Security and Communication Networks

Social Security and Privacy for Social IoT

Lead Guest Editor: Laura Pierucci
Guest Editors: Houbing Song and Hua Wang

Social Security and Privacy for Social IoT

Security and Communication Networks

Social Security and Privacy for Social IoT

Lead Guest Editor: Laura Pierucci
Guest Editors: Houbing Song and Hua Wang

Contents

Hindawi
Security and Communication Networks
Volume 2019, Article ID 9108759, 14 pages
https://doi.org/10.1155/2019/9108759

Research Article

Privacy Protection of Social Networks Based on Classified Attribute Encryption

Lin Zhang,[1,2,3] **Li Li**,[1] **Eric Medwedeff**,[2] **Haiping Huang**,[1,3] **Xiong Fu**,[1,3] **and Ruchuan Wang**[1,3]

[1]*College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China*
[2]*Department of Computer Science, San Diego State University, San Diego, CA 92182, USA*
[3]*Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210003, China*

Correspondence should be addressed to Lin Zhang; zhangl@njupt.edu.cn

Received 12 February 2019; Revised 12 April 2019; Accepted 25 August 2019; Published 26 September 2019

Guest Editor: Hua Wang

With the rapid development of social networks, privacy has also attracted attention. Based on this problem, a privacy protection scheme for social networks based on classified attribute encryption (PPSSN) is proposed for the data owner and attribute management server to manage user permissions; the approach reduces data owner overhead and also avoids use of a property management server to limit access user collusion attacks. To balance the privacy and security of data publication, this scheme classifies users and designs access control for different users and different privileges. In addition, this paper also introduces a good friend data cache mechanism to improve and optimize the original scheme to reduce the cost of decryption. The efficiency and system overhead of the proposed scheme are compared and analyzed based on experiments. The experiments show that the proposed scheme improves query efficiency, reduces system cost, and enhances privacy security.

1. Introduction

In this era of rapid information technology and networking, people have greater access to data. With the convenience of communication, social networks have come into being and gradually became popular. At present, the definition of social networking is relatively mature; that is, social networking services are built [1] for social networking applications. Through social networking, we can communicate with friends by sending messages and sharing content. Because of its convenience, ease of operation, and so on, social networks have become more and more common, given the maturity of network technology. At the same time, with the rapid development of social networking, the problem of privacy has also attracted attention. Private data in social networks mainly include user identity information, user login information, user friend information, and data published on the social network platform. The root cause of privacy security in social networks is that the data owner's privacy data are separated from the direct physical control of the data owner when the data are transmitted on the social network platform. As a result, the data may be leaked, and thus users who have not been able to view the permission or even users who are malicious steal information to see the content of the data.

To date, lots of research attention has been given to security and privacy of network technology. Zhang et al. [2] established a theoretical framework for the study of eavesdropper-tolerance capability in a two-hop wireless network, where the cooperative jamming is adopted to ensure security defined by secrecy outage probability (SOP) and opportunistic relaying is adopted to guarantee reliability defined by transmission outage probability. To tackle the identified challenges, Wang et al. [3] summarized high-quality submissions for the special issue, the paper generalized that Lomotey et al. [4] proposed a provenance technique, which leverages the associative rules and lexical chaining methodologies to enable data traceability, as well as the detection of faulty data propagation, through the identification of propagation routes of data and object-to-

object communications. And Chen et al. [5] proposed a set of cryptographic techniques to protect medical IoT with respect to data transmission, storage and access, including a multipath asymmetric encryption fragment transmission mechanism, a distributed symmetric encryption cloud storage scheme, and a dynamic access control scheme. Huang et al. [6] proposed an access control mechanism based on hierarchical attribute-based encryption (ABE) scheme to preserve the confidentiality of collected data in transmission and at rest (i.e., stored at resource constrained IoT devices).

For the above privacy protection, encryption is a common solution. For sensitive data, ciphertext access control can be used to control user decryption privileges by encrypting data. Shamir [7] puts forward the concept of the identity-based system, and Boneh and Franklin [8] put forward the self-identity-based encryption scheme based on this. In the attribute-based scheme proposed by Sahai and Waters [9], the user's qualification certificate is expressed by the attribute set, and the operation is expressed by the formula of these attributes to solve the problem of decryption of the ciphertext by sharing the data. However, one drawback is that their initial structure is limited to a formula made up of thresholds.

Goyal et al. [10] then divided the property attribute-based encryption schemes into two categories: one is an encryption scheme based on the key strategy, and the other is an attribute encryption scheme based on the ciphertext strategy. In the encryption scheme based on the key strategy, the key is associated with the access control strategy, and in the cryptographic scheme based on the ciphertext strategy, the ciphertext is associated with the control strategy, and the key is associated with the attribute set. Bethencourt et al. [11] proposed an attribute encryption technology based on the ciphertext strategy (CP-ABE, ciphertext-polity attribute-based encryption) to associate the user's private key to a set of attributes, and the user ciphertext is associated with the access tree. When the user attribute set satisfies the access tree, the user can decrypt the data. The algorithm contains the decryption rules in the encryption algorithm, which greatly optimizes the frequent key distribution in ciphertext access. Waters [12] proposed a new method of the attribute-based encryption of the ciphertext policy that combines the linear secret-sharing scheme. In the framework, three structures are proposed to realize the specific and noninteractive assumptions of the general access structure of the cipher policy ABE system from the standard model.

Because most of the existing attribute-based schemes are mostly based on pairing-based operations, the size and decryption time of the ciphertext will increase with the expansion of the access scale, requiring huge computing costs. Aiming at this problem, Green et al. [13] put forward a secure outsourcing computing technology that converts most of the decoding work to the proxy cloud server. However, because the identity of the user is not always fixed in the system, any change in user identity means that the attribute will be replaced, revoked, or added, so the scheme has the problem of user flexibility in attribute revocation.

In the property revocable scheme proposed by Yu et al.[14], the agent can delegate the agent's re-encryption and key update to the proxy server by proxy re-encryption. When users access encrypted shared data, the proxy server re-encrypts them. Although the scheme has provided formal security proof for unauthorized users to use a semitrusted proxy server, it does not provide formal security proof for the revoked user.

In Naruse et al. [15] scheme, no user can decrypt ciphertext encrypted by a public key generated by authority. When users download encrypted shared data from cloud servers, the encrypted proxy of the cloud server is re-encrypted, and users can decrypt it. When the cancelled user downloads the encrypted shared data, the cloud server will not re-encrypt it. So, the revoked user cannot decrypt it; however, because the scheme only supports the "and" policy, the access strategy is limited.

In a one-to-many encryption mechanism, users with the same attributes will share the same decryption privileges. It is possible that the malicious users can divulge the decryption key or decryption privileges to others in the form of a decryption black box. To solve this problem, Liu et al. [16, 17] used the black box traceability of system's access to detect whether the user was leaking the decryption device's behavior. However, the problem of how to effectively revoke permissions after malicious users and to design efficient black boxes in a short ciphertext-based system without sacrificing other performance remains unsolved.

In CP-ABE, according to the authorization center, it can be divided into two categories: the data center and authority center. In [18, 19], the idea is that the data owner is the center, and the generation and distribution of the private key is completed by the data owner when the data owner establishes contact with the user. Liang et al. [20] proposed that the authority of the attribute authority is the center, which is responsible for the work of generating and distributing the private key by the authority, and the data owner is only responsible for the encryption of the data. The two ideas are to ensure that only users who satisfy the attribute policy can decrypt the published data. However, the cost of generating the private key is linearly related to the corresponding set of attributes, so the data-owner center leads to bottlenecks in data-owner computing when access to the user is greatly increased; the authority-centered scheme has the effect of reducing the data's main cost; the authority will bring the risk of data leakage at the same time so that the privacy of the network data cannot be guaranteed.

Lv et al. [1] have made new improvements on the basis of attribute-based encryption and designed the attribute-based encryption algorithm with trapdoor to achieve efficient cancellation. However, it is not satisfied with the refine service.

On the basis of [1], in order to adapt to diverse user needs, Lin et al. [21] proposed a privacy protection scheme based on attribute encryption to support more precise services. It provides three modes to perform a simple classification of friends, make different attribute strategies for them, and get different precise ciphertexts from different levels. Although the scheme is capable of resisting social

network friend attacks, it is necessary to encrypt three different operating attributes according to the difference in key information, which increases the amount of computation and increases the cost. Moreover, this study only focuses on location information encryption and is not widely applicable.

In view of the above problems, this paper proposes a social network privacy protection scheme based on attribute encryption (PPSSN) and designs the following improvements:

(1) The access authority of visitors is limited by data owners and attribute management servers, which not only reduces the overhead of the data owner but also handles part of the work by the attribute management server, thus avoiding conspiracy attacks between the attribute management server and the nonprivileged access user.

(2) To tradeoff the usability of users' published data and of information privacy protection, according to the data owner classification of the relationship between all users on a social network, the mechanism of different close relationships accessing data with different degrees of accuracy is adopted, and the access control of different rights is realized.

(3) To reduce the decryption cost of the system, the friends data buffer mechanism is introduced to improve and optimize it.

The organizational structure of this paper is as follows: Section 2 introduces the related concepts; Section 3 describes the overall design of the scheme in detail, including the system model, the algorithm design, and the scheme description; Section 4 proves the security of the scheme; and Section 5 compares the feasibility and efficiency analysis of the case with the experiment.

2. Related Works

2.1. Bilinear Mapping

Definition 1 (bilinear mapping [8]). For the multiplicative groups G_1 and G_2, for which the rank is a large prime number p and g is a generator of G_1. The bilinear map e: $G_1 \times G_1 \longrightarrow G_2$ has the following properties:

Bilinear: for any a, $b \in Zp$, for $g \in G_1$, $e(g_a, g_b) = e(g, g)ab$

Non degeneracy: given $g \in G_1$, make $e(g, g) \neq 1$

Computability: for any $u, v \in G_1$, $e(u, v)$ can be effectively calculated

Definition 2 (access structure [11]). Let us set up $P = \{P_1, P_2, \ldots, P_n\}$ is a collection of all user attributes, and each user-associated attribute set is $A \subseteq P = \{P_1, P_2, \ldots, P_n\}$, then the access structure A is the nonempty subset of the attribute set P. The access structure A represents an attribute judgment condition, and the set of elements in the A is called an authorization set, and the set not made up of elements in the A is called an unauthorized set, and only the authorized user's key can decrypt the corresponding file.

2.2. Access Structure and Access Tree. The access tree is used in this article to describe the access structure. The access tree T is used to describe an access control strategy, which is a nonempty subset of the complete set of attributes P, T represents an attribute judgment condition, the leaf node of T represents an attribute, and a non-leaf node represents a relational function, in which the relation function is an "and", "or", "n of m" threshold. The set of attributes in T is authorized and those not in T is not authorized. Only the authorized user's key can decrypt the file. In an attribute-based encryption structure, the access control strategy is represented by the access structure tree, which can be decrypted only when the attribute set associated with the user's private key meets the access structure tree [1].

The schematic diagram of the access tree structure is shown in Figure 1.

2.3. Security Hypothesis. In this scheme, it is assumed that the attribute management system (AMS) and the data owner (DO) server are not fully trusted (honest but curious), that the AMS and DO servers will execute the corresponding program not to disclose the information actively but may try to obtain more information such as access rights and the exact data content of the distribution, resulting in some information being used by malicious users or other external attackers. It is assumed that the additional information obtained by the AMS and DO server is user-related identity information, permission to file, and the content of published file data. In addition, this paper assumes that the information transmission channel between the DO and AMS and between the DO and DO server is secure.

Because this scheme is designed for access control for general sensitive data, not top secret information, the AMS will choose the correct execution program, so the security hypothesis of this article is reasonable.

3. Model and Algorithm Design

3.1. System Model. The concepts in trust valuation and roles of nodes are as follows:

As shown in Figure 2, the social network privacy protection system (PPSSN) model contains the data owner (DO), the DO sever, the attribute management sever (AMS), the visitors, and the social networking service platform (SNP). Among them, the function of the DO is to classify and generate a friend list according to the intimate degree of the friends, determine the attribute encryption strategy, and generate and distribute the main private key of the data. The DO server encrypts and stores the friends list of DO. When the user requests access to data, it queries the encrypted list and returns the friendship data, which can share the DO overhead, but does not affect the security of the data. After receiving the user's access request, the AMS applies for a friend relationship to the DO, judges the friend's

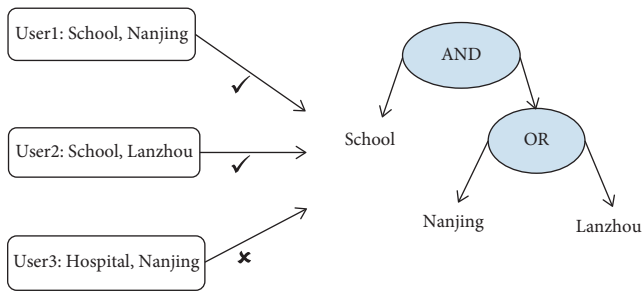


Figure 1: A schematic diagram of the structure of an access tree.

relationship through the data returned by the DO, and distributes the private key to the visitor. The social networking service platform (SNP) is responsible for user data publishing. The visitor V is the user on the social networking platform. When checking the publishing data of the DO, they need the corresponding access authority to access the data on the SNP.

When the visitor requests access to data, it needs to first apply to the DO through authentication of the AMS and DO and next receive the private key to access the SNP data; different identities will have different access rights. Close friends of the DO can get the first encryption parameters and can access the original accurate data of the DO, and the general friends of the DO can only access the published data that are encrypted after one time. According to this, users of the non-DO's friends can only access data that were last posted on the SNP, and illegal users identified by the AMS as social networks cannot access any DO published data.

3.2. Algorithm Design. This section introduces some core algorithm functions in the process of file re-encryption. On the basis of [8], the revocation is improved and set up to support the user level. The main functions are: the Grek function used to generate the re-cryptographic key, the Fre function for the re-encrypted operation of the file, and the CoA function to update the private key of the user.

3.2.1. Grek(): Generates a Re-Cryptographic Key. The main ideas of the algorithm are as follows: first, input the tree's leaf node attribute set, generate a random number, and recalculate the new private key of leaf node, then generate a new key, that is, encryption key, and finally replace the new key and the related parameters.

The basic description of the algorithm is shown in Algorithm 1.

3.2.2. Fre(): File Re-Encryption. The main ideas of the algorithm are as follows: on the basis of the original ciphertext, according to the new key generated by the function Grek (), the structure tree and the original ciphertext, recalculate the ciphertext and update the relevant parameters to realize the re-encryption of the file.

The algorithm flow is shown in Algorithm 2.

3.2.3. CoA(): The Replacement of the Permissions. The main ideas of the algorithm are as follows: according to the corresponding input, the AMS is responsible for revocation of authority and updating the private key of the unrevoked user at the same time, so that revoke the access rights of the unauthorized users and avoid the unauthorized access.

The algorithm flow is shown in the following Algorithm 3.

4. Scheme Design

4.1. Initialization

4.1.1. System Initialization. The DO completes the system's key initialization and sets related parameters.

Setup() $\longrightarrow$ OP, SMK: Define a bilinear mapping e: $G_0 \times G_0 \longrightarrow G_T$, where G_0 is a bilinear group, and it is constructed by P, whose rank is prime, and generator g. Define attribute space $A = \{A_1, A_2, \ldots, A_n\}$, for any property $A_i \in A\,(1 \le i \le n)$ randomly selected $x_{A_i} \in Z_p$ (p and cyclic group), randomly generated $\alpha, \beta \in Z_p$.

Published parameter $\mathrm{OP} = \{G_0, g, g^{\beta}, g^{x_{A_i}}, e(g,g)^{\alpha}\}$

Generate system master key $\mathrm{SMK} = \{\beta, g^{\alpha}, \{x_{A_i}\}\}$

The DO runs Setup(), generating public parameter OP and system master key SMK.

4.1.2. Friends List Initialization. Set up a list of friends in the DO server. The statistics and display information in the friend list are serial number (SN), access user ID, friend relationship parameter (FR), and verification parameter (VP).

The initial list shows close friends and ordinary friends identified by the DO. If you are close friends, the friend relationship parameter is 1; if you are an ordinary friend, you can get a friend relationship parameter (FR) to 0.

The BLori is defined as a list of original friends. In the BLori, for each record, the friend list has a serial number (SN), a user ID, a friend relationship parameter (FR), and a validation parameter (VP).

Among them, $\mathrm{BL_{ori}} = \{\mathrm{Rl}[1], \mathrm{Rl}[2], \ldots, \mathrm{Rl}[n]\}$, where Rl $[i]$ represents the record of article i in the $\mathrm{BL_{ori}}$ $(1 \le i \le n)$.

4.1.3. Friend List Encryption. Encrypted $\mathrm{BL_{ori}}$ to generate $\mathrm{BL_{en}}$, which is an encrypted friends list.

$\mathrm{BL_{en}} = \{\mathrm{Rl}'[1], \mathrm{Rl}'[2], \ldots, \mathrm{Rl}'[n]\}$, and Rl$'$ $s[i]$ represents the record of article i in the friend list $\mathrm{BL_{en}}$ $(1 \le i \le n)$; the function of the $\mathrm{BL_{en}}$ list is that the DO server does not understand the correspondence between the encrypted list of friends and the record of the original friends list and prevents the DO server from leaking the friend list information.

