, pp. 721–724.

[5] M. Mitchell, Complex systems: Network thinking, Artificial Intelligence 170 (18) (2006) 1194–1212.

[6] S. Huang, J. Zhang, L. Wang, X. Hua, Social friend recommendation based on multiple network correlation, IEEE Transactions on Multimedia 18 (2) (2016) 287–299.

[7] M. von Arb, M. Bader, M. Kuhn, R. Wattenhofer, VENETA: Serverless Friend-of-Friend Detection in Mobile Social Networking, in: IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, Avignon, France, 2008, pp. 184–189.

[8] J.H. Abawajy, M.I.H. Ninggal, T. Herawan, Privacy preserving social network data publication, IEEE Communications Surveys and Tutorials 18 (3) (2016) 1974–1997.

[9] L. Guo, C. Zhang, Y. Fang, A trust-based privacy-preserving friend recommendation scheme for online social networks, IEEE Transactions on Dependable and Secure Computing 12 (4) (2015) 413–427.

[10] X. Yi, E. Bertino, F. Rao, A. Bouguettaya, Practical privacy-preserving user profile matching in social networks, in: 32nd IEEE International Conference on Data Engineering, Helsinki, Finland, 2016, pp. 373–384.

[11] R. Lu, H. Zhu, X. Liu, J.K. Liu, J. Shao, Toward efficient and privacy-preserving computing in big data era, IEEE Network 28 (4) (2014) 46–50.

[12] L. Jin, Y. Chen, T. Wang, P. Hui, A.V. Vasilakos, Understanding user behavior in online social networks: A survey, IEEE Communications Magazine 51 (9) (2013) 144–150.

[13] Y. Seo, Y. Kim, E. Lee, D. Baik, Personalized recommender system based on friendship strength in social network services, Expert Systems with Applications 69 (2017) 135–148.

[14] C. Xu, M. Zhou, F. Chen, A. Zhou, Detecting user preference on microblog, in: Proceedings of the 18th International Conference on Database Systems for Advanced Applications, Wuhan, China, Part II, 2013, pp. 219–227.

[15] S.S. Rodríguez, R.P.D. Redondo, A.F. Vilas, Y. Blanco-Fernández, J.J. Pazos-Arias, A tie strength based model to socially-enhance applications and its enabling implementation: Mysocialsphere, Expert Systems with Applications 41 (5) (2014) 2582–2594.

[16] Y. Ma, Z. Yu, J. Ding, A method of user recommendation in social networks based on trust relationship and topic similarity, in: Proceedings of the Third National Conference on Social Media Processing, Beijing, China, 2014, pp. 240–251.

[17] M. Moricz, Y. Dosbayev, M. Berlyant, PYMK: friend recommendation at myspace, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Indianapolis, Indiana, USA, 2010, pp. 999–1002.

[18] Y. Sun, C.C. Chen, A novel social event recommendation method based on social and collaborative friendships, in: Proceedings of the 5th International Conference on Social Informatics, SocInfo, Kyoto, Japan, 2013, pp. 109–118.

[19] E.M. Daly, M. Haahr, Social network analysis for information flow in disconnected delay-tolerant manets, IEEE Transactions on Mobile Computing 8 (5) (2009) 606–621.

[20] H. Yin, B. Cui, L. Chen, Z. Hu, Z. Huang, A temporal context-aware model for user behavior modeling in social media systems, in: International Conference on Management of Data, SIGMOD, Snowbird, UT, USA, 2014, pp. 1543–1554.

[21] W. Chen, S. Fong, Social network collaborative filtering framework and online trust factors: A case study on Facebook, in: Fifth IEEE International Conference on Digital Information Management, Lakehead University, Thunder Bay, Canada, 2010, pp. 266–273.

[22] Z. Wang, J. Liao, Q. Cao, H. Qi, Z. Wang, Friendbook: A semantic-based friend recommendation system for social networks, IEEE Transactions on Mobile Computing 14 (3) (2015) 538–551.

[23] C.C. Chen, S. Shih, M. Lee, Who should you follow? Combining learning to rank with social influence for informative friend recommendation, Decision Support Systems 90 (2016) 33–45.

[24] Q. Jiang, S. Zeadally, J. Ma, D. He, Lightweight three-factor authentication and key agreement protocol for internet-integrated wireless sensor networks, IEEE Access 5 (2017) 3376–3392.

[25] Q. Jiang, J. Ma, C. Yang, X. Ma, J. Shen, S.A. Chaudhry, Efficient end-to-end authentication protocol for wearable health monitoring systems, Computers & Electrical Engineering (2017). http://dx.doi.org/10.1016/j.compeleceng.2017.03.016.