Let $\mathrm{BL_{ori}}$ and $\mathrm{BL_{en}}$ be the matrices of $\{0,1\}^{n*1}$, where n is the number of lists recorded and l is the length of the list record (unit bit); that is, an arbitrary record i is a vector as $\{0, 1\}^{l}(1 \le i \le n)$, $\mathrm{Rl}[i] = (b_{i1}, b_{i2}, \ldots, \mathrm{b}_{im})$, $\mathrm{Rl}'[i] = (b'_{i1}, b'_{i2}, \ldots, b'_{im})$ $(m = 1/32 + 1)$, Rl$[i]$ and Rl$'[i]$ are divided into 32 bit blocks, $\mathrm{BL_{ori}}$ and $\mathrm{BL_{en}}$ can be expressed as

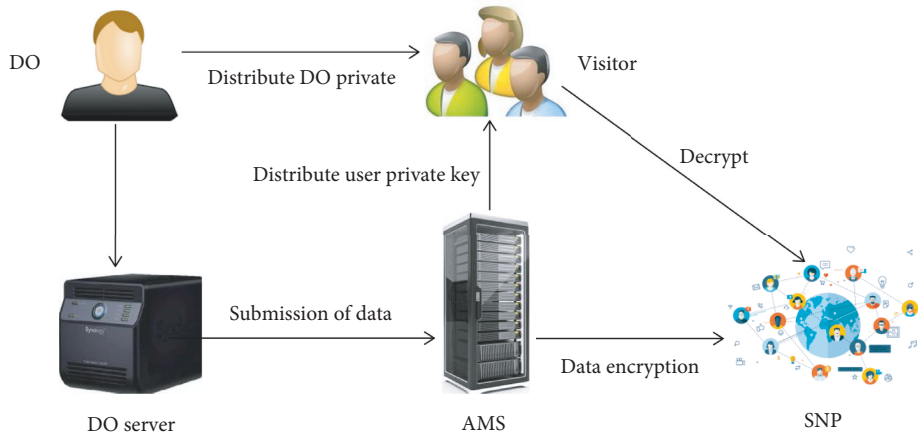


Figure 2: System model.

```
Input: the set of leaf node attributes of the access tree
Output: new key
//Generate a new key for each leaf node
(1) if (i ∉ URL)
        exit;
(2) else
(3)     take any s' ∈ Z;
(4)     for z ∈ Z, generate a new set of leaf node attributes q'_z(0)
(5)     compute new private key of leaf node;
(6)     take any x'_i ∈ Z_p;
(7)     calculate the re-encryption key according to the new access tree; //Generate a new key
(8)     replace the new key PPK';
(9)     replace a new signature;
(10)    OP'_x = OP'_{1,x} = {g^{βx}, OP'_{2,x} = g^{1/βx}} //Compute new public parameters
```

Algorithm 1: Generates re-encryption key.

```
Input: access structure tree, new encryption key, ciphertext CT
Output: CT'
(1) //Calculate the encrypted C'_1, C'.
(2) C'_1 = CT · e(g, g)^{αs'};
(3) C' = g^{βs'};
(4) C'^{(1)}_z = g^{q'z(0)};
(5) C'^{(2)}_z = g^{xiq'z(0)}_{att(z)};
(6) CT' = {T', C'_1, C, C'^{(1)}_z, C'^{(2)}_z};
```

Algorithm 2: File re-encryption.

```
Input: the user i, the new private key
(1) if (i ∈ URL)//Determine it in the user revocation list or not
(2)     exit;
(3) else
(4)     distribute the updated parameters to the user i;
(5)     for each attribute of the user i
(6)         calculate the corresponding key;
(7)         replace the property private key;
(8)         the user updates the private key;
(9)     end for
```

Algorithm 3: Permission replacement algorithm.

$$\mathrm{BL}_{\mathrm{ori}} = \begin{matrix} b_{11} & b_{12} & \dots & b_{13} \\ b_{21} & b_{22} & \dots & b_{23} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nm} \end{matrix}, \quad \mathrm{BL}_{\mathrm{en}} = \begin{matrix} b'_{11} & b'_{12} & \dots & b'_{13} \\ b'_{21} & b'_{22} & \dots & b'_{23} \\ \vdots & \vdots & & \vdots \\ b'_{n1} & b'_{n2} & \dots & b'_{nm} \end{matrix}, \tag{1}$$

where $\mathrm{BL}_{\mathrm{ori}}$ encryption generates the $\mathrm{BL}_{\mathrm{en}}$ process as shown in Figure 3.

The main idea of the encryption process is to first set up a mapping table that records the corresponding relationship between $\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$ in DO, select a random number as the seed, and generate a random number sequence; then, the process reads into the data of $\mathrm{BL}_{\mathrm{ori}}$, performs a modular operation of the corresponding bit with a pseudorandom sequence, saves the result into the register, rearranges the register according to the mapping table, and finally writes the register data to $\mathrm{BL}_{\mathrm{en}}$. To ensure the privacy of data in the encryption process, a random number of encrypted $\mathrm{BL}_{\mathrm{ori}}$ and some parameters are generated by DO. Among them, the encryption algorithm is based on the pseudorandom encryption algorithm of [18], and the specific encryption algorithm is shown in the following Algorithm 4.

4.1.4. Validation List V Initialization. A verification list V is built in the DO server, and it is used to record users who have been visited after the DO data release and are judged to be close friends by the DO.

The information shown in the verification list is: serial number (SN) and access to the user ID.

In addition, the information in the initial time list is empty except for the header, and users will update the verification list after the user's access is satisfied. When the user deletes the published content, the corresponding verification list page will be deleted.

Since only close friends who have already visited can join the list, the DO can first query Algorithm 5 when accessing the user's application data access. If it has previously been designated as a close friend, it can jump directly over the friend relationship judgment and go directly to the next step. It can reduce the query amount of subsequent queries and the cost.

4.1.5. Initialize the User Revocation List (URL). A user revocation list (URL) is set up in the AMS, which records all users who will be revoked by the DO. When an attribute is revoked, the user access to the file is revoked by referring to the URL table.

The information in the URL table includes the identification of the user ID and its cancelled files.

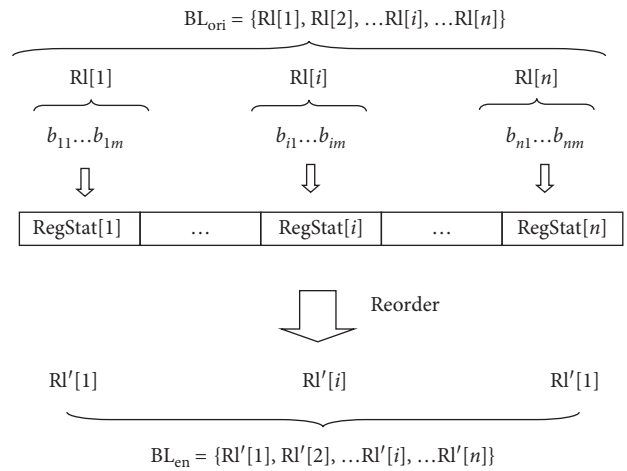


Figure 3: Encryption process.

4.2. DO Data Release. The DO will publish the content to be shared through the SNP platform. Before the data are released on the platform, the source data will be processed accordingly. This scheme describes the encryption processing of single file data, including the encryption of the original data by the DO, the generation and distribution of the private key, and the data release in the social network platform.

4.2.1. The DO Processes the Raw Data to Satisfy the Privacy Difference with Privacy Budget ε. For the stored source data di, the DO is processed first, and only the DO authenticated as a close friend can obtain the relevant parameters and ensure that it has the right to obtain the accurate data issued by the DO.

Step:

(1) Input di;

(2) Add noise to make it satisfy the differential privacy protection;

(3) Output: di′

4.2.2. Attribute-Based Encryption

Step 1: in the DO, define the mapping relationship between the attribute space A and the user, and the corresponding attribute set S is generated according to the user identity information. The KeyGen function is executed to generate the corresponding private key and send it to the user through the security channel.

PPK = KeyGen(S, SMK): The set of attributes associated with the user private key is defined as $S = \{S_1, S_2, \dots, S_k\}$ (S is a subset of A). For any $S_j \in S$ ($1 \le j \le k$), randomly select $y_{Sj} \in Z_{\mathrm{p}}$, randomly generates gamma $\gamma \in z_p$ and then generates a master key PPK. PPK = $\{D = g^{(\alpha+\gamma)/\beta}, \forall S_j \in S: D_{s_j}^{(1)} = g^{\gamma} g^{X_{ai}} g^{y_{sj}}, D_{s_j}^{(2)} = g^{y_{sj}}\}$.

```
Input: BL_ori
Output: BL_en
for (i = 1; n; i++)
    map[i] = random (1, n)//Return an integer type random number between 0 and 1;
    end for (i = i; n; i++)
for (i = i; n; i++)
    RegStat[0] = (rand() << 17) | (rand() << 3) | (rand()); //Generate random number store in the register;
    RegStat[0]take as seed, and generate pseudorandom number sequence PRS = (p_1, p_2, ..., p_n);
    for (j = 1; n; j++)
        RegStat[j] = b_ji ⊕ p_j;
        RegStat[map[t]] = RegSta[i]; //Rearrangement of registers by mapping table
    end for
    for (k = 1; n; k++)
        add RegStat[k] to the K record of BL_en.
    end for
end for
```

Algorithm 4: Encryption algorithm (En).

```
Input: ID′, R′
Output: R
i = ID′;
R = p′_i1, p′_i2, ..., p′_im + R′;
Return R
```

Algorithm 5: De algorithm.

After receiving the private key, the user i sends the parameter OP, the attribute set, and the signature to the AMS.

Step 2: The AMS verifies the signature sent by the user; if it is right, it will expose the OP.

Step 3: in the DO, define an access tree structure T'_{di} for the data di ready to be released;

Perform the attribute-based encryption algorithm encrypt function, input parameters (SMK, Di, Mdi′), encrypt plaintext data di′, and get ciphertext di″.

For access tree T_i, the node is x; the root node is R; the leaf node is z; the set of the leaf node is Z; and the pa (x) returns the parent node of the node x; num (x) returns the number of the node x, and the att (z) is returned to the corresponding attribute of the leaf node. Set the threshold value of node x as Kk_x $(0 < k_x < \mathrm{num}_x)$, and select the $(k_x - 1)$ polynomial q_x for it. q_x (0) is the leaf node, which represents the secret of the node.

Random selection of secrets $s \in Z_p$; let $q_R(0) = s$, $q_x(0) = q_{\mathrm{pa}(x)}(\mathrm{num}\ (x))$; Encrypted ciphertext $\mathrm{CT} = \{T, C_1 = M \cdot e(g, g)^{\alpha s}, \quad C = g^{\beta s}, \quad \forall Z \in Z_p, \quad C_z^{(1)} = g^{q_{z(0)}}, C_z^{(2)} = g_{\mathrm{att}(z)}^{xiq_z(1)}$. Send di″and signature to the AMS.

Step 4: the AMS will verify the signature after receiving di′, and the signature, if it passes, stores data di″.

4.2.3. Published in SNSP. The data file is published in the SNSP social network in the form of di″.

4.3. User Data Access

4.3.1. Visitor i Requests an Access Request. Visit i requests access to the AMS by sending requests and ID.

4.3.2. AMS Applies DO's Friend Relationship. After receiving the access request from user i, the AMS applies DO's friend relationship. (Let the parameter of the application friend relationship be t, where the parameter t is the parameter to determine the friend relationship between the AMS and DO. $T = 1$ means it is a close friend; otherwise, it is not a close friend.)

4.3.3. DO Retrieval Validation List V. If the ID of the user is retrieved in the list, it shows that the access user has already requested access to the data and is a close friend, so he jumps directly to Section 4.3.5 and sets it as a close friend.

If no access to the user ID is retrieved, it means that the user requests access to the data for the first time, so the process of encrypting the user ID is entered.

Step of encrypting user ID:

Step 1: enter the access user ID_i, $\mathrm{BL}_{\mathrm{in}}$.

Step 2: retrieve $\mathrm{BL}_{\mathrm{en}}$ with user ID_i.

(i) If you get the query result, record R; this shows that the friend information is saved in the friend list, querying map[t], and determining that the ID of R in $\mathrm{BL}_{\mathrm{en}}$ is the encrypted ID. Enter the next step.

(ii) If not found in the encrypted list (retrieval failure), it is not in the friend list, so it must not be a friend; the query result is returned, and this triggers the update of the user information when the next friend list is updated. Jump directly to step (5).

Step 3: output encrypted ID.

4.3.4. Query the Encrypted Data Table

Step 1: the DO server retrieves encrypted friend list.

Input: encrypted visitor's ID.

Retrieval. In the encrypted data list, find the input data and get the data after the data are returned, that is, the query result R.

Output: decryption the query result R' in the list.

Step 2: the DO server decrypts the query results.

In this paper, the De algorithm is used to decrypt the query results. The main process of the De algorithm is to generate pseudorandom sequence matrix FRS$[n * m]$ using RanSee[0], which is received by the initialization stage. According to ID$'$, to determine the row in the pseudorandom sequence matrix, the pseudorandom code of the matrix is used to perform the XOR operation with R', and the final query result R is obtained, and the query result is returned to the DO.

The De algorithm process is shown in Algorithm 5.

4.3.5. Determine the Parameter Value. When the DO receives the R, it detects that if it is a close friend, it will be added to the verification list V.

According to the result of the query, set the value of parameter t, and send it to the AMS.

4.3.6. AMS Processing. The AMS gets the friend relationship value after receiving the parameter t and distributes different data according to the friend relationship.

If it is a close friend, then send the encrypted file di" and the AMS's parameters of the process of noise adding; If not, then send encrypted file di$'$.

In this way, if it is a common friend, it can obtain the encrypted data file after the first encryption through the decryption; if not, then it fails.

4.3.7. Visitors for Decryption Operations

Step 1: visitor received the result of the query.

Step 2: runs the CoA function to update the private key.

Step 3: if the user is a nonfriend, execute the (1); if the user is a common friend, execute (2), and the first encrypted data file is obtained; otherwise, if the user is a close friend, execute (1)(2), and the user can get the accurate data from the user.

(1) Using the private key decryptioned by the attribute set to get the file di$'$; execute function Decrypt (CT, SK) to decrypt: for each leaf node z, define a recursive function DecryptNode (CT, SK, z). If $\mathrm{att}(z) \in S$, calculate DecryptNode (CT, SK, z) $= (e(\mathrm{Datt}_{(z)}{}^{(1)}, C_z{}^{(1)}))/(e(\mathrm{Datt}_{(z)}{}^{(2)}, C_z{}^{(2)})) = e(g,g)^{yqz(0)}$; if att $(z) \notin S$, then DecryptNode (CT, SK, z) $= \perp$. For every nonleaf node x, let its child node be ck, and k_x is used as a Lagrange interpolation node; calculate $e(g,g)^{yq_{c_k}(0)}$ $M = (Me(g,g)as) / (e(C,D) / e(g,g)^{yqz(0)}) = C_1 / (e(C,D)/e(g,g)^{ys})$.

(2) The user's exact data di are obtained by using encryption parameters.

4.4. Friend List Update

Step 1: add new friendship information.

New friendships include intimate relationships and nonintimate relationships, including new users who need to be recorded in the list in the process of accessing the user's query as well as the user's normal supplement to the information of their friends.

Step 2: delete the users who are not friends.

When the user relationship deteriorates, it will cancel the close friend relationship, so we will update the friend relationship.

Step 3: set the corresponding parameters.

If the intimate relationship has had a user deleted from a friend list, the verification list should also be modified.

4.5. Attribute Revocation. Attribute revocation mainly includes three parts, namely, generate re-encryption key, file re-encryption, and privilege replacement. When the DO revokes a number of user permissions to the file, the re-encryption key was first generated, and the file was re-encrypted with the key. After the user receives the feedback, the key is updated first. If the user is revoked by the DO, the file cannot be accessed. Otherwise, the user's access rights are not revoked.

Step 1: DO updates the user list URL that is needed to revoke permissions and add user value to the list; executing the function Grek to generate the re-encrypted private key; send URL table, re-encryption key, signature, and di$'$ to the AMS.

Step 2: AMS authentication signature.

If the signature is correct, the function Fre is executed to generate the re-encrypted ciphertext; the updated private key is sent to the user who is revoked; and then the URL list is updated after the completion of the transmission.

5. Proof of Security

5.1. Analysis of Privacy Security. The security of the social network privacy protection scheme is mainly to ensure that the AMS will not disclose the information of the encryption process and prevent the AMS from collusion with other illegal users. The security of the algorithm in PPSSN is demonstrated in detail in reference to [22].

5.1.1. The Security of the Algorithm

Theorem 1. *For algorithms En and De, according to the scheme to query operation $q_1, q_2, \ldots, q_n$, the scheme has query security when and only when $q_1, q_2, \ldots, q_m$'s joint information entropy is maximum.*

Proof. This paper is proved by the method of number induction.

When $m = 1$, a query operation is executed, and the record order of the friends list is randomly disturbed, and

the encrypted records change the sequence of the original list data and no longer correspond.

$p(q_1 = 1) = \cdots = p(q_1 = n) = 1/n$, $H(q_1) = H_{\max}(q_1) = \log n$. Therefore, the entropy of every query is the largest, so

$$H(q_1) = H(q_2) = H(q_m) = \log n. \quad (2)$$

When $m = 2$, since the read encrypted records are not related to the two query, Q1 and Q2 are independent for reading record operations, so there are $p(q_1, q_2) = p(q_1)p(q_2)$ and $H(q_1) = H(q_2) = \log n$. Then, each query is independent of each other.

Suppose $p(q_1, \ldots, q_m) = p(q_1)p(q_2)\ldots p(q_m)$; $H(q_1) = \ldots H(q_m) = \log n$, for the $m + 1$ query, the entropy of each query is the largest, and the queries are independent from each other; and $p(q_1, \ldots, q_m, q_m + 1) = p(q_1)p(q_2)\ldots p(q_m)p(q_{m+1})$:

$$H(q_1) = \cdots H(q_m) = H(q_{m+1}) = \log n. \quad (3)$$

We prove the following theorem. □

Theorem 2. *For algorithm En and De, the scheme satisfies nonrelevancy and nontraceability.*

Proof. Because when $\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$ are represented as

$$\begin{aligned} \mathrm{BL}_{\mathrm{ori}} &= \{\mathrm{Rl}[1], \mathrm{Rl}[2], \ldots, \mathrm{Rl}[n]\}, \\ \mathrm{BL}_{\mathrm{en}} &= \{\mathrm{Rl}'[1], \mathrm{Rl}'[2], \ldots, \mathrm{Rl}'[n]\}. \end{aligned} \quad (4)$$

$\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$ are the set of vectors. At this time, $\mathrm{Rl}[k]$ and $\mathrm{Rl}'[j]$ act as random vectors of $\mathrm{BL}_{\mathrm{ori}}$ and $\mathrm{BL}_{\mathrm{en}}$, respectively; then, the probability that the enemy can correctly obtain the mapping relationship between $\mathrm{Rl}[k]$ and $\mathrm{Rl}'[j]$ is $1/n$, and the information entropy of any query is

$$\begin{aligned} H(q) &= \sum_{t-1}^{m} p(xi)\log\frac{1}{p(xi)} = \sum_{t-1}^{m}\frac{1}{p(xi)}\log m \\ &= \log m. \end{aligned} \quad (5)$$

It can be seen that the information entropy of any single query is the largest.

Because each query is independent of each other, we know that the joint information entropy of n secondary query is

$$H(q_1 L q_n) = \sum_{i=1}^{m} H(q_n) = n\log m. \quad (6)$$

It can be obtained that the joint information entropy of n queries is the maximum.

From Theorem 1, we can get the query security of this scheme and satisfy the requirement that it is not related and cannot be traced. □

5.2. Confidentiality

5.2.1. Confidentiality of User Attribute Information. In this PPSNS scheme, the DO generates the user's private key and is responsible for the user's attribute revocation. In addition, the friend list is stored and updated by the DO, and the user's attribute information is only responsible for the DO, so the attribute information is confidential.

5.2.2. Confidentiality of Ciphertext Data. Differential privacy-based data file encryption can set parameters according to certain safety requirements, so it meets certain safety requirements.

5.2.3. Confidentiality of Attribute-Based Encryption Algorithm. The CP-ABE algorithm is proved to be safe under the standard model. The improvements made by this paper are as follows: the approach to generating the key and the phase of encryption. The above has proved the security of the improved algorithm, so the algorithm based on the attribute encryption is also safe.

5.2.4. Confidentiality of Pseudorandom Algorithm. After classifying the users' friends, we store our friend relationship in the DO's friend list. In the user access phase, the DO server will query the friend's list according to the user's access request to judge a friend's relationship, and the query operation of the DO server is not directly queried in the list of friends; however, the friend list $\mathrm{BL}_{\mathrm{en}}$ after the pseudorandom encryption and the DO server cannot directly access the friend list data $\mathrm{BL}_{\mathrm{ori}}$ of the DO. The security of the pseudorandom encryption algorithm has been proved in [18]. Therefore, we can see that the confidentiality of the friend list encryption algorithm is guaranteed in this scheme.

5.2.5. The Security of the Property Revocation Mechanism. When the attribute is revoked, the key will be re-encrypted, and the DO will generate s' randomly, generate a new access tree structure, and generate a new secret of restoring s. If the user has been revoked and cannot update the new property key, the updated cipher cannot be accessed because the new key is re-encrypted with the newly generated random number s'. Although the user's nonupdated $e(g, g)^{\alpha s}$ cannot be used to compute the new $e(g, g)^{\alpha s'}$, the revocation party in this case has forward security.

Assuming that the key generated by the re-encryption is leaked to the AMS, it is still unable to get a new secret, so it does not affect the confidentiality of the re-encrypted ciphertext, and the revocation scheme has a backward security.

5.2.6. Confidentiality of Access Control Policy. In the access control scheme, the re-encrypted operation is only AMS's leaf node operation on the access structure tree. It does not handle the internal nodes, so it cannot obtain the relational function associated with the internal nodes; thus, the access control strategy realizes confidentiality.

5.3. The Resistance of Collusion Attacks

5.3.1. Conspiracy Attack between Illegal Users and AMS. Because the user's private key generation, distribution, and verification are performed by the DO and AMS together, the

AMS is only responsible for verification and does not store private keys. Therefore, even if an unauthorized user is unauthorized to decrypt with the AMS, the private key cannot be illegally obtained, so the attacker cannot conspire to obtain the private key of the user.

5.3.2. Conspiracy Attack between the AMS and DO Server. In this paper, the encryption work based on differential privacy and the core steps based on attribute encryption are performed within the DO; the friends list is also stored in the DO, although the encryption process is in the DO server, but the DO server does not directly view the permissions of the friends list; although it can be verified by a given map[t] when decrypted, the friend parameter t used to communicate between the AMS and DO is not consistent with the friend parameters that are passed between the DO and DO servers. That is, the AMS cannot identify the classification parameters of the friends of the DO server. Therefore, the AMS cannot skip the verification of the DO to directly obtain the parameters of the friend relationship judgment in the DO server. So, although the DO is partially encrypted and distributed to the AMS and DO servers, the core privacy data are kept by the DO personally, which can prevent the AMS and DO servers from obtaining DO release data without DO authentication.

Because of this, illegal users are unable to illegally obtain decryption information with the AMS and DO servers. In sum, this paper can resist conspiracy attacks between AMS and DO servers and illegal users with AMS and DO servers.

5.3.3. Conspiracy Attacks of Illegal Users and DO Servers. The DO server does not store decryption related private keys, so illegal users cannot get information about decrypted data. There is no conspiracy attack between illegal users and DO servers.

6. Performance Analysis

6.1. Complexity Analysis. Compared with the EASiER [19] and scheme (I-WT-LPP) [21], this paper analyzes the complexity from four aspects: the overhead of the private key generation, the storage overhead of the user private key, the overhead of the DO to the data encryption, and the overhead of the decryption of the data by the user.

In the private key generation phase, each user must be considered in the EASiER scheme, so the complexity of the private key is $O(na)$, (n is the average number of users of social networks and a is the average number of attributes corresponding to the private key of the user). In the I-WT-LPP scheme, the DO generates only the main private key, so its complexity is $O(1)$. The PPSSN scheme in this paper does not generate a private key for each user, the complexity of this scheme is related to the number of attributes associated with the user's private key, which is $O(a)$.

For the storage of private keys, each user in the EASiER scheme needs to store the private key that the DO sends, so the complexity is $O(ma)$, (m is the average number of DO related to the user). In the I-WT-LPP scheme, users only store the private key and the private key of the DO distribution, so its complexity is $O(m) + O(a)$. In this paper, we also need to store the private key and private key issued by PPSSN for each user. The complexity of the scheme is $O(m) + O(a)$ for each user in the DO scheme.

D represents the size of the data file, and b indicates the average number of attributes required for encryption. The I-WT-LPP scheme performs three attribute encryptions on the attribute encryption, so its complexity is $O(D) + O(3b)$. The PPSSN encryption is complex in the phase of data encryption, and the schemes of EASiER and PPSSN adopt a double decker encryption mechanism, and their encryption complexity is $O(D) + O(b)$.

In the decryption phase of the user, similar to data encryption, since the three schemes are based on a hybrid encryption mechanism, the complexity of the EASiER schemes is $O(D) + O(c)$ where c represents the average number of attributes required to decrypt. The I-WT-LPP scheme performs three attribute encryptions on the attribute encryption, so its complexity is $O(D) + O(3c)$. The PPSSN encryption complexity is $O(D) + O(c)$.

6.2. Experimental Analysis

6.2.1. Experimental Environment. The experimental environment of this paper is as follows: Intel (R) Core (TM) i3-2370M, main frequency 2.40 GHz, 4.00 GB memory, 200 GB available disk space, Windows7 operating system, and the algorithm implementation code is written in C language. Among them, the attribute-based encryption algorithm is written based on cpabe-0.11 library [23]. In this paper, the scheme PPSSN is compared with the scheme (I-WT-LPP) [21] and the scheme AR-ABE [24]. By comparing the DO's processing consumption time and the storage consumption of each scheme under the same condition, it is possible to analyze the performance.

6.2.2. Experimental Results and Analysis. This paper analyzes the performance of the PPSSN scheme from the following three aspects. First, it analyzes the time consumption changes of the DO of the system with the increase in the number of access queries. Second, it analyzes the time consumption of the DO when the proportion of the close friends in the total number of users is different. Finally, it analyzes the changes in storage space required when the number of visitors is increased.

(1) Set the Number of Users to 100. The proportion of close friends is 10%, and the proportion of ordinary friends is 70%. Three groups of experiments were designed each time, and the average value of experimental results was taken. In Figure 4, the number of leaf nodes from structural tree T was 5, and in Figure 5, the number of leaf nodes of structural tree T was 10. The x-axis represents the number of requests per user access. The y-axis represents the DO time consumption of the system, and the data volume is 5 MB and 10 MB, respectively. When the data volume is

constant, the time spent on the DO side of the three schemes varies with the number of visitor queries, as shown in Figures 4 and 5.

As shown in Figure 4, when the number of leaf nodes is 5 and the data file is 5 MB, the overhead of the DO in the I-WT-LPP scheme and the AR-ABE scheme increased with the increase in the number of access queries. While the system overhead of the PPSSN scheme has not changed much when the number of access queries is increased, it will not increase significantly with the increase in access queries for a certain period of time. When the data file is 10 MB, the system overhead of this scheme is slightly higher than that of the other two schemes when the number of access queries is 1, but with the increase in the number of access queries, the overhead of the I-WT-LPP scheme and the AR-ABE scheme will increase, and the overhead of this scheme has not been greatly fluctuated.

As shown in Figure 5, when the number of leaf nodes is 10, the overhead of the three schemes is basically the same as when the number of leaf nodes is 5. This scheme has an obvious overhead advantage when the number of query times is greater than 1. At the same time, combined with Figures 4 and 5, we can see that when the number of access queries is certain, the cost of the DO of the system will also increase when the number of leaf nodes increases.

In sum, we know that when this scheme is compared with the I-WT-LPP and AR-ABE schemes, the number of access queries is 1, and the overhead of this scheme is slightly larger than the other two schemes; however, when the number of access queries is more than 1, the system overhead is maintained in a certain range, and it will not have an obvious increasing trend, and the advantage is obvious. Therefore, it shows that this scheme can provide better quality query services through the classification of friends with different users with different rights to provide, and it has a certain advantage over the other two schemes.

(2) Set the Number of Requests per User as Two. The total number of users is 100, of which the proportion of unfamiliar users is 10%. In Figure 6, the number of leaf nodes from the structural tree T is 5, and Figure 7 takes the number of leaf nodes of structural tree T to 10. The x-axis indicates the percentage of close friends in the total number of users. The y-axis represents the DO time consumption of the system, and the data volume is 5 MB and 10 MB, respectively. When the proportion of close friends in the total number of users is different, the time consumption changes of the query at the DO of the system are shown in Figures 6 and 7.

As shown in Figures 6 and 7, when the number of leaf nodes and the size of data files are certain and when the proportion of close friends increases, the time consumption of the system DO of the I-WT-LPP and AR-ABE schemes is basically unchanged, but the time consumption of the PPSSN scheme is significantly reduced, indicating that the implementation efficiency of this scheme is related to the ratio of close friends. In addition, because this scheme can store access records and reduce the cost of authentication in future access requests after receiving close friends access, the higher the ratio of intimate friends, the more obvious the superiority of this scheme is. At the same time, when the number of leaf nodes is increased, the overhead of the system is increasing; when the data file increases, the overhead of the DO increases, which is in accordance with the experimental results of the previous article.

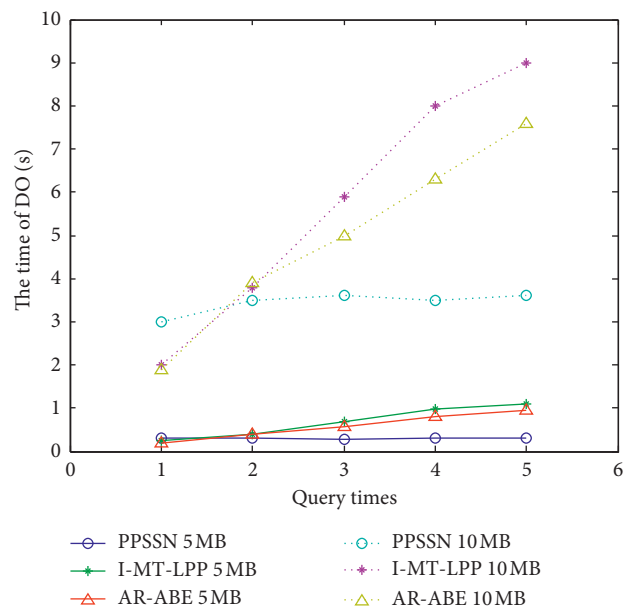


Figure 4: The time consumption of the DO when the leaf node is 5.

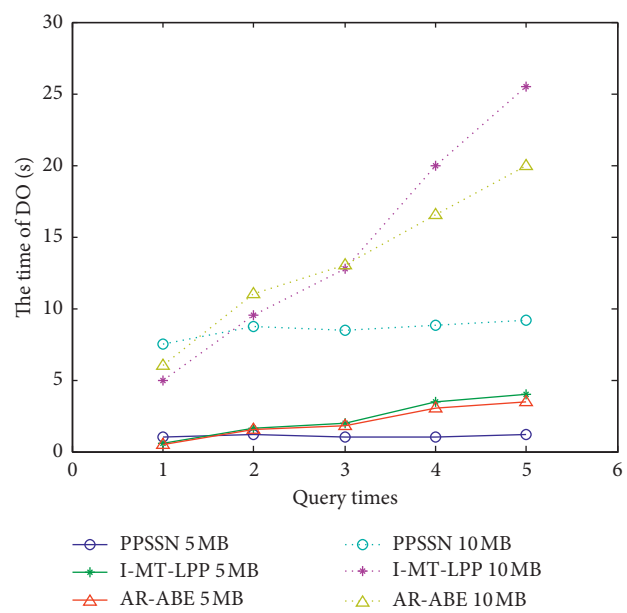


Figure 5: The time consumption of the DO when the leaf node is 10.

(3) Set the Number of Requests per User to Two. The number of leaf nodes is 5, the proportion of close friends is 10%, and the proportion of general friends is 70%. The x-axis represents the number of users accessing the social network, and the y-axis represents the actual storage space of the system. The amount of data is 20 MB and 50 MB,

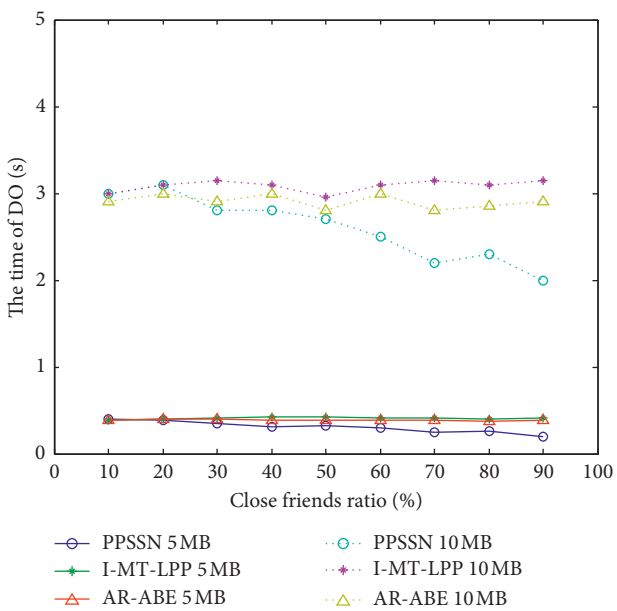


Figure 6: The time consumption of the DO when the leaf node is 5.

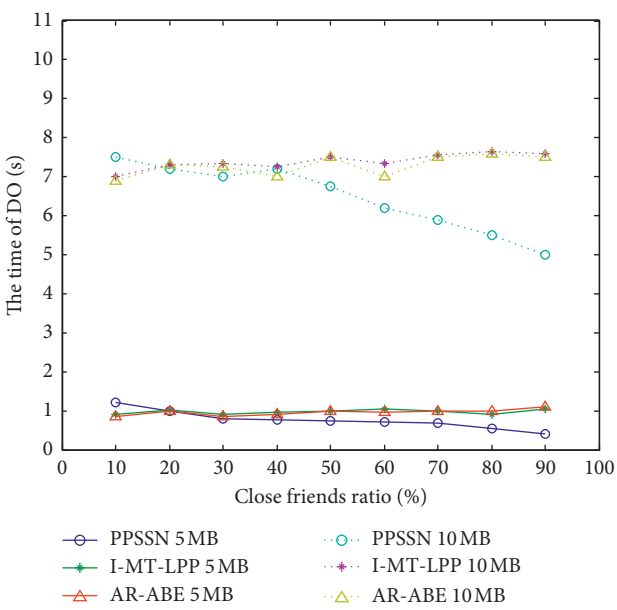


Figure 7: The time consumption of the DO when the leaf node is 10.

respectively. When the number of access users increases, the storage space of the system DO must be changed as shown in Figure 8.

As shown in Figure 8, when the number of data files of the leaf node number and the number of instances of user access is certain, this scheme is basically poor given the storage space occupied by the I-WT-LPP and AR-ABE schemes. Under the same conditions, the increase in the number of leaf nodes brings additional system storage cost, and the increase in data file size will also increase the storage cost of the system. When the number of leaf nodes and the size of the data file is certain, with the increase in the number of users, the three schemes increase the cost of the system at the same time, and the

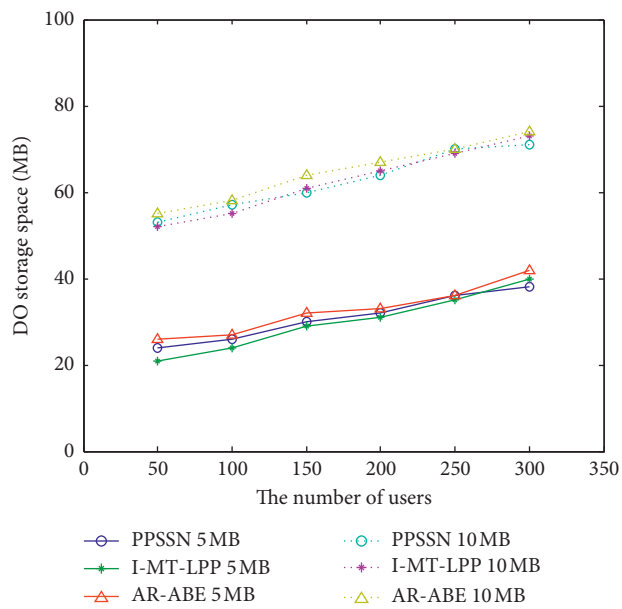


Figure 8: Changes in DO storage space.

increase in the amount of storage is not much different; this scheme adds a friend list to the DO and verifies the buddy relationship through a list, which adds a certain amount of overhead, but in the private key generation, the I-WT-LPP scheme and the AR-ABE scheme increase the storage overhead, so the total system overhead of the three cases is not much different. In this paper, the query efficiency is improved without additional storage overhead.

In sum, because the buffer mechanism is set up in the PPSSN scheme, the DO cost is greatly reduced when the number of visitors per capita increases. At the same time, the DO overhead of the PPSSN scheme is associated with the proportion of close friends, and as the proportion of close friends increases, the cost of the system is reduced as the access user is classified according to the intimacy. In addition, the storage cost of the system has not increased greatly, but it has been controlled within a certain range and is almost the same as the other two schemes. Therefore, PPSSN can improve the quality of service and provide fine-grained queries; meanwhile, the system overhead is basically unchanged, and there is a significant advantage.

7. Conclusion

In this paper, a social network privacy protection scheme is proposed. In the aspect of key generation, the identity of the user is verified by the AMS, and the property key is distributed to the legitimate user. The DO will generate the corresponding user key to its friends according to the friends list, and the visitors who get two keys can access the accurate information of the user according to the key. Since this scheme is classified according to the identity of the visitor, the data owner has a list of friends, and the user in the list can provide higher availability when the data are accessed on the platform. Under certain privacy protection levels, some users can improve the availability of data.

In addition, this scheme proposes a verification list V to reduce the subsequent access overhead to the users that have been visited. The buffer mechanism has an obvious advantage in the case of increasing query, and it is stored in the DO, so there is no risk of disclosing information.

Of course, this paper also has some shortcomings: while improving the quality of the query service and efficiency, the storage space would benefit from improvement, which will be considered in future research.

Data Availability

The all type data used to support the findings of this study have been deposited in the UCI Machine Learning Repository (http://archive.ics.uci.edu/ml/datasets.html). There are three data sets in this paper. The first data set is Amazon Commerce reviews set Data Set (http://archive.ics.uci.edu/ml/datasets/Amazon+Commerce+reviews+set). This data set mainly collects the online business evaluation of different users on Amazon website and records it as data set D1. The second data set is the credit card customer liquidated damages data set (http://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients). It makes statistics and collates the situation of customer liquidated damages in Taiwan and records it as data set D2. The third data set is an anonymous sample of access records to Amazon's internal resources (http://archive.ics.uci.edu/ml/datasets/Amazon+Access+Samples). The sampling time is from 0:00 on March 1, 2005, to 31:23 p.m. on August 31, 2010, and is recorded as data set D3.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 61572260, 61872196, 61872194, and 61402241, in part by the Jiangsu Natural Science Foundation for Excellent Young Scholar under Grant no. BK20160089, in part by Scientific & Technological Support Project of Jiangsu Province under Grant no. BE2017166, in part by Research of Natural Science of NJUPT under Grant no. NY217050, and in part by Jiangsu Government Scholarship for Overseas Studies.

References

[1] Z. Lv, H. Cheng, A. Chang, F. Dengguo, and C. K. Qu, "Privacy protection scheme for social network," *Journal of Communication*, vol. 35, no. 8, pp. 23–32, 2014.

[2] Y. Zhang, Y. Shen, H. Wang, Y. Zhang, and X. Jiang, "On secure wireless communications for service oriented computing," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 318–328, 2018.

[3] H. Wang, Z. Zhang, and T. Taleb, "Editorial: special issue on security and privacy of IoT," *World Wide Web*, vol. 21, no. 1, pp. 1–6, 2018.

[4] R. K. Lomotey, J. C. Pry, and C. Chai, "Traceability and visual analytics for the Internet-of-Things (IoT) architecture," *World Wide Web*, vol. 21, no. 1, 2018.

[5] F. Chen, Y. Luo, J. Zhang et al., "An infrastructure framework for privacy protection of community medical internet of things," *World Wide Web*, vol. 21, no. 1, 2018.

[6] Q. Huang, L. Wang, and Y. Yang, "DECENT: secure and fine-grained data access control with policy updating for constrained IoT devices," *World Wide Web-Internet & Web Information Systems*, vol. 11, pp. 1–17, 2017.

[7] A. Shamir, "Identity-based cryptosystems and signature schemes," in *LNCS 196: Proceedings of the Advances in Cryptology (CRYPTO)*, pp. 47–53, SpringerVerlag, Berlin, Germany, 1985.

[8] D. Boneh and M. Franklin, "Identity-based Encryption from the Weil pairing," in *Proceedings of the Advances in Cryptology-(CRYPTO 2001)*, pp. 213–2293, SpringerVerlag, Berlin, Germany, 2001.

[9] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT 2005, LNCS*, R. Cramer, Ed., vol. 3494, Springer, Heidelberg, Germany, pp. 457–473, 2005.

[10] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security-CCS '06*, pp. 89–98, Alexandria, VA, USA, November 2006.

[11] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP '07)*, pp. 321–334, Washington, DC, USA, May 2007.

[12] B. Waters, "Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization," in *Proceedings of the 14th IACR International Conference on Practice and Theory of Public Key Cryptography (PKC 2011)*, pp. 53–70, Taormina, Italy, March 2011.

[13] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proceedings of the USENIX Security Symposium*, vol. 3, Bellevue, WA, USA, August 2011.

[14] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security-ASIACCS '10*, pp. 261–270, Sydney, Australia, March 2010.

[15] T. Naruse, M. Mohri, and Y. Shiraishi, "Attribute revocable attribute-based encryption with forward secrecy," *IPSJ Journal*, vol. 55, no. 10, pp. 2256–2264, 2014, in Japanese.

[16] Z. Liu, Z. F. Cao, and D. C. S. Wong, "Traceable CP-ABE: how to trace decryption devices found in the wild," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 55–68, 2015.

[17] Z. Liu, Z. F. Cao, and D. C. S. Wong, "Blackbox traceable CP-ABE: how to catch people leaking their keys by selling decryption devices on ebay," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security-CCS '13*, pp. 475–486, New York, NY, USA, November 2013.

[18] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication (SIGCOMM 2009)*, pp. 135–146, Barcelona, Spain, August 2009.

[19] S. Jahid, P. Mittal, and N. Borisov, "EASiER: encryption-based access control in social networks with efficient revocation," in *Proceedings of the 6th ACM Symposium on Information,*

Computer and Communications Security (ASIACCS 2011), pp. 411–415, HongKong, China, March 2011.

[20] X. Liang, X. Li, R. Lu, X. Lin, and X. Shen, "An efficient and secure user revocation scheme in mobile social networks," in *Proceedings of the International Conference on Global Telecommunications Conference (GLOBECOM 2011)*, Houston, TX, USA, December 2011.

[21] X. Lin, Y. Han, K. Yan, and X. Yang, "Location privacy preserving scheme against attack from friends in SNS," *Journal of Communication*, vol. 37, no. S1, pp. 224–230, 2016.

[22] S. T. Yang, C. G. Ma, and C. L. Zhou, "Privacy protection model and scheme for LBS," *Journal of Communication*, vol. 35, no. 8, pp. 116–124, 2014.

[23] J. Bethencourt, A. Sahai, and B. Waters, "Advanced crypto software collection: the cpabe toolkit [EB/OL]," March 2011, http://acsc.cs.utexas.edu/cpabe/.

[24] T. Naruse, M. Mohri, and Y. Shiraishi, "Attribute revocable attribute-based encryption with forward secrecy for fine-grained access control of shared data," *IEICE Technical Report Information & Communication System Security*, vol. 114, no. 10, pp. 181–186, 2015.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 5498375, 16 pages
https://doi.org/10.1155/2019/5498375

Research Article

Social Security and Privacy for Social IoT Polymorphic Value Set: A Solution to Inference Attacks on Social Networks

Yunpeng Gao [1] **and Nan Zhang** [2]

[1] *School of Engineering and Applied Science, George Washington University, Washington, DC, USA*
[2] *Kogod School of Business, American University, Washington, DC, USA*

Correspondence should be addressed to Yunpeng Gao; ypgao@gwu.edu

Received 27 April 2019; Accepted 4 August 2019; Published 28 August 2019

Guest Editor: Houbing Song

Social Internet of Things (SIoT) integrates social network schemes into Internet of Things (IoT), which provides opportunities for IoT objects to form social communities. Existing social network models have been adopted by SIoT paradigm. The wide distribution of IoT objects and openness of social networks, however, make it more challenging to preserve privacy of IoT users. In this paper, we present a novel framework that preserves privacy against inference attacks on social network data through ranked retrieval models. We propose PVS, a privacy-preserving framework that involves the design of polymorphic value sets and ranking functions. PVS enables polymorphism of private attributes by allowing them to respond to different queries in different ways. We begin this work by identifying two classes of adversaries, authenticity-ignorant adversary, and authenticity-knowledgeable adversary, based on their knowledge of the distribution of private attributes. Next, we define the measurement functions of utility loss and propose PVSV and PVST that preserve privacy against authenticity-ignorant and authenticity-knowledgeable adversaries, respectively. We take into account the utility loss of query results in the design of PVSV and PVST. Finally, we show that PVSV and PVST meet the privacy guarantee with acceptable utility loss in extensive experiments over real-world datasets.

1. Introduction

1.1. Motivation. SIoT integrates social network schemes into IoT systems, which provides opportunities for IoT objects to form social networks. The SIoT paradigm is promising as it is believed that SIoT structures are helpful in enhancing the navigability of IoT networks, identifying levels of trustworthiness and reusing existing social network models [5]. In this scenario, privacy and security issues have been extensively studied [6, 7, 24, 48]. However, current studies in privacy preservation of IoT systems focus on access control [23, 46], communication and authentication protocols [4, 34, 44], and attribute-based encryption [39, 43]. The features of social network have not been thoroughly considered.

The nature of online social networks (OSN) requires sharing of information. User information, including activity patterns and descriptive attributes, is mined and analyzed to improve user experience of OSN applications. Third party users also take advantage of the huge amount of data collected by social networks [21]. As part of the improvement of user experience, ranked retrieval models have been extensively studied and applied to many OSN features, e.g., link prediction and recommendation systems [32, 36, 47]. For instance, given a user's information (which can be a tuple in a database), a ranked retrieval model returns a ranking result that serves OSN features, e.g., "People you may know" and "Recommended for you." Furthermore, many OSN providers improve the accuracy of ranked results by taking into account private attributes of users in the ranked retrieval model. For example, sensitive demographics such as race, religion, and income can help in friend recommendation features as intuitively people sharing similar demographics are more likely to be interested in each other.

OSN providers relieve users' concern of privacy leakage by allowing them to mark attributes as "private" and hiding

private attributes from profiling pages and ranked results. Users believe that their privacy is well protected since private attributes are invisible to the public in their profiles or any ranked results. However, Rahman et al. [35] proposed Rank Inference and showed that privacy of private attributes in the ranked retrieval model is not guaranteed. In their approach, the value of a private attribute can be inferred through a ranked retrieval interface given the premises that the domain of the private attribute is finite and that the ranking function has both monotonicity property and additivity property [35]. To be more specific, given monotonicity and additivity conditions, an adversary is always able to find a pair of differential queries, q_θ and q'_θ, such that (a) q_θ and q'_θ share the same predicate on all attributes except for a private attribute B_1, (b) the value of B_1 in q'_θ is θ while the value of B_1 in q_θ is not, and (c) the ranked result of q_θ contains the victim tuple v while the ranked result of q'_θ does not. Rahman et al. showed that given the above conditions, the adversary was able to conclude that the value of v's B_1 is not equal to θ. Furthermore, the adversary was able to infer the value of v's B_1 by finding more differential queries and excluding more values from the domain of B_1, as long as the domain is finite.

Privacy issues arise when private attributes of users are taken into account in the ranked retrieval model. Intuitively, the issues can be solved by removing private attributes from ranking functions, which decreases the utility of many OSN features, the recommendation system cannot provide accurate results. From OSN providers' perspective, removing private attributes is not a practical solution. Therefore, this work aims not only to address the issue of rank inference but also to propose a framework that preserves the privacy of users against all inference attacks through the ranked retrieval model while minimizes utility loss.

1.2. Related Work. Encryption technologies have been used throughout history in security and privacy preservation. Searching on encrypted data [10, 38] has been introduced to ensure data privacy. Cao et al. [9] proposed a scheme that allows privacy-preserving ranked search over encrypted data. A query consisting of multiple keywords is conducted by searching over encrypted documents with secure k-nearest neighbor (kNN) technique. Chen et al. [11] took into consideration correlations between documents before conducting a search query and achieved better performance. Vertical fragmentation [13, 15, 16, 22] has been applied to encrypted data, which hides identities of users by separating identifier attributes with descriptive attributes. However, encrypted searching is developed to preserve privacy against adversaries in a cloud computing environment. Adversaries can still access decrypted searching results from which private attribute values can be inferred.

As ONS has been emerging as an important source for big data, many studies have been carried out for privacy-preserving data mining (PPDM) and privacy-preserving data publishing (PPDP). Perturbative methods implement the "camouflage" paradigm where original data are directly modified [28]. Agrawal et al. [3] proposed an algorithm that perturbs data with random additive noise. Liu et al. proposed data perturbation with multiplicative noise. However, random noise has predictable structures in the spectral domain, and thus, privacy provided by additive noise is questionable [25]. Furthermore, additive or multiplicative noise can only be applied to numerical data. Data swapping approaches [19, 30, 31] perturb data by swapping values between records that are close to each other. Other distance-based approaches include [12] in which data points are perturbed without changing their relevant closeness relationships and [2] in which data points are clustered and each data point's value is replaced by the value of the cluster center. However, those approaches rely on a universal measurement of closeness between data points in multidimensional space. Furthermore, they are limited by the distributions of data points.

Generalization is the process of replacing a group of values with a more general value that can represent the group. Suppression is the ultimate state of generalization such that the representative value is "not applicable" and as a result, the group of values is removed from the dataset [45]. k-anonymity [40, 41] is a widely studied approach that preserves privacy of records by grouping at least k records into an equivalence class. The attribute values of the k records are suppressed so that the k records are indistinguishable from adversaries. Machanavajjhala et al. [27] proposed L-diversity that focuses on attribute privacy. L-diversity forces each equivalent class to have at least l different values for each attribute. Li et al. [26] proposed T-closeness that further considers the distribution of attribute values. T-closeness sets a threshold for the variance between the distribution of a private attribute in each equivalence class and the distribution of the same attribute in the entire dataset. However, those approaches are developed to preserve privacy of published data, while the ranked retrieval model of ONS does not directly reveal private attributes to the public. Furthermore, suppression of private attribute values introduces unnecessary utility loss to the ranked retrieval model. Equivalent Set [20] was proposed to preserve privacy against inference attacks through the ranked retrieval model. This approach groups different tuples into a set such that they are indistinguishable in ranked results. However, this approach requires that tuples in the same equivalent set have different values in every private attribute and have the same value in every public attribute. Assume that a kNN query q is sent to a ranked retrieval interface protected by Equivalent Set and that a tuple t is equal to q in most private attributes. Since the other tuples in the same equivalent set of t are different from t in every private attribute, they must be different from q in most private attributes too. Therefore, the original rank of t given q should be much higher than that of any other tuple in the same equivalent set. In this case, the rank of t given q will be significantly lowered by Equivalent Set in order to achieve indistinguishability, which reduces the accuracy of the ranked result of q. Furthermore, it is possible that there is no t' such that t' is different from t in every private attribute and is same with t in every public attribute. In this case, we have to suppress the private attribute values of t, which further introduces utility loss.

Differential privacy [8, 17, 18] is another widely studied framework that preserves privacy of published datasets or hidden databases. It imposes a strong guarantee of privacy on tuples in statistical databases by adding noise to the process of query results. However, the ranked retrieval model outputs ranks of tuples, instead of their values or aggregate statistics. We cannot directly add noise to ranked results as the rank of a tuple is determined by not only the tuple itself but also by other tuples in the database. Furthermore, the optimization of the ranked retrieval model has not been considered.

1.3. Contributions. This work presents a novel scheme for privacy-preserving the ranked retrieval model. We start with an introduction to the adversary model and introduce our definition of privacy guarantee. We identify two categories of adversaries based on their prior knowledge and assume that adversaries can launch optimal inference attacks through ranked results.

We propose the polymorphic value set (PVS), a privacy-preserving framework for the ranked retrieval model. Different from existing methods, PVS does not directly modify values of tuples or query results. Instead, PVS enables polymorphism of private attributes such that a private attribute of a tuple can respond to different queries in different ways. We prove that our framework meets the privacy guarantee stated in Problem Statement. For adversaries with and without prior knowledge, we design and implement the polymorphic value set with true values (PVST) and polymorphic value set with Virtual Values (PVSV), respectively. In the design of PVST and PVSV, we consider utility loss in the ranked retrieval model and propose a practical measurement of utility loss. We prove that the task of minimizing utility loss is NP-hard and present two heuristic algorithms that implement PVST and PVSV, respectively. We run our implementations of PVST and PVSV on a real-world dataset from eHarmony [29] that contains 486,464 tuples. The experiments yield excellent results with respect to privacy guarantee and utility loss.

The remainder of this paper is organized as follows. Problem Statement introduces the adversary model and the privacy guarantee. Privacy-Preserving Framework introduces our privacy-preserving framework. Framework with Virtual Values presents the design and implementation of PVSV, along with our analysis of utility loss. The implementation of PVST and analysis of utility loss are presented in Framework with True Values. Experimental Results contains our experimental evaluation of PVSV and PVST. In Conclusions, we conclude this paper with a summary of our key contributions and a discussion of some open problems.

2. Problem Statement

2.1. Ranked Retrieval Model. In information retrieval, we have witnessed extensive research in the ranked retrieval model. Unlike the Boolean retrieval model where only results that exactly match the predicates can be returned, the ranked retrieval model allows users to retrieve a list of records sorted by a proprietary ranking function. Therefore, the ranked retrieval model provides an alternative solution for users seeking results sorted by their relevance to the query.

As discussed in the introduction, many OSN applications have been using the ranked retrieval model to process incoming queries. Upon a query q, the ranked retrieval model would calculate each tuple t's score according to a proprietary score function $s(t \mid q)$ and return top-k tuples in descending order of their scores. The attributes of tuples could be either categorical or numerical. In this paper, we consider only categorical data, which does not limit the scope of our research. Actually, numerical data can be treated as categorical data by categorizing the numerical domain into small intervals such that no more than one tuple in the database falls into the same interval. Without loss of generality, we also assume that there is no duplicate tuple that is equal to another tuple in every attribute.

We now formalize our ranked retrieval model with categorical attributes. Consider an n-tuple database D with m public attributes $A_1, \ldots, A_m$ and m' private attributes $B_1, \ldots, B_{m'}$. Let V_i^A(resp. V_j^B) denote the value domain of A_i(resp. B_j). Let $t[A]_i$(resp. $t[B]_j$) denote the value of t in A_i(resp. B_j). Upon query q, the score function $s(t \mid q)$ computes a score for each tuple $t \in D$. The ranked retrieval model will then sort all tuples in D in the descending order and return them as the ranked result. We consider the case where the score function is linear. Therefore, $s(t \mid q)$ can be defined as

$$s(t \mid q) = \sum_{i=1}^{m} w_i^A \rho(q[A_i], t[A_i]) + \sum_{j=1}^{m'} w_i^B \rho(q[B_i], t[B_i]), \tag{1}$$

where w_i^A(resp. w_j^B) $\in (0, 1)$ is the weight of attribute A_i(resp. B_j) in the score function, and the matching function $\rho(q[A_i], t[A_i])$, (resp. $\rho(q[B_i], t[B_i])$) indicates if t matches q in attribute A_i (resp. B_j). Therefore, the value of $\rho(\beta_1, \beta_2)$ is 1 if β_1 is equal to β_2, and the value of $\rho(\beta_1, \beta_2)$ is 0 if β_1 is not equal to β_2. Note that our ranked retrieval model satisfies the monotonicity and additivity properties defined in [35].

2.2. Adversary Model. In Motivation, we mentioned that we do not make any assumption about the method adopted by an adversary when attacking a database. We also assume that the adversary has prior knowledge about the metadata of tables in the database, as well as the proprietary ranking function. Furthermore, the adversary is assumed to be able to issue queries to the ranked retrieval model, view ranked results, and insert tuples to the database. As a result, the adversary is able to retrieve all public attribute values by crawling the database through the query interface [37]. We denote the set of queries issued by the adversary as Q_A, the set of tuples inserted by the adversary as I_A, and the corresponding set of ranked results as R_D. R_D is fully determined by Q_A and I_A given fixed D. We name all information regarding a tuple $t \in D$ that an adversary can find in R_D as the *trace* of t and denote the trace of t as T_t. The trace of t includes, but not limited to, the rank of t and the

relationship between t and any other tuple (e.g., t has a higher or lower rank than another tuple t') in a ranked result. Therefore, given fixed D, I_A and Q_A, T_t is fully determined by the attribute values of t.

Another capability of the adversary we model is the adversary's prior knowledge. Consider an extreme case where the adversary knows the equivalence relation between two attributes B_1 and A_1. In this case, even without R_D, the adversary is still able to infer the value of B_1 of any $t \in D$. In reality, an adversary can acquire such attribute correlations by being or consulting an expert in the domain of the dataset or by adopting data mining methods [42]. For example, based on the personal information (e.g., gender, ethnicity, age, and blood type which can be used to infer the gene) stored as public attributes and published in public medical data repositories, genetic epidemiologists can generally conclude that an individual does not have some diseases, merely based on the fact that these diseases would never be found by the candidate gene among historic medical datasets. Therefore, an adversary with prior knowledge could eliminate the possibility of a certain tuple in the database. We model prior knowledge as a function $\mathrm{PK}(t \mid D)$ that takes as input t and D and returns either 0 or 1. $\mathrm{PK}(t \mid D) = 0$ indicates that, given prior knowledge, the possibility of $t \in D$ is zero. $\mathrm{PK}(t \mid D) = 1$ indicates that the adversary cannot eliminate the possibility of $t \in D$. As prior knowledge helps adversaries in launching an inference attack, adversaries can be partitioned into two classes: adversaries with prior knowledge and adversaries without prior knowledge of the dataset.

Definition 1. We name adversaries without prior knowledge of the authenticity of any tuple as authenticity-ignorant adversaries. For authenticity-ignorant adversaries, $\mathrm{PK}(t \mid D)$ always outputs 1. We name adversaries with such prior knowledge as authenticity-knowledgeable adversaries. For authenticity-knowledgeable adversaries, $\mathrm{PK}(t \mid D) = 1$ if $t \in D$ and $\mathrm{PK}(t \mid D) = 0$ if $t \notin D$.

The objective of both classes of adversaries is to maximize the following $g(v, B_j)$ value when inferring the value of victim tuple v in attribute B_j:

$$g\left(v, B_j\right) = \Pr\left(v\left[B_j\right] = a\right), \tag{2}$$

where $\Pr(v[B_j] = a)$ is the probability of $v[B_j] = a$ and a is the value inferred by the adversary given prior knowledge and ranked results.

For authenticity-ignorant adversaries, $\mathrm{PK}(t \mid D)$ always outputs 1 regardless of t and D. We only assume the cases where users input true information to the databases. Therefore, for authenticity-knowledgeable adversaries, $\mathrm{PK}(t \mid D) = 1$ if $t \in D$ and $\mathrm{PK}(t \mid D) = 0$ if $t \notin D$. In this paper, we assume a strong adversary that can infer the private attribute values of an arbitrary tuple, given the premise that the trace of the target tuple is unique. The premise can be easily met because as long as there is no duplicate tuple in the dataset, the adversary can always construct well-designed I_A and Q_A such that the trace of the target tuple is different from the trace of any other tuple. Therefore, the adversary can always find a such that $\Pr(v[B_j] = a) = 100\%$.

2.3. Problem Statement. A privacy breach can be described by a successful inference of a private attribute value in the database. We view privacy of $v[B_j]$ as the upper bound on the possibility that an adversary succeeds in inferring the value of $v[B_j]$. Note that we do not make any assumption about the adversary's attacking method. Our objective in this paper is to present a framework that sets an upper bound on the probability of successful inference of an arbitrary private attribute for tuple $t \in D$. Therefore, we define the objective of the framework as

$$\forall t \in D, j \in \{1, \ldots, m'\}, g\left(v, B_j\right) \leq \varepsilon. \tag{3}$$

We present the upper bound ϵ as our privacy guarantee.

However, a privacy-preserving framework should provide not only a privacy guarantee but also a notion of utility—after all, a framework that removes all private attribute values or replaces them with randomly generated values can surely preserve privacy. Therefore, we use a measurement based on the variance of ranked results before and after adopting our framework. Given D and a set of all possible queries denoted as Q, we define the utility loss for our ranked retrieval model as follows:

$$U = \sum_{t \in D} \sum_{q \in Q} \left|\mathrm{Rank}(t \mid q) - \mathrm{Rank}'(t \mid q)\right|, \tag{4}$$

where $\mathrm{Rank}(t \mid q)$ and $\mathrm{Rank}'(t \mid q)$ refer to the ranks of tuple t in the ranked result given query q before and after applying our frameworks, respectively.

3. Privacy-Preserving Framework

The only information an adversary can obtain from a database through the ranked retrieval model is public attribute values and ranked results. For an adversary without prior knowledge, information regarding private attribute values can only be retrieved from ranked results. Therefore, in order to preserve privacy, we have to modify the ranked retrieval model such that the adversary cannot retrieve any useful information about private attributes from ranked results.

An idea is to group different tuples together in ranked results. As in our prior work [20], we can group two tuples v_1 and v_2 together such that they share the same rank in any ranked result. This can be achieved by adopting a new ranking function $s'(t \mid q)$ such that $s'(v_1 \mid q) = s'(v_2 \mid q)$ for all q. If v_1 and v_2 have different values on every private attributes, then the adversary is unable to infer the private values of v_1 since v_1 and v_2 are indistinguishable in any ranked results. However, this method suffers from high utility loss. In order to preserve the privacy of all private attributes, v_1 and v_2 have to be different over all private attributes. Thus, the original scores of v_1 and v_2, i.e., $s(v_1 \mid q)$ and $s(v_2 \mid q)$, differ a lot, which leads to a higher variance between the rank of v_1 before and after adopting this method.

In this work, we present a novel framework that preserves privacy of private attributes while minimizes the utility loss. We observe that for a tuple v's private attribute $v[B_j]$ if there are at least two potential values for $v[B_j]$ and

an adversary cannot differentiate any one of them, then the privacy of $v[B_j]$ can be preserved. For instance, if the probability of $v[B_j] = a$ is equal to the probability of $v[B_j] = a'$, given ranked results and prior knowledge, then the adversary cannot exclude any one of them. If both the probabilities are 50%, then the adversary may choose to randomly pick a value from a and a' as the inferred result. In this case, $g(v, B_j)$ will not exceed 50% and privacy of $v[B_j]$ can be preserved. To prove this statement, suppose that v is an arbitrary tuple in database D, and we want to preserve the privacy of $v[B_j]$. Let β_j^B be an arbitrary value in $V_j^B \backslash \{v[B_j]\}$. We construct tuple v' such that v' and v differ in only one attribute B_j: $v'[B_j] = \beta_j^B$. We also construct database D' such that D and D' differ in only one tuple $v \in D$ while $v' \in D'$. We define a new score function $s'(t \mid q)$:

$$s'(t \mid q) = \sum_{i=1}^{m} w_i^A \rho(q[A_i], t[A_i]) + \sum_{j=1}^{m'} w_i^B \rho'(q[B_j], t[B_j]), \tag{5}$$

where

$$\begin{aligned} &\rho'(q[B_j], t[B_j]) = \rho(q[B_j], t[B_j]), \quad \text{if } t \neq v \text{ and } t \neq v', \\ &\rho'(q[B_j], t[B_j]) = \text{Max}(\rho(q[B_j], v[B_j]), \rho(q[B_j], v'[B_j])), \end{aligned} \tag{6}$$

if $t = v$ or $t = v'$

Imagine a case where an adversary queries D and D' with the same query workload Q_A. We denote the ranked results from D as R_D and the ranked results from D' as $R_{D'}$. As in the new score function, $s'(v \mid q) = s'(v' \mid q)$ for $\forall q \in Q_A$, R_D is identical to $R_{D'}$. Therefore, given only ranked results, the adversary cannot tell the difference between the two databases being queried. Furthermore, even if we exchange the values of v and v', the adversary still cannot observe any change in R_D or $R_{D'}$. As a result, the value of $v[B_j]$ and β_j^B are equivalent from the perspective of the adversary, and the privacy of $v[B_j]$ can be well preserved. Intuitively, v can be seen as a tuple that has two polymorphic forms in B_j: $v[B_j]$ and β_j^B. When calculating the score of v with $s'(t \mid q)$, we always choose the value that can maximize $s'(v \mid q)$.

We can further extend the statement to a more general case. For each tuple $v \in D$ and each private attribute B_j, we can select e distinct values $\beta_1, \beta_2, \ldots, \beta_e$ from $V_j^B \backslash \{v[B_j]\}$, $e < |V_j^B| - 1$. The new score function can be defined as

$$s'(t \mid q) = \sum_{i=1}^{m} w_i^A \rho(q[A_i], t[A_i]) + \sum_{j=1}^{m'} w_i^B \rho'(q[B_j], t[B_j]), \tag{7}$$

where

$$\begin{aligned} &\rho'(q[B_j], t[B_j]) = \rho(q[B_j], t[B_j]), \quad \text{if } t \neq v, \\ &\rho'(q[B_j], t[B_j]) = \text{Max}(\rho(q[B_j], t[B_j]), \rho(q[B_j], \beta_1), \\ &\qquad \ldots, \rho(q[B_j], \beta_e)), \quad \text{if } t = v. \end{aligned} \tag{8}$$

Consider e tuples $v(1)', \ldots, v(e)'$ which are identical to v in all attributes except for B_j. Let $v(i)'[B_j]$ be β_i, $\forall i = 1, \ldots, e$. Then for $\forall q$, we have $s'(v \mid q) = s'(v(i)' \mid q)$. As a result, from the adversary's perspective, there are $e + 1$ potential values for $v[B_j]$: $v[B_j]$, $\beta_1, \ldots, \beta_e$, which are indistinguishable from each other from any ranked results. The privacy of $v[B_j]$ can be preserved by grouping it with e equivalent values.

As such, we introduce the construction of the polymorphic value set (PVS). We put $v[B_j]$ into a set in which all values are indistinguishable when calculating the score with respect to B_j in the ranking function, i.e., $\rho'(q[B_j], v[B_j])$. We name the above set as the polymorphic value set and denote the polymorphic value set of tuple v in attribute B_j as $P_v^{B_j}$. We define $P_v^{B_j}$ as follows.

Definition 2. $P_v^{B_j}$ is a set containing all indistinguishable values of tuple v's private attribute B_j. Assigning $v[B_j]$ with an arbitrary value in $P_v^{B_j}$ will not change the value of $s'(v \mid q)\, \forall q$, i.e., $\rho'(q[B_j], v[B_j]) = \rho'(q[B_j], \beta), \forall \beta \in P_v^{B_j}$ and $\forall q$.

Since the adversary cannot distinguish different values in $P_v^{B_j}$ by launching any inference attacks based only on ranked results, the privacy guarantee of $v[B_j]$ is

$$\varepsilon = \frac{1}{|P_v^{B_j}|}. \tag{9}$$

In order to achieve the privacy guarantee defined in (3), for each $v \in D$ and each private attribute B_j, we have to ensure that (a) there is one and only one polymorphic value set $P_v^{B_j}$ for v's attribute B_j and (b) the privacy guarantee defined in (9) is always valid.

4. Framework with Virtual Values

4.1. Design. In this section, we introduce how polymorphic value sets can be constructed with generated values to meet the privacy guarantee in (9) against authenticity-ignorant adversaries. We name values generated by our framework as virtual values.

As we proved in Privacy-Preserving Framework, an authenticity-ignorant adversary cannot distinguish the value of $v[B_j]$ from other valid values in $P_v^{B_j}$, given ranked results. Since an adversary without prior knowledge cannot validate the authenticity of any value in $P_v^{B_j}$, all values in $P_v^{B_j}$ are "valid" in the perspective of the adversary, no matter if they are generated by our framework or collected from real data in D. Therefore, we observe that $P_v^{B_j}$ can be formed by any values in V_j^B.

In order to achieve the privacy guarantee of ε,, we have to ensure that $|P_v^{B_j}| > (1/\varepsilon)$, $\forall v \in D$, and $\forall j \in \{1, \ldots, m'\}$. An intuitive algorithm to generate $P_v^{B_j}$ of size $1/\varepsilon$ is to randomly pick $(1/\varepsilon) - 1$ values from $V_j^B \backslash v[B_j]$. Specifically, let the initial $P_v^{B_j} = \{v[B_j]\}$. Then, we can randomly pick distinct values from $V_j^B \backslash v[B_j]$ and insert them into $P_v^{B_j}$ until $P_v^{B_j}$ contains at least $(1/\varepsilon)$ distinct values. In the same manner, we can construct a polymorphic value set for each tuple's each private attribute.

4.1.1. Privacy Guarantee. For a database D where every tuple v's every private attribute B_j is included by one polymorphic value set with virtual values (PVSV) whose size is at least l, if the adversary has no prior knowledge of D, a privacy level of $\varepsilon = (1/l)$ is achieved.

For an authenticity-ignorant adversary, $\mathrm{PK}(v \mid D) = 1$ $\forall v$. As proved in Privacy-Preserving Framework, for an authenticity-ignorant adversary, it is impossible to distinguish $v[B_j]$ with at least $l-1$ other values. Thus we have $g(v, B_j) \leq (1/l)$ for $\forall v \in D$ and $\forall j \in \{1, \ldots, m'\}$. According to equation (3), a privacy guarantee of $(1/l)$ can be achieved.

4.2. Utility Optimization. In this section, we discuss how to reduce utility loss caused by polymorphic value sets. We introduced a metric of utility loss in (4) that calculates the sum of difference in ranked results given all possible queries. To practically calculate utility loss, we limit the range of queries to a finite set named *query workload*. In practice, the query workload of a database D can be a set of queries that are more frequently issued than any other queries. A query workload may contain duplicate queries, which reflect the distribution of frequent queries. With a query workload, denoted as Q, we can define the practical utility loss as

$$U_Q = \sum_{q \in Q} \sum_{v \in D} \left| \mathrm{Rank}(v \mid q) - \mathrm{Rank}'(v \mid q) \right|. \tag{10}$$

In order to reduce utility loss, we have to find assignments of $P_v^{B_j}$ for $\forall v \in D$ and $\forall j \in \{1, \ldots, m'\}$ such that the overall U_Q can be minimized. Without loss of generality, we only consider constructing polymorphic value sets of size 2. In this case, the privacy guarantee is 1/2. For each $v[B_j]$, we need to find a value that is indistinguishable from $v[B_j]$. We denote the polymorphic value of $v[B_j]$ as $v'[B_j]$.

Definition 3. We define the 2-PVSV problem as follows: given database D and query workload Q, find a polymorphic value $v'[B_j]$ from $V_j^B \backslash \{v[B_j]\}$ for each $v[B_j]$, $\forall v \in D$ and $\forall j \in \{1, \ldots, m'\}$, such that U_Q defined in (10) is minimized.

Theorem 1. *The 2-PVSV problem is NP-hard.*

The proof of Theorem 1 in detail can be found in Appendix A.

4.3. Heuristic Algorithm. We have proved that the 2-PVSV problem is an NP-hard problem that may not be solved in polynomial time. Therefore, we propose PVSV-Constructor, a heuristic algorithm that can return an approximate solution in polynomial time.

We observe that $|s(v \mid q) - s'(v \mid q)|$ is relevant to $|\mathrm{Rank}(v \mid q) - \mathrm{Rank}'(v \mid q)|$. A smaller difference between $s(v \mid q)$ and $s'(v \mid q)$ leads to a smaller difference between $\mathrm{Rank}(v \mid q)$ and $\mathrm{Rank}'(v \mid q)$. Therefore, $|\mathrm{Rank}(v \mid q) - \mathrm{Rank}'(v \mid q)|$ can be approximately minimized by a solution that minimizes $|s(v \mid q) - s'(v \mid q)|$. As such, we propose an approximation of U_Q that calculates the score difference before and after adopting our framework. We denote the score difference as U_Q^S and define U_Q^S as follows:

$$U_Q^S = \sum_{q \in Q} \sum_{v \in D} \left| s(v \mid q) - s'(v \mid q) \right|. \tag{11}$$

As shown in Algorithm 1, given input database D, the number of public and private attributes m and m' respectively, query workload Q and privacy guarantee ϵ, PVSV-Constructor constructs an equivalent value set for each $v \in D$ and $j \in \{1, \ldots, m'\}$ that minimizes U_Q^S. Recall the definition of $s'(t \mid q)$ in (7), and we have

$$U_Q^S = \sum_{v \in D} \sum_{j=1}^{m'} \sum_{q \in Q} w_j^B \left| \rho'\left(q[B_j], v[B_j]\right) - \rho\left(q[B_j], v[B_j]\right) \right|. \tag{12}$$

We denote the score difference of $v[B_j]$ contributed by $P_v^{B_j}$ as $U(P_v^{B_j})$.

$$U\left(P_v^{B_j}\right) = \sum_{q \in Q} w_j^B \left| \rho'\left(q[B_j], v[B_j]\right) - \rho\left(q[B_j], v[B_j]\right) \right|. \tag{13}$$

Therefore, U_Q^S is the sum of $U(P_v^{B_j})$ over all tuples and private attributes:

$$U_Q^S = \sum_{v \in D} \sum_{j=1}^{m'} U\left(P_v^{B_j}\right). \tag{14}$$

Since the construction of each $P_v^{B_j}$ is independent from other polymorphic value sets, we can minimize U_Q^S by minimizing the score difference contributed by each $P_v^{B_j}$ for $\forall j \in \{1, \ldots, m'\}$ and $\forall v \in D$. Note that $|\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])| = 0$ if $q[B_j] = v[B_j]$ or $q[B_j] \notin P_v^{B_j}$. Also note that $|\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])| = 1$ when $q[B_j] \neq v[B_j]$ and $q[B_j] \in P_v^{B_j}$. Therefore, if $\forall q \in Q$, $v[B_j] = q[B_j]$, then any assignment of $P_v^{B_j}$ cannot contribute to a higher $U(P_v^{B_j})$ since $\rho'(q[B_j], v[B_j]) = 1$, $\rho(q[B_j], v[B_j]) = 1$, and $|\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])|$ is always zero. In this situation, $U(P_v^{B_j}) = 0$. On the contrary, if $\exists q \in Q$, $v[B_j] \neq q[B_j]$, then $\rho(q[B_j], v[B_j]) = 0$, and thus, $|\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])| = \rho'(q[B_j], v[B_j])$. According to equation (13), the value of $U(P_v^{B_j})$ is

$$\begin{aligned} &\sum_{q \in Q} w_j^B \left| \rho'\left(q[B_j], v[B_j]\right) - \rho\left(q[B_j], v[B_j]\right) \right| \\ &\quad = \sum_{q \in Q, q[B_j] \neq v[B_j]} w_j^B \rho'\left(q[B_j], v[B_j]\right). \end{aligned} \tag{15}$$

In order to minimize $U(P_v^{B_j})$, we have to find an assignment of $P_v^{B_j}$ which minimizes $|\{q \in Q \mid q[B_j] \neq v[B_j], \exists \beta \in P_v^{B_j}, q[B_j] = \beta\}|$. Consider the simplest case where we want to construct $P_v^{B_j}$ of size 2: $\{v[B_j], \beta\}$. We have

```
(i) Input: ε, D, m, m′, Q
(ii) Output: P_v^{B_j}, ∀v ∈ D and ∀j ∈ {1, ..., m′}
(1) k = (1/ε) − 1
(2) for t in D do
(3)     for j ∈ {1, ..., m′} do
(4)         P_v^{B_j} = {v[B_j]}
(5)         Set = V_j^B \ {v[B_j]}
(6)         Sort elements of Set by their frequencies in {q[B_j] | q ∈ Q} in ascending order.
(7)         Remove the last |Set| − k + 1 elements in Set.
(8)         P_v^{B_j} = P_v^{B_j} ∪ Set
(9)     end
(10) end
```

Algorithm 1: PVSV-Constructor.

$$\sum_{q\in Q, q[B_j]\neq v[B_j]} w_j^B \rho'\left(q[B_j], v[B_j]\right) = w_j^B \left|\left\{q \in Q \mid q[B_j] = \beta\right\}\right|. \tag{16}$$

In this case, β has to be a value in $V_j^B \backslash v[B_j]$ such that β has the lowest frequency among all $q[B_j]$ values for $q \in Q$. Note that β does not have to be an element of $\{q[B_j] \mid q \in Q\}$. If $\beta \notin \{q[B_j] \mid q \in Q\}$, its frequency is 0. Similarly, if we want to construct $P_v^{B_j}$ of size k, then we should insert the $k-1$ least frequent values among all $q[B_j]$ values into $P_v^{B_j}$.

In line 4 of Algorithm 1, we initialize the polymorphic value set of $v[B_j]$ by inserting $v[B_j]$ into $P_v^{B_j}$. For $v[B_j]$, a set *Set* is constructed from $V_j^B \backslash \{v[B_j]\}$, which contains all values that can be inserted into $P_v^{B_j}$. In order to minimize $U(P_v^{B_j})$, we insert the $k-1$ least frequent values from Set into $P_v^{B_j}$ by their frequencies in $\{q[B_j] \mid q \in Q\}$. The above construction of $P_v^{B_j}$ is repeated for each $t \in D$ and $j \in \{1, \ldots, m'\}$. The computational complexity of Algorithm 1 is $O(|D| \cdot |Q| \cdot m')$.

5. Framework with True Values

5.1. Authenticity-Knowledgeable Adversaries. As we mentioned in the adversary model, an authenticity-knowledgeable adversary is able to tell if $t \in D$ is possible. As a result, the authenticity-knowledgeable adversary can launch a more efficient attack on private attributes by examining the authenticity of values learned from R_A. We show a simple case where an authenticity-knowledgeable adversary breaks the privacy guarantee provided by polymorphic values sets constructed with virtual values. Consider that the objective of the adversary is to infer the value of $v[B_0]$ and B_0 is the only private attribute in D. $P_v^{B_0}$ is the polymorphic value set generated for $v[B_0]$ and $|P_v^{B_0}| = 1/\varepsilon$. Without prior knowledge, $P_{v[B_0]} = \varepsilon$ and the privacy guarantee is achieved. However, for a value $\beta \in P_v^{B_0} \backslash \{v[B_0]\}$, if the adversary can conclude that $\mathrm{PK}(v' \mid D) = 0$ for any tuple v' such that v' is equal to v in all public attributes and $v'[B_0] = \beta$, then the adversary can exclude β from $P_v^{B_0}$. Therefore,

$$g(v[B_0]) = \frac{\varepsilon}{1-\varepsilon} > \varepsilon, \tag{17}$$

and equation (9) no longer holds.

As shown above, if values in $P_v^{B_0}$ are marked by an adversary as invalid for v given $\mathrm{PK}(v' \mid D)$, then the adversary can successfully break the privacy guarantee defined in framework with virtual values.

5.2. Design. In this section, we propose polymorphic value sets with true values (PVST) that construct polymorphic value sets with values that cannot be excluded by $\mathrm{PK}(t \mid D)$. PVST considers nontrivial prior knowledge of adversaries and presents the same degree of privacy guarantee introduced in (9).

We have shown above that the implementation with virtual values can be compromised by adversaries with prior knowledge. Consider a tuple $v \in D$. Given privacy guarantee ϵ, we construct m' polymorphic values sets $P_v^{B_0}, \ldots, P_v^{B_{m'}}$ for each private attributes of v where $|P_v^{B_j}| \geq 1/\varepsilon$. Let set M_v^A be

$$M_v^A = \{v[A_0]\} \times \cdots \times \{v[A_m]\} \times P_v^{B_0} \times \cdots \times P_v^{B_{m'}}. \tag{18}$$

M_v^A is the Cartesian product of sets each of which contains v's all equivalent values in an attribute. For public attribute A_i, the corresponding set is $\{v[A_i]\}$ since public attribute values are open to the adversary. For private attribute B_j, the corresponding set is $P_v^{B_j}$. Therefore, M_v^A contains all possible tuples that are indistinguishable with v with respect to R_A (including v itself).

With prior knowledge on $\mathrm{PK}(t \mid D)$, an adversary is able to exclude a value β from $P_v^{B_j}$ if

$$\forall t \in \left\{t \mid t \in M_v^A, t\left[B_j\right] = \beta\right\}, \mathrm{PK}(t \mid D) = 0. \tag{19}$$

As described above, it is safe for the adversary to conclude that $\beta \neq v[B_j]$, if there is no $t \in M_v^A$ such that $t[B_j] = \beta$ and $\mathrm{PK}(t \mid D) = 1$. Alternatively, if for every β in $P_v^{B_j}$, there is a t such that $t[B_j] = \beta$ and $\mathrm{PK}(t \mid D) = 1$, then the adversary cannot exclude any value in $P_v^{B_j}$, and therefore, $P_{v[B_j]} \leq \varepsilon$ is guaranteed. Since $\mathrm{PK}(t \mid D) = 1$ iff $t \in D$, we construct polymorphic value sets with true values from $t \in D$.

Definition 4. If there exists l distinct values $\beta_1, \ldots, \beta_l \in P_v^{B_j}$ such that $\forall r \in \{1, \ldots, l\}$, β_r holds the following property:

$$\exists t \in M_v^{\mathrm{A}}, t\left[B_j\right] = \beta_r \text{ and } t \in D. \tag{20}$$

Then, we say that $P_v^{B_j}$ *covers* l true values. We denote $T_v^{B_j} = \{\beta_1, \ldots, \beta_l\}$ as the *true value set* of t in B_j.

Privacy guarantee: for a database D where every tuple's every private attribute is included by one equivalent value set which covers at least l true values, a privacy level of $\varepsilon = 1/l$ is achieved.

Assume that the adversary's objective is to infer the value of $v[B_j]$. As mentioned in the adversary model, we make no assumption on the attacking methods adopted by an adversary. Consider M_v^{A} defined in (18), $\forall t,\ t' \in M_v^A$ and $\forall q \in Q_{\mathrm{A}}, s(t \mid q) = s(t' \mid q)$. Therefore, the adversary cannot distinguish tuples in M_v^A by observing R_{A}. In this situation, the adversary would use $\mathrm{PK}(t \mid D)$ to exclude all tuple t in M_v^{A} such that $\mathrm{PK}(t \mid D) = 0$. However, as $P_v^{B_j}$ covers l true values, there exists l tuples $t_1, \ldots, t_l \in M_v^{\mathrm{A}}$ such that $\mathrm{PK}(t \mid D) = 1$ and $t_r[B_j] \neq t_s[B_j]$ for arbitrary $r, s \in \{1, \ldots, l\}$. As a result, values in $\left\{t_1[B_j], \ldots, t_l[B_j]\right\}$ are indistinguishable given R_{A} and $\mathrm{PK}(t \mid D)$. Thus, $P_{v[B_j]} = 1/l$. According to (3), a privacy level of $1/l$ is achieved.

5.3. Utility Optimization. An intuitive method of constructing $P_v^{B_j}$ is to insert the value of B_j of all tuples that share the same public attribute values with v into $P_v^{B_j}$, i.e., $P_v^{B_j} = P_v^{B_j} = \{v'[B_j] \mid \forall i \text{ and } v' \in D, v'[A_i] = v[A_i]\}$. The privacy guarantee is met if $P_v^{B_j}$ covers at least $1/\varepsilon$ true values. Nevertheless, utility loss cannot be ignored as in the intuitive method, $s(v' \mid q)$ would be the same for all $v' \in \{v' \mid \forall i$ and $v' \in D, v'[A_i] = v[A_i]\}$, and thus information of private attributes are missing in the ranked result of v'. Therefore, the size of $P_v^{B_j}$ is critical in balancing privacy and utility. With no loss of generality, we limit the size of each polymorphic value set to 2. We show that constructing such polymorphic value sets is an NP-hard problem.

Definition 5. We define the 2-PVST problem as follows: given database D and a query workload Q, construct $P_v^{B_j}$ of size 2 for each tuple $v \in D$, $\forall j \in \{1, \ldots, m'\}$, and minimize U_Q defined in (10).

Theorem 2. *The 2-PVST problem is NP-hard.*

The proof of Theorem 2 can be found in Appendix B.

5.4. Heuristic Algorithm. We have shown that the 2-PVST problem is NP-hard. In this subsection, we present PVST-Constructor, a heuristic algorithm that constructs PVST within polynomial time. PVST-Constructor tries to minimize $|P_v^{B_j}|$ and $\sum_{q \in Q} |s'(v \mid q) - s(v \mid q)|$ for each $v \in D$ and $j \in \{1, \ldots, m'\}$ with the greedy algorithm.

The pseudo-code of PVST-Constructor is shown in Algorithm 2. In lines 2 to 6, we initialize each $P_v^{B_j}$ with $\{v[B_j]\}$. Then, for each $v \in D$, PVST-Constructor constructs $P_v^{B_1}, \ldots, P_v^{B_m'}$ by finding a v' with the greedy algorithm and inserting $v'[B_j]$ into $P_v^{B_j}, \forall j \in \{1, \ldots, m'\}$. The above process will be taken multiple times until $|P_v^{B_j}| \geq 1/\varepsilon$. Since every v' is a real tuple existing in D and we insert at least k tuples in the above processes, $P_v^{B_j}$ covers at least $k+1$ true values, and thus, the privacy guarantee is met. As mentioned in Utility Optimization, the size of $P_v^{B_j}$ is critical in minimizing utility loss. Therefore, the heuristic algorithm tries to minimize the utility loss by minimizing the size of each $P_v^{B_j}$. We count the number of polymorphic value sets of v, if the set contains less than k values, that can be enlarged by inserting v''s private attribute values. We denote this count as $\mathrm{cover}_{v'}$:

$$\mathrm{cover}_{v'} = \left|\left\{j \mid \left|P_v^{B_j}\right| < k, v'\left[B_j\right] \notin P_v^{B_j}\right\}\right|. \tag{21}$$

Tuple v' with a higher $\mathrm{cover}_{v'}$ value can enlarge the size of more polymorphic value sets of v, and thus, we can reduce the number of tuples that we have to insert into $P_v^{B_1}, \ldots, P_v^{B_m'}$.

Furthermore, we take into consideration queries in Q. In order to minimize U_Q, we have to minimize $|s'(v \mid q) - s(v \mid q)|$ for $q \in Q$. Note that for each attribute B_j, $|\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])| = 1$ if $v[B_j] \neq q[B_j]$ and $q[B_j] \in P_v^{B_j}$. We denote the value of $\sum_j |\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])|$ as $\mathrm{loss}_{v'}$. Thus, we have

$$\begin{aligned}\mathrm{loss}_{v'} &= \sum_{q \in Q} \sum_{j=1,\ldots,m'} \left|\rho'\left(q\left[B_j\right], v\left[B_j\right]\right) - \rho\left(q\left[B_j\right], v\left[B_j\right]\right)\right| \\ &= \sum_{q \in Q} \sum_{j=1,\ldots,m'} \delta(v, v', q, j),\end{aligned} \tag{22}$$

where $\delta(v, v', q, j) = 1$ if $v[B_j] \neq q[B_j]$ and $v'[B_j] = q[B_j]$ and $\delta(v, v', q, j) = 0$ otherwise. Therefore, tuple v' with a smaller $loss_{v'}$ can reduce the value of $\sum_j |\rho'(q[B_j], v[B_j]) - \rho(q[B_j], v[B_j])|$, and thus, we can reduce the value of U_Q.

The computation of $\mathrm{cover}_{v'}$ and $\mathrm{loss}_{v'}$ is done in line 11. Then, we compute $\mathrm{score}_{v'}$, the score of v' that indicates how preferable v' is, relative to other tuples in $D \backslash \{v\}$. We adopt the greedy algorithm to find the next v' for $P_v^{B_1}, \ldots, P_v^{B_{m'}}$, i.e., in each iteration, we choose the tuple that has the highest score value. In line 15, we insert the private attribute values of the chosen tuple (denoted as $v_{\max}$) into $P_v^{B_1}, \ldots, P_v^{B_{m'}}$. The above process is repeated until the sizes of $P_v^{B_1}, \ldots, P_v^{B_{m'}}$ are no less than k.

6. Experimental Results

6.1. Experimental Setup. To validate PVSV-Constructor and PVST-Constructor algorithms, we conducted experiments on a real world dataset [29] from eHarmony which contains 58 attributes and 486,464 tuples. We removed 5 noncategorical attributes and randomly picked 20 categorical attributes from the remaining 53 attributes. The domain sizes of the 20 attributes range from 2 to 15. After removing duplicate tuples, we randomly picked 300,000 tuples as our testing bed.

By default, we use the ranking function from the ranked retrieval model with all weights set to 1. All experimental results were obtained on a Mac machine running Mac OS

(i) **Input**: ε, D, m, m', Q
(ii) **Output**: $P_v^{B_j}$, $\forall v$ and $\forall B_j$
(1) $k = (1/\varepsilon) - 1$
(2) **for** v in D **do**
(3) **for** $j \in \{1, \ldots, m'\}$ **do**
(4) $P_v^{B_j} = \{v[B_j]\}$
(5) **end**
(6) **end**
(7) **for** v in D **do**
(8) **while** $\exists j \in \{1, \ldots, m'\}, |P_v^{B_j}| < k + 1$ **do**
(9) **for** $v' \in \{t \in D | \forall A_i, t[A_i] = v[A_i]\}$ **do**
(10) compute $\text{cover}_{v'}$, $\text{loss}_{v'}$
(11) $\text{score}_{v'} = \text{cover}_{v'} = 1/1 + \exp(-\text{loss}_{v'}) =$
(12) **end**
(13) $v_{\max} = \text{argmax}_{v' \in \{|t \in D | \forall A_i, t[A_i] = v[A_i]\}} \text{score}_{v'}$
(14) **for** $j \in \{1, \ldots, m'\}$ **do**
(15) $P_v^{B_j} = P_v^{B_j} \cup \{t_{\max}[B_j]\}$
(16) **end**
(17) **end**
(18) **end**

Algorithm 2: PVST-Constructor.

with 8 GB of RAM. The algorithms were implemented in *Python*.

6.2. Privacy. The privacy guarantee of PVSV-Constructor and PVST-Constructor were tested by performing Ranked Inference attack [35], including Point-Query, In-Query, Point-Query&Insert, and In-Query&Insert attacking methods, on the dataset. From a total of 20 attributes, 5 attributes were randomly chosen as public attributes and another 5 attributes were randomly chosen as private attributes. We randomly picked 20,000 distinct tuples from the dataset as the testing bed and randomly generated 10 tuples as the query workload. For PVSV-Constructor, we constructed a polymorphic value set of size 2 for each private attribute and each tuple in the testing bed. For PVST-Constructor, we constructed a polymorphic value set that covers at least two true values for each private attribute and each tuple. We randomly picked 1,000 tuples from the testing bed as our targets and performed 1,000 Rank Inference attacks (250 attacks for each of the four methods) on the five private attributes of target tuples. We measured the attack success guess rates based on the frequency of successful inference among all inference attempts. Figure 1 shows the success guess rates of Rank Inference attacks on the unprotected testing bed, the testing bed with PVSV, and the testing bed with PVST. As the size of each polymorphic value set is 2, the success guess rates on PVSV are around 50%, which are significantly lower than those of unprotected dataset. We also observe that the success guess rates on PVST are slightly lower than those of PVSV. The reason is that PVSV-Constructor will be inserting values into tuple v's polymorphic value sets until all v's polymorphic value sets cover at least 2 true values. Thus, some polymorphic value sets of v may contain more than 2 values.

6.3. Utility. In this subsection, we quantify utility loss of PVSV-Constructor and PVST-Constructor algorithms. The privacy guarantee of both PVSV-Constructor and PVST-Constructor is 1/2, i.e., the polymorphic value sets constructed by PVSV-Constructor contains 2 values, and the polymorphic value sets constructed by PVST-Constructor contains at least 2 true values. The key parameters here are the size of query workload Q, the size of database D, the number of public and private attributes, and the weight ratios in the ranking function. We randomly generated 20 tuples as the query workload. By default, we picked 10 tuples from Q and set $|D| = 300,000$, $m = 10$, and $m' = 10$. Therefore, we randomly picked 10 attributes from the testing bed and set them as public attributes. The rest of the 10 attributes were set as private attributes.

Many recommendation systems of ONS applications feature top-k recommendation [1, 14, 47] where the ranked result contains a set of k tuples that will be of interest to a certain user, as it is impractical and unnecessary to return all tuples in the database to the user. Therefore, we introduce average top-k utility loss U_{aul}, a variant of U_Q that focuses on utility loss of the top-k tuples in a ranked result. U_{aul} is defined as

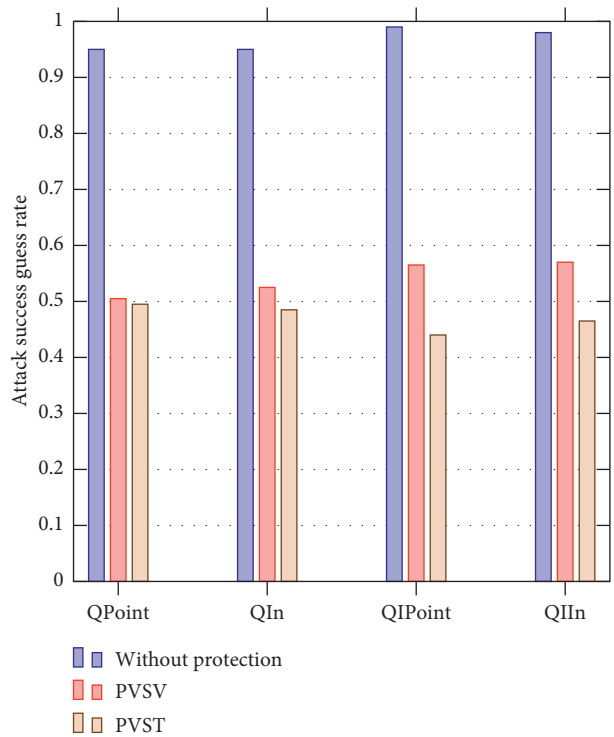


Figure 1: Attack success guess.

$$U_{aul} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{|D'|}\sum_{v\in D'}\frac{\left|\min\{\mathrm{Rank}(v\mid q), k+1\} - \min\{\mathrm{Rank}'(v\mid q), k+1\}\right|}{k}, \tag{23}$$

where $D' = \{v \in D \mid \mathrm{Rank}(v\mid q)) < k \text{ or } \mathrm{Rank}(v\mid q)) < k\}$. Intuitively, U_{aul} represents the average percentage rank difference relative to k over all queries and all tuples in top-k. U_{aul} is equivalent to $U_Q/|Q||D|^2$ when $k = |D|$. By default, we set $k = 100$.