[26] J.D. Zhang, G. Ghinita, C.Y. Chow, Differentially private location recommendations in geosocial networks, in: IEEE 15th International Conference on Mobile Data Management, vol. 1, 2014, pp. 59–68.

[27] S. Gao, J. Ma, C. Sun, X. Li, Balancing trajectory privacy and data utility using a personalized anonymization model, Journal of Network and Computer Applications 38 (2014) 125–134.

[28] S. Gao, J. Ma, W. Shi, G. Zhan, C. Sun, Trpf: A trajectory privacy-preserving framework for participatory sensing, IEEE Transactions on Information Forensics and Security 8 (6) (2013) 874–887.

[29] Z. Fu, F. Huang, X. Sun, A. Vasilakos, C.-N. Yang, Enabling semantic search based on conceptual graphs over encrypted outsourced data, IEEE Transactions on Services Computing PP (2016) 1–11. http://dx.doi.org/10.1109/TSC.2016.2622697.

[30] Z. Fu, X. Sun, Q. Liu, L. Zhou, J. Shu, Achieving efficient cloud search services: Multi-keyword ranked search over encrypted cloud data supporting parallel computing, IEICE Transactions 98-B (1) (2015) 190–200.

[31] Z. Fu, X. Wu, C. Guan, X. Sun, K. Ren, Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement, IEEE Transactions on Information Forensics and Security 11 (12) (2016) 2706–2716.

[32] Z. Xia, X. Wang, X. Sun, Q. Wang, A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data, IEEE Transactions on Parallel and Distributed Systems 27 (2) (2016) 340–352.

[33] I. Kayes, A. Iamnitchi, A survey on privacy and security in online social networks, CoRR abs/1504.03342 (2015) 1–40.

[34] M. Fire, R. Goldschmidt, Y. Elovici, Online social networks: Threats and solutions, IEEE Communications Surveys and Tutorials 16 (4) (2014) 2019–2036.

[35] N. Kökciyan, Privacy management in agent-based social networks, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, 2016, pp. 4299–4300.

[36] G.P. Cheek, M. Shehab, Privacy management for online social networks, in: Proceedings of the 21st World Wide Web Conference, WWW, Lyon, France, (Companion Volume), 2012, pp. 475–476.

[37] D.A. Albertini, B. Carminati, E. Ferrari, Privacy settings recommender for online social network, in: 2nd IEEE International Conference on Collaboration and Internet Computing, CIC, Pittsburgh, PA, USA, 2016, pp. 514–521.

[38] N. Polatidis, C.K. Georgiadis, E. Pimenidis, H. Mouratidis, Privacy-preserving collaborative recommendations based on random perturbations, Expert Systems with Applications 71 (2017) 18–25.

[39] D. Li, Q. Lv, L. Shang, N. Gu, Efficient privacy-preserving content recommendation for online social communities, Neurocomputing 219 (2017) 440–454.

[40] S. Puglisi, J. Parra-Arnau, J. Forné, D. Rebollo-Monedero, On content-based recommendation and user privacy in social-tagging systems, Computer Standards & Interfaces 41 (2015) 17–27.

[41] S. Liu, A. Liu, G. Liu, Z. Li, J. Xu, P. Zhao, L. Zhao, A secure and efficient framework for privacy preserving social recommendation, in: Proceedings of 17th Asia–PacificWeb Conference on Web Technologies and Applications, Guangzhou, China, 2015, pp. 781–792.

[42] B.K. Samanthula, L. Cen, W. Jiang, L. Si, Privacy-preserving and efficient friend recommendation in online social networks, Transactions on Data Privacy 8 (2) (2015) 141–171.

[43] L. Chen, P. Zhu, Preserving the privacy of social recommendation with a differentially private approach, in: IEEE International Conference on Smart City/SocialCom/SustainCom, Chengdu, China, 2015, pp. 780–785.

[44] F. Abbas, U. Rajput, H. Oh, PRISM: Privacy-aware interest sharing and matching in mobile social networks, IEEE Access 4 (2016) 2594–2603.

[45] L. Zhang, X. Li, K. Liu, T. Jung, Y. Liu, Message in a sealed bottle: Privacy preserving friending in mobile social networks, IEEE Transactions on Mobile Computing 14 (9) (2015) 1888–1902.

[46] B.K. Samanthula, W. Jiang, Structural and message based private friend recommendation, in: International Conference on Advances in Social Networks Analysis and Mining, Istanbul, Turkey, 2012, pp. 684–690.