6.3.1. Evaluation of U_{aul} with Varying k. We first discuss the average top-k utility loss of PVSV-Constructor, PVST-Constructor, and a baseline algorithm on varying k with other parameters set to default values. For a tuple v's attribute B_j, the baseline algorithm constructs $P_v^{B_j}$ with $v[B_j]$ and a value randomly picked from $V_j^B \backslash \{v[B_j]\}$. The results are presented in Figure 2, which shows that the U_{aul} of PVSV-Constructor is significantly lower than that of the baseline algorithm. The U_{aul} of PVST-Constructor is also lower than that of the baseline algorithm when $k \geq 5$, even though the baseline algorithm cannot preserve privacy against authenticity-knowledgeable adversaries. The experimental results show that both PVSV and PVST can reduce utility loss with respect to rank differences. Also, note that PVST-Constructor constructs polymorphic value sets with true values, and thus, from Figure 3, we can see that some polymorphic value sets constructed by PVST-Constructor contains more than 2 values.

6.3.2. Evaluation of U_{aul} with Varying Sizes of Q. Figure 4 presents the average top-k utility loss of PVSV-Constructor on varying $|Q|$. When $|Q|$ is set to 1, 5, or 10, we randomly picked 1, 5, or 10 queries from the original Q, respectively. With increasing number of queries in the query workload, the U_{aul} of PVSV-Constructor increases monotonically. The reason is that PVSV-Constructor always generates the least frequent value (denoted as $v'[B_j]$) in $\{q[B_j] \mid q \in Q\}$ for $v[B_j]$. If $|Q|$ is small, then it is possible that $v'[B_j] \notin \{q[B_j] \mid q \in Q\}$, and thus, $s(v \mid q) = s'(v \mid q)$ $\forall q \in Q$. However, a larger query workload covers more private attributes values, and thus, it will be harder for PVSV-Constructor to generate a value for $v[B_j]$ that has no impact on the rank of v for any $q \in Q$. Figure 5 presents the average top-k utility loss of PVST-Constructor on varying $|Q|$. We can see that the size of Q has no significant impact on the U_{aul} of PVST-Constructor, as the PVST-Constructor always pick a tuple v' that is different from v in most attributes and then insert $v'[B_j]$ into $P_v^{B_j}$.

6.3.3. Evaluation of U_{aul} with Varying Sizes of the Dataset. Figures 6 and 7 depict the impact of the size of datasets on U_{aul} of PVSV-Constructor and PVST-Constructor. Datasets of 100,000 and 200,000 tuples were randomly sampled from

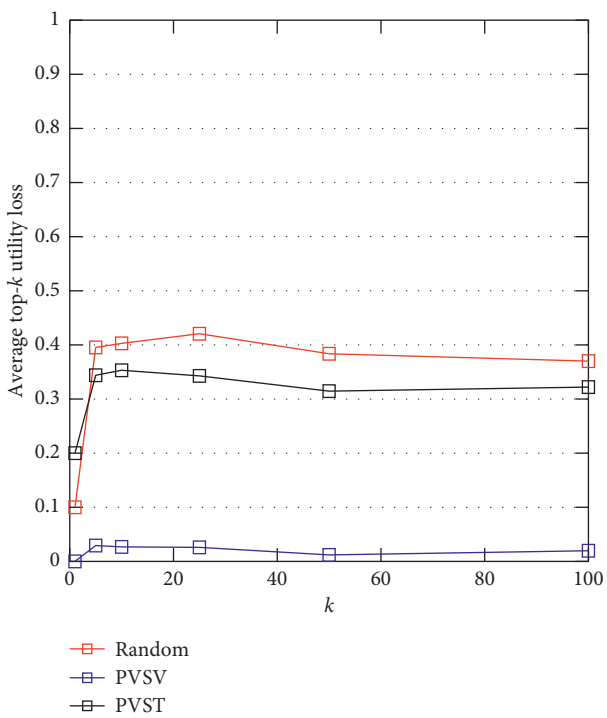


Figure 2: Utility loss vs. k.

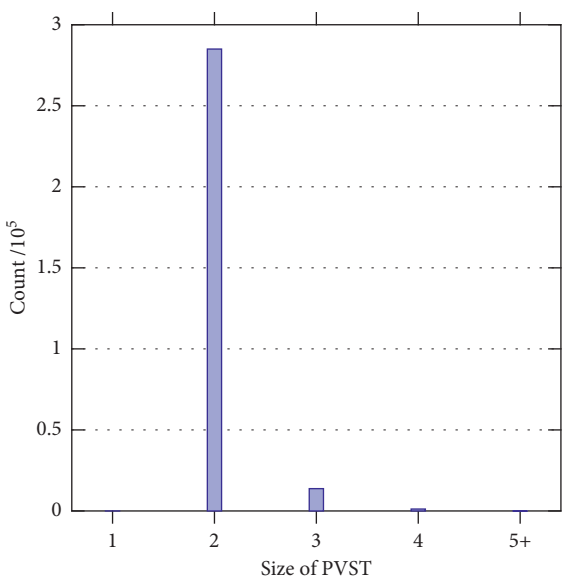


Figure 3: Size of polymorphic value set (PVST).

the testing bed of 300,000 tuples. As expected, $|D|$ has no significant impact on the U_{aul} of PVSV-Constructor since PVSV-Constructor generates values from the domain of each private attributes. $|D|$ has no impact on the U_{aul} of PVST-Constructor, which indicates that a dataset containing 100,000 tuples is sufficient for PVST-Constructor to generate polymorphic value sets with true values.

6.3.4. Evaluation of U_{aul} with Varying m. We investigate the impact of the number of private and public attributes on

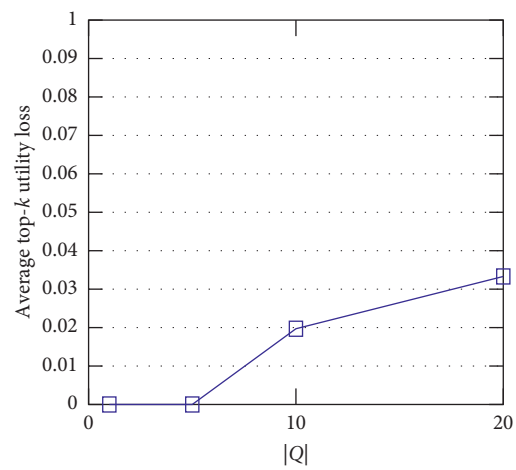


Figure 4: Utility loss vs. $|Q|$.

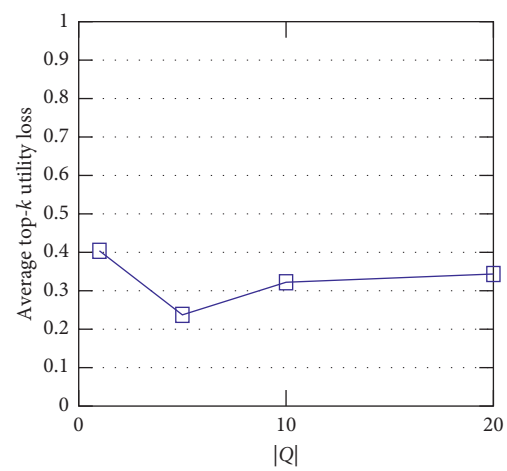


Figure 5: Utility loss vs. $|Q|$.

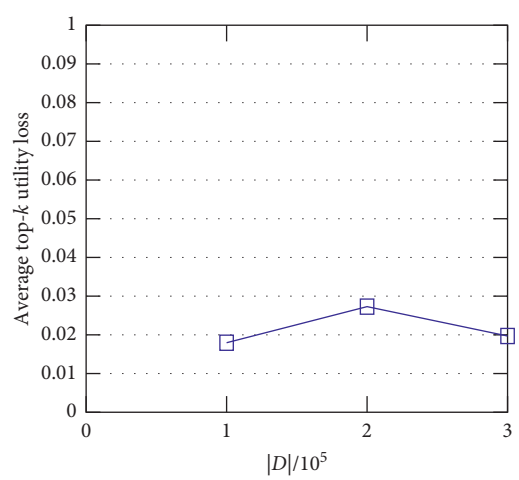


Figure 6: Utility loss vs. $|D|$.

average top-k utility loss. Figure 8 presents the U_{aul} of PVSV-Constructor with fixed m' and varying m and with fixed m and varying m'. When m/m' is set to 5, we randomly removed 5 attributes from the testing bed. When m/m' is set to

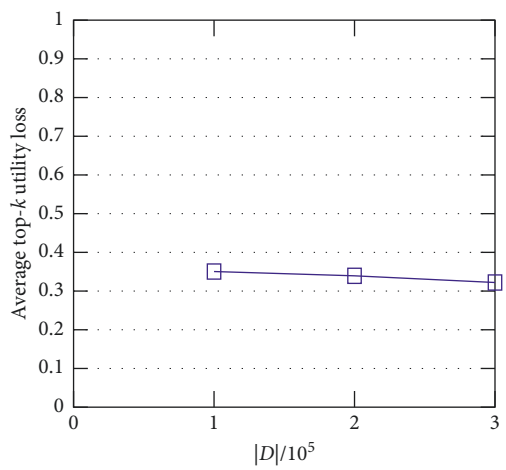


Figure 7: Utility loss vs. $|D|$

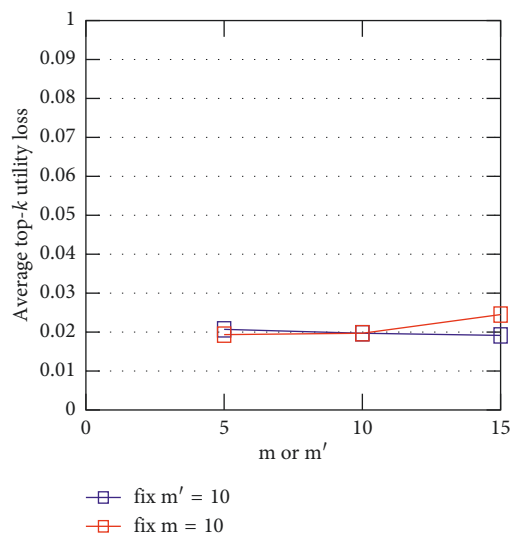


Figure 8: Utility loss vs. m and m'.

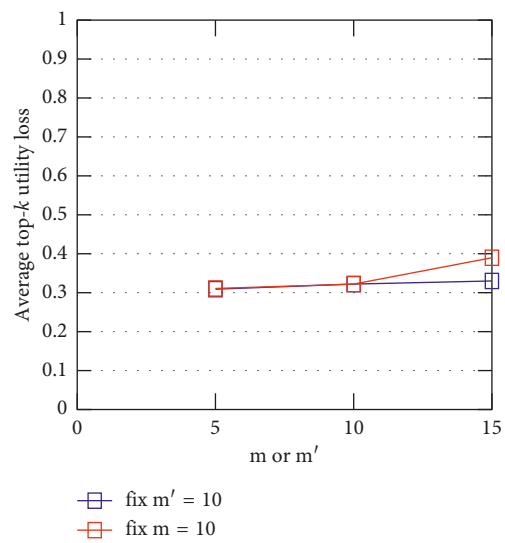


Figure 9: Utility loss vs. m and m'.

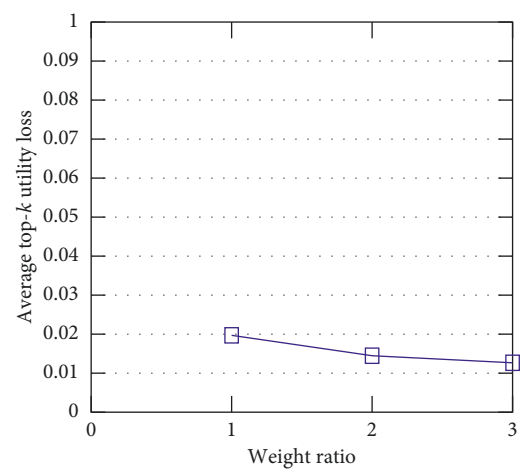


Figure 10: Utility loss vs. weight ratio.

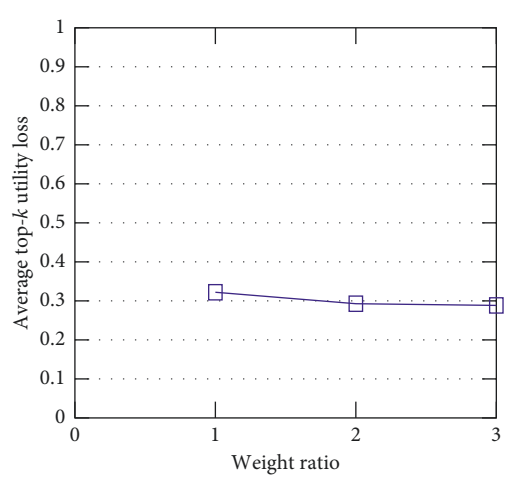


Figure 11: Utility loss vs. weight ratio.

15, we added 5 more categorical attributes randomly chosen from the unused attributes. We observe that the U_{aul} monotonically decreases with increasing number of public attributes and monotonically increases with increasing number of private attributes. As expected, a higher proportion of public attributes leads to less variant between $s(v \mid q)$ and $s'(v \mid q)$. The results of the same experiment with PVST-Constructor are shown in Figure 9. U_{aul} increases as increasing number of private attributes as expected. However, U_{aul} also increases slightly with increasing number of public attributes. This is due to the fact that with more public attributes, there will be few tuples that share the same public attribute values. Since PVST-Constructor inserts only private attribute values from tuples sharing same public attribute values, more values will be inserted to each polymorphic value set, which introduces higher utility loss.

6.3.5. Evaluation of U_{aul} with Varying Weight Ratios. Figures 10 and 11 illustrate the impact of weight ratios on the average utility loss. The experiment was conducted with a fixed private attribute weight of 1 and varying public attribute weights of 1, 2, and 3. As expected, U_{aul} of both PVSV-Constructor and PVST-Constructor decreases as

increasing weight ratio of public attributes. The reason is that as the public attribute weight increases, the part in $s'(v \mid q)$ caused by private attributes decreases. Therefore, less impact would be made to $s'(v \mid q)$ by PVSV and PVST. As a result, $s'(v \mid q)$ would be closer to $s(v \mid q)$ and the utility loss could be decreased.

7. Conclusions

In this paper, we proposed a novel framework that preserves privacy of private attributes against arbitrary attacks through the ranked retrieval model. Furthermore, we identify two categories of adversaries based on varying adversarial capabilities. For each kind of adversaries, we presented implementation of our framework. Our experimental results suggest that our implementations efficiently preserve privacy against Rank Inference attack [35]. Moreover, the implementations significantly reduce utility loss with respect to the variance in ranked results.

It is our hope that this paper can motivate further research in privacy preservation of SIoT with consideration of social network features and/or variant information retrieval models, e.g., text mining.

Appendix

A. 2-PVSV

In this subsection, we prove that constructing an optimal PVSV for each attribute of a tuple v is an NP-hard problem.

Definition A.1. For a tuple v in D, we create $P_v^{B_j}$ for each B_j. We say v satisfies query q if the following hold for any tuple t $(t \neq v)$ in D: (1) If $s(v \mid q) < s(t \mid q)$, then $s'(v \mid q) < s'(t \mid q)$, and (2) if $s(v \mid q) \geq s(t \mid q)$, then $s'(v \mid q) \geq s'(t \mid q)$

For a database containing 2 tuples, the 2-PVSV problem can be redefined as follows: given a query workload Q, a database D, construct $P_v^{B_1}, \ldots, P_v^{B_m'}$ of size 2 such that v satisfies the most queries in Q.

Definition A.2. Max-3Sat Problem: given a 3-CNF formula Φ, find the truth assignment that satisfies that most clauses.

Lemma A.1. *Max-3Sat $\leq_P$ 2-PVSV Problem.*

Proof. We construct a reduction function $f(\Phi) = (D, s, Q)$ which takes a Max-3Sat instance as input and returns a 2-PVSV instance. Without loss of generality, we suppose that Φ is a conjunction of l clauses and each clause is a disjunction of 3 literals from set $X = \{x_1, \ldots, x_n\}$. We construct database D as follows: D has 0 public attributes and $n+1$ private attributes $B_1, \ldots, B_n, C_1$. Let V_j^B be the attribute domain of B_j, $j \in \{1, \ldots, n\}$ and V_1^C be the attribute domain of C_1. Let $V_i^B = \{-1, 0, 1, 2, \ldots, n+2\}$ and $V_1^C = \{0, 1\}$. Two tuples, v_1 and v_2, are inserted into D: $v_1[B_j] = 0$ and $v_2[B_j] = j+2$ for $j \in \{1, \ldots, n\}$, while $v_1[C_1] = 0$ and $v_2[C_1] = 1$ We simplify the score function defined in (1) by setting all weights to 1. Also, note that $\rho(q[B_j], t[B_j]) = 0$, if $q[B_j]$ is null.

We construct query workload Q based on Φ. For each clause $L_k \in \Phi$, we construct a query q_k such that $q_k[B_i] = i$ iff the corresponding literal x_i is a positive literal in L_k, and $q[B_i] = i+1$ iff x_i is a negative literal in the clause. For example, given a clause $(x_1 \vee x_2 \vee x_3)$, the corresponding query q should satisfy $q[B_1] = 1, q[B_2] = 3, q[B_3] = 3$, and $q[C_1] = 0$. All other attributes in q are set to *null* by default. Therefore, $s(v_1 \mid q) = 1$ and $s(v_2 \mid q) = 0$, $\forall q \in Q$.

Since $s(v_2 \mid q) < s(v_1 \mid q)$ $\forall q \in Q$, in order to minimize utility loss, the value of $s'(v_2 \mid q)$ should be as small as possible. We observe that the minimum possible $s'(v_2 \mid q)$ is 1 because $P_{v_2}^{C_1}$ must be $\{0, 1\}$ as $V_1^C = \{0, 1\}$. Without loss of generality, we assume that we have already constructed $P_{v_2}^{B_1}, \ldots, P_{v_2}^{B_m'}, P_{v_2}^{C_1}$ for v_2 such that $s'(v_2 \mid q) = 1$, $\forall q \in Q$.

Now, we have an instance of 2-PVSV problem that given D, Q, constructs $P_{v_1}^{B_1}, \ldots, P_{v_1}^{B_m'}, P_{v_1}^{C_1}$ that satisfies the most queries in Q. Since $V_1^C = \{0, 1\}$, $P_{v_1}^{C_1}$ must be $\{0, 1\}$ too as $P_{v_1}^{C_1}$ contains two distinct values. Therefore, for an arbitrary query $q \in Q$, we have $\rho'(q[C], v_1[C]) = 1$ and $\rho'(q[C], v_2[C]) = 1$. In order to let v_1 satisfy q, we have to ensure that $s'(v_1 \mid q) > s'(v_2 \mid q) = 1$. Thus, we have

$$\begin{aligned} & s'(v_1 \mid q) > 1, \\ & \Longleftrightarrow \sum_{i=1}^{m'} \rho'(q[B_i], v_1[B_i]) > 1, \\ & \Longleftrightarrow \exists i \in \{1, \ldots, m'\}, \rho'(q[B_i], v_1[B_i]) = 1, \\ & \Longleftrightarrow \exists i \in \{1, \ldots, m'\}, q[B_i] \in P_{v_1}^{B_i}. \end{aligned} \tag{A.1}$$

Now, we show that the solution of 2-PVSV problem constructed above can answer the corresponding Max-3Sat Problem. Suppose that we have the solution $P_{v_1}^{B_1}, \ldots, P_{v_1}^{B_m'}$ such that v_1 satisfies the most queries in Q. As in (A.1), if v_1 satisfies q_k, we have $q_k[B_i] \in P_{v_1}^{B_i}$. If v_1 does not satisfy q_k, then $q_k[B_i] \notin P_{v_1}^{B_i}$. Recall that we assign value i or $i+1$ to $q_k[B_i]$ if x_i is a positive or negative literal in clause L_k, respectively. Therefore, the assignment of x_i given $P_{v_1}^{B_i}$ is

$$\begin{aligned} x_i &= \text{true}, \quad \text{if } i \in P_{v_1}^{B_i}, \\ x_i &= \text{false}, \quad \text{if } i+1 \in P_{v_1}^{B_i}. \end{aligned} \tag{A.2}$$

Furthermore, from equation (A.1), we have

$$s'(v_1 \mid q) > 1 \Longleftrightarrow L = \text{true}, \tag{A.3}$$

where L is q's corresponding clause in Φ. Note that $\{i, i+1\} \not\subset Q$ as $|P_{v_1}^{B_i}| = 2$ and $0 \in P_{v_1}^{B_i}$. Thus, the value of x_i is either true or false.

As we proved above, v_1 satisfies q_i if and only if L_i is true given assignment $\{x_1, \ldots, x_l\}$ constructed according to equation (A.2). Therefore, $\{x_1, \ldots, x_l\}$ satisfies the most clauses in ϕ if and only if v_1 satisfies the most queries in Q.

We now prove that function f can be conducted in polynomial time. Given a formula Φ with n variables and l clauses, we construct a 2-PVSV instance with 2 tuples each

of which has $n+1$ attributes and l queries each of which has 4 attributes. Therefore, $O(2n+4l)$ assignments are needed and f can be conducted in polynomial time. □

Proof of Theorem 1. We now prove that the 2-PVSV problem is NP-hard. In Lemma 3 we proved that Max-3Sat Problem can be reduced to the 2-PVSV problem in polynomial time. Furthermore, as Max-3Sat Problem is a NP-hard problem [33], the 2-PVSV problem is NP-hard. □

B. 2-PVST

In this section, we prove that constructing an optimal PVST for each private attribute of a tuple $v \in D$ is NP-hard. We use the definition of v satisfying q from (9). The 2-PVST problem can be redefined as follows.

Definition B.1. 2-PVST problem: given a query workload Q, a database D, the optimization problem of 2-PVST is to construct an arbitrary tuple v's polymorphic sets, $P_v^{B_1}, \ldots, P_v^{B'_m}$, that satisfies the most queries in Q. The size of each polymorphic vale set is 2.

Lemma B.1. *Max-3Sat $\leq_P$ 2-PVST problem.*

Proof. We construct a reduction function $f(\Phi) = (D, s, Q)$ which takes a Max-3Sat instance as input and returns a 2-PVST instance. We assume that Φ is a conjunction of l clauses and each clause is a disjunction of 3 literals from set $X = \{x_1, \ldots, x_n\}$. Let D have no public attribute and $n+1$ private attributes $B_1, \ldots, B_n, C_1$. For each literal x_i, we insert two tuples, v_i and v'_i, into D where

$$\begin{aligned} &v_i[C_1] = 1, v_i[B_i] = 1, v_i[B_j] = -1\ \forall j \neq i, \\ &v'_i[C_1] = 1, v'_i[B_i] = -1, v'_i[B_j] = 1\ \forall j \neq i. \end{aligned} \tag{B.1}$$

Then, we insert tuple v into D where $v[C_1] = 0$ and $v[B_j] = 0\ \forall j \in \{1, \ldots, m'\}$. We also set the domain of each B_j as $\{-1, 0, 1\}$ and the domain of C_1 as $\{0, 1\}$.

Without loss of generality, the score function defined in (1) is simplified by setting all weights to 1.

Next, we construct query workload Q. For each clause $L_k \in \Phi$, we construct a query q_k in which $q_k[B_j] = -1$ iff x_j is a positive literal in L_k, and $q_k[B_j] = 1$ iff x_j is a negative literal in L_k. We also set $q_k[C_1] = 1$. The rest of the attribute values are set to *null* by default. Therefore, if $L_k = (x_1 \vee \neq x_2 \vee x_3)$, then we have $q_k[C_1] = 1$, $q_k[B_1] = 1$, $q_k[B_2] = 1$, $q_k[B_3] = -1$, and $q_k[B_j] = \text{null}$ for $j \in \{4, \ldots, m'\}$.

Note that $\rho(q[B_j], t[B_j]) = 0$ if $q[B_j]$ is null. Thus, $s(v, q_k) = 0\ \forall q \in Q$ and $s(v_i, q_k)$, $s(v'_i, q_k) \geq 1\ \forall q \in Q$. We observe that for $q \in Q$ and $i \in \{1, \ldots, n\}$, $s(v \mid q) < s(v_i \mid q)$ and $s(v \mid q) < s(v'_i \mid q)$. In order to reduce U_Q, we have to maximize the number of $q \in Q$ such that $s'(v \mid q) < s'(v_i \mid q)$ and $s'(v \mid q) < s'(v'_i \mid q)$. The maximum value of $s'(v_i \mid q)$ and $s'(v'_i \mid q)$ is 4, which can be achieved by inserting $v_i[B_j]$ into $P_{v'_i}^{B_j}$ and inserting $v'_i[B_j]$ into $P_{v_i}^{B_j}$. Since $P_v^{C_1} = \{0, 1\}$, the value of $s'(v \mid q)$ is at least 1. Therefore, we have

$$\begin{aligned} &s'(v \mid q) < s'(v_i \mid q), \\ &\Longleftrightarrow s'(v \mid q) < 4, \\ &\Longleftrightarrow \sum_{j=1}^{m'} \rho'\left(q[B_j], v[B_j]\right) < 3, \\ &\Longleftrightarrow \exists j \in \{1, \ldots, m'\},\ \rho'\left(q[B_j], v[B_j]\right) = 0, \\ &\Longleftrightarrow \exists j \in \{1, \ldots, m'\},\ v[B_j] \neq q[B_j], \\ &\Longleftrightarrow \exists j \in \{1, \ldots, m'\},\ -q[B_j] \in P_v^{B_j}. \end{aligned} \tag{B.2}$$

Since $|P_v^{B_j}|$ is limited to 2, $P_v^{B_j} = \{0, 1\}$ or $\{0, -1\}$, i.e., we must insert either v_i's or v'_i's private attribute values into $P_v^{B_1}, \ldots, P_v^{B_m}$. Recall that we assign -1 or 1 to $q_k[B_j]$ if x_j is positive or negative, respectively, in L_k. In order to reduce Max-3Sat to 2-PSVT, we assign literals in the following way:

$$\begin{aligned} &x_j = \text{true}, \quad \text{if } -1 \in P_v^{B_j}, \\ &x_j = \text{false}, \quad \text{if } 1 \in P_v^{B_j}. \end{aligned} \tag{B.3}$$

Therefore, we denote the corresponding clause of q as L, and from equation (B.2), we have

$$\begin{aligned} &\exists j \in \{1, \ldots, m'\},\ -q[B_j] \in P_v^{B_j}, \\ &\Longleftrightarrow \exists j \in \{1, \ldots, m'\}, x_j = \text{true} \quad \text{if } x_j \text{ is positive in } L, \\ &\text{or } x_j = \text{false if } x_j \text{ is negative in } L, \\ &\Longleftrightarrow L = \text{true}. \end{aligned} \tag{B.4}$$

From the above equation, we observe that v satisfying q_k is equivalent to L_k being true. Therefore, we can reduce Max-3Sat to 2-PVST. Given a solution to the 2-PVST problem that maximizes the number of queries satisfied by v, assignment $\{x_1, \ldots, x_n\}$ produced by equation (B.3) can satisfy the largest number of clauses in Φ.

Given a Max-3Sat instance Φ, the construction of the 2-PVST instance can be conducted in polynomial time as we construct n queries with 3 attributes and $2n+1$ tuples with n attributes. The transformation from the optimal solution of 2-PVST to the optimal solution of Max-3Sat also takes polynomial time as we assign the value to each one of $x_1, \ldots, x_n$ once. Therefore, f is a polynomial time function. □

Proof of Theorem 2. In 4, we proved that a Max-3Sat instance can be reduced to a 2-PSVT instance in polynomial time. Furthermore, as Max-3Sat Problem is an NP-hard problem, the 2-PVSV problem is an NP-hard problem. □

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was partially supported by the National Science Foundation (grant nos. 0852674, 0915834, 1117297, and 1343976) and the Army Research Office (grant no. W911NF-15-1-0020).

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[2] C. C. Aggarwal and S. Y. Philip, "A condensation approach to privacy preserving data mining," in *Proceedings of the International Conference on Extending Database Technology*, pp. 183–199, Springer, Heraklion, Crete, Greece, March 2004.

[3] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29, no. 2, pp. 439–450, ACM, 2000.

[4] A. Alcaide, E. Palomar, J. Montero-Castillo, and A. Ribagorda, "Anonymous authentication for privacy-preserving IoT target-driven applications," *Computers & Security*, vol. 37, pp. 111–123, 2013.

[5] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social internet of things (SIoT)—when social networks meet the internet of things: concept, architecture and network characterization," *Computer Networks*, vol. 56, no. 16, pp. 3594–3608, 2012.

[6] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[7] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 24, p. 1, 2018.

[8] Z. Cai, X. Zheng, and J. Yu, "A differential-private framework for urban traffic flows estimation via taxi companies," *IEEE Transactions on Industrial Informatics*, vol. 17, 2019.

[9] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, 2014.

[10] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, pp. 442–455, Springer, New York, NY, USA, June 2005.

[11] C. Chen, X. Zhu, P. Shen et al., "An efficient privacy-preserving ranked keyword search method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 4, pp. 951–963, 2016.

[12] K. Chen and L. Liu, "Privacy preserving data classification with rotation perturbation," in *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05)*, p. 4, IEEE, Houston, TX, USA, November 2005.

[13] V. Ciriani, S. D. C. Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "Fragmentation and encryption to enforce privacy in data storage," in *Proceedings of the Fifth European symposium on research in computer security*, pp. 171–186, Springer, Dresden, Germany, September 2007.

[14] M. Deshpande and G. Karypis, "Item-based top-*N* recommendation algorithms," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 143–177, 2004.

[15] S. D. C. di Vimercati, R. F. Erbacher, S. Foresti, S. Jajodia, G. Livraga, and P. Samarati, "Encryption and fragmentation for data confidentiality in the cloud," in *Foundations of Security Analysis and Design VII*, A. Aldini, J. Lopez, and F. Martinelli, Eds., pp. 212–243, Springer, Berlin, Germany, 2013.

[16] S. D. C. di Vimercati, S. Foresti, G. Livraga, S. Paraboschi, and P. Samarati, "Confidentiality protection in large databases," in *A Comprehensive Guide through the Italian Database Research over the Last 25 Years*, pp. 457–472, Springer, Berlin, Germany, 2018.

[17] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[18] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2014.

[19] S. E. Fienberg and J. McIntyre, "Data swapping: variations on a theme by dalenius and reiss," in *Privacy in Statistical Databases*, pp. 14–29, Springer, Berlin, Germany, 2004.

[20] Y. Gao, T. Yan, and N. Zhang, "A privacy-preserving framework for rank inference," in *Proceedings of the IEEE Symposium on Privacy-Aware Computing (PAC)*, pp. 180-181, IEEE, Washington DC, USA, August 2017.

[21] Z. He, Z. Cai, and J. Yu, "Latent-data privacy preserving with customized data utility for social network data," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 1, pp. 665–673, 2017.

[22] T. Hong, S. Mei, Z. Wang, and J. Ren, "A novel vertical fragmentation method for privacy protection based on entropy minimization in a relational database," *Symmetry*, vol. 10, no. 11, p. 637, 2018.

[23] X. Huang, R. Fu, B. Chen, T. Zhang, and A. Roscoe, "User interactive internet of things privacy preserved access control," in *Proceedings of the International Conference for Internet Technology and Secured Transactions*, pp. 597–602, IEEE, London, UK, December 2012.

[24] Y. Huo, X. Fan, L. Ma, X. Cheng, Z. Tian, and D. Chen, "Secure communications in tiered 5G wireless networks with cooperative jamming," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3265–3280, 2019.

[25] K. Liu, H. Kargupta, and J. Ryan, "Random projection-based multiplicative data perturbation for privacy preserving distributed data miningin latex," *IEEE Transactions on knowledge and Data Engineering*, vol. 18, no. 1, pp. 92–106, 2005.

[26] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: privacy beyond k-anonymity and l-diversity," in *Proceedings of the IEEE 23rd International Conference on Data Engineering*, pp. 106–115, IEEE, Istanbul, Turkey, April 2007.

[27] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity," *ACM Transactions on Knowledge Discovery from Data*, vol. 1, no. 1, pp. 3–es, 2007.

[28] S. Matwin, "Privacy-preserving data mining techniques: survey and challenges," in *Studies in Applied Philosophy, Epistemology and Rational Ethics*, pp. 209–221, Springer, Berlin, Germany, 2013.

[29] B. McFee and G. Lanckriet, "Metric learning to rank," in *Proceedings of the 27th International Conference on Machine Learning (ICML'10)*, Haifa, Israel, June 2010.

[30] K. Muralidhar and R. Sarathy, "Data shuffling—a new masking approach for numerical data," *Management Science*, vol. 52, no. 5, pp. 658–670, 2006.

[31] J. Nin, J. Herranz, and V. Torra, "Rethinking rank swapping to decrease disclosure risk," *Data & Knowledge Engineering*, vol. 64, no. 1, pp. 346–364, 2008.

[32] K. Okamoto, W. Chen, and X.-Y. Li, "Ranking of closeness centrality for large-scale social networks," in *Frontiers in Algorithmics*, pp. 186–195, Springer, Berlin, Germany, 2008.

[33] C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," *Journal of computer and system sciences*, vol. 43, no. 3, pp. 425–440, 1991.

[34] L. Peng, R.-C. Wang, X.-Y. Su, and C. Long, "Privacy protection based on key-changed mutual authentication protocol in internet of things," in *Communications in Computer and Information Science*, pp. 345–355, Springer, Berlin, Germany, 2013.

[35] M. F. Rahman, W. Liu, S. Thirumuruganathan, N. Zhang, and G. Das, "Privacy implications of database ranking," *Proceedings of the VLDB Endowment*, vol. 8, no. 10, pp. 1106–1117, 2015.

[36] R. Schenkel, T. Crecelius, M. Kacimi et al., "Efficient top-k querying over social-tagging networks," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 523–530, ACM, Singapore, July 2008.

[37] C. Sheng, N. Zhang, Y. Tao, and X. Jin, "Optimal algorithms for crawling a hidden database in the web," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1112–1123, 2012.

[38] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE Symposium on Security and Privacy. S&P 2000*, pp. 44–55, IEEE, Berkeley, CA, USA, May 2000.

[39] J. Su, D. Cao, B. Zhao, X. Wang, and I. You, "ePASS: an expressive attribute-based signature scheme with privacy and an unforgeability guarantee for the Internet of Things," *Future Generation Computer Systems*, vol. 33, pp. 11–18, 2014.

[40] L. Sweeney, "k-anonymity: a Model for Protecting Privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[41] T. Tassa, A. Mazza, and A. Gionis, "k-concealment: An alternative model of k-type anonymity," *Transactions on Data Privacy*, vol. 5, no. 1, pp. 189–222, 2012.

[42] J. H. Y. F. W. Wang, J. C. W. G. K. Koperski, D. Li, Y. L. A. R. N. Stefanovic, and B. X. O. R. Z.. Dbminer, "A system for mining knowledge in large relational databases," in *Proceedings of the International Conference on Knowledge Discovery and Data Mining and Knowledge Discovery (KDD'96)*, pp. 250–255, San Diego, CA, USA, August 1996.

[43] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: toward data privacy in the IoT," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 725–730, IEEE, Sydney, Australia, June 2014.

[44] Y. Wang and Q. Wen, "A privacy enhanced dns scheme for the internet of things," in *Proceedings of the IET International Conference on Communication Technology and Application (ICCTA 2011)*, The Institution of Engineering & Technology, Beijing, China, October 2011.

[45] Y. Xu, T. Ma, M. Tang, and W. Tian, "A survey of privacy preserving data publishing using generalization and suppression," *Applied Mathematics & Information Sciences*, vol. 8, no. 3, pp. 1103–1116, 2014.

[46] J.-C. Yang and B.-X. Fang, "Security model and key technologies for the internet of things," *The Journal of China Universities of Posts and Telecommunications*, vol. 18, pp. 109–112, 2011.

[47] X. Yang, H. Steck, Y. Guo, and Y. Liu, "On top-k recommendation using social networks," in *Proceedings of the sixth ACM conference on Recommender systems—RecSys'12*, pp. 67–74, ACM, Dublin, Ireland, September 2012.

[48] X. Zheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Follow but no track: privacy preserved profile publishing in cyber-physical social systems," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1868–1878, 2017.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 1914038, 11 pages
https://doi.org/10.1155/2019/1914038

Research Article

A Novel Method for Location Privacy Protection in LBS Applications

Dan Lu,[1] **Qilong Han**,[1] **Kejia Zhang**,[1] **Haitao Zhang**,[1] **and Bisma Gull**[2]

[1] *College of Computer Science and Technology, Harbin Engineering University, Harbin, 150001, China*
[2] *Department of Electronic and Communications, National Institute of Technology Srinagar, Jammu and Kashmir, 190006, India*

Correspondence should be addressed to Kejia Zhang; kejiazhang@hrbeu.edu.cn

Received 17 April 2019; Accepted 16 June 2019; Published 15 July 2019

Guest Editor: Houbing Song

Location-based services have become a mainstream in people's daily lives due to continuous innovations in the field of mobile networking and GPS technologies. Recently they have advanced into a hot topic to which the majority of researchers pay close attention about how to enjoy them while safeguarding the location privacy of mobile users. Existing works involve the injection of random noise that cannot pledge the quality of service. Herein this manuscript, we propose a novel location privacy protection model based on the loss of service quality. This model allows the user to express his/her requirement of service quality by specifying the maximum service quality loss L_{max}, which is the user's tolerance. L_{max} can be set to 0. Our comprehensive experimental evaluation using a real-world dataset demonstrates that our modus outdoes other state-of-the-art approaches.

1. Introduction

Location-based services (LBS) are swelling owing to the innovations in technology and the dominance of location-cognizant devices [1–3]. Such services take the user's location information in a query as an input, execute the query at the server, and then provide the user with the information of nearby points of interest (POI), such as gas stations, banks, and restaurants [4]. A wide range of LBS