[47] B.K. Samanthula, W. Jiang, Interest-driven private friend recommendation, Knowledge and Information Systems 42 (3) (2015) 663–687.

[48] Y. Wang, B. Zheng, Preserving privacy in social networks against connection fingerprint attacks, in: 31st IEEE International Conference on Data Engineering, ICDE, Seoul, South Korea, 2015, pp. 54–65.

[49] R. Lu, X. Lin, X. Liang, X. Shen, A secure handshake scheme with symptoms-matching for mhealthcare social network, Mobile Networks and Applications 16 (6) (2011) 683–694.

[50] X. Ma, H. Li, J. Ma, Q. Jiang, S. Gao, N. Xi, D. Lu, Applet: A privacy-preserving framework for location-aware recommender system, Science China Information Sciences 60 (9) (2017) 092101.

[51] G. Liu, Y. Wang, M.A. Orgun, Quality of trust for social trust path selection in complex social networks, in: 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), vols. 1–3, Toronto, Canada, 2010, pp. 1575–1576.

[52] G. Liu, Y. Wang, M.A. Orgun, E. Lim, Finding the optimal social trust path for the selection of trustworthy service providers in complex social networks, IEEE Transactions on Services Computing 6 (2) (2013) 152–167.

[53] J. Golbeck, J.A. Hendler, Inferring binary trust relationships in web-based social networks, ACM Transactions on Internet Technology 6 (4) (2006) 497–529.

[54] F.E. Walter, S. Battiston, F. Schweitzer, A model of a trust-based recommendation system on a social network, Autonomous Agents and Multi-Agent Systems 16 (1) (2008) 57–74.

[55] Y. Hu, Y. Fan, Z. Di, Orientation in social networks, arXiv Preprint, arXiv:0902.3329, (2009) 1–10.

[56] R.K. Merton, The role-set: Problems in sociological theory, The British Journal of Sociology 8 (2) (1957) 106–120.

[57] J.F. Kenney, E.S. Keeping, Mathematics of Statistics-Part One, D. Van Nostrand Company Inc, Toronto, New York, London, 1954.

[58] O. Goldreich, The Foundations of Cryptography, Basic Applications, vol. 2, Cambridge University Press, 2004.

[59] R. Bost, R.A. Popa, S. Tu, S. Goldwasser, Machine learning classification over encrypted data, in: 22nd Annual Network and Distributed System Security Symposium, NDSS, San Diego, California, USA, 2015, pp. 1–14.

[60] A.L. Traud, P.J. Mucha, M.A. Porter, Social structure of facebook networks, Physica A: Statistical Mechanics and its Applications 391 (16) (2012) 4165–4180.

[61] A. Barabási, Linked: How everything is connected to everything else and what it means for business, science, and everyday life, Plume, 2003.

[62] J. Leskovec, E. Horvitz, Planetary-scale views on a large instant-messaging network, in: Proceedings of the 17th International Conference on World Wide Web, WWW, Beijing, China, 2008, pp. 915–924.

Xindi Ma received the B.S. degree in school of computer science and technology from Xidian University, China in 2013. He is currently working toward the Ph.D. degree at the School of Cyber Engineering, Xidian University, China. His current research interests include database security, location-based service and recommender system with focus on security and privacy issues.

Jianfeng Ma received the B.S. degree in computer science from Shaanxi Normal University in 1982, and M.S. degree in computer science from Xidian University in 1992, and the Ph.D. degree in computer science from Xidian University in 1995. Currently he is a Professor at School of Computer Science and Technology, Xidian University. His research interests include information security, cryptography, and network security.

Hui Li received the B.Eng. from Harbin Institute of Technology in 2005 and Ph.D. degree from Nanyang Technological University, Singapore in 2012, respectively. He is an Associate Professor in School of Cyber Engineering, Xidian University, China. His research interests include data mining, knowledge management and discovery, privacy-preserving query and analysis in big data.

Qi Jiang received the B.S. degree in Computer Science from Shaanxi Normal University in 2005 and Ph.D. degree in Computer Science from Xidian University in 2011. He is now an associate professor at School of Cyber Engineering, Xidian University. His research interests include security protocols and wireless network security, cloud security, etc.

Sheng Gao is an Assistant Professor in the School of Information at Central University of Finance and Economics. He received the B.S. degree in information and computation science from Xi'an University of Posts and Telecommunications, in 2009, and the Ph.D. degree in computer science and technology from Xidian University, in 2014. His current research interests include finance information security and privacy computing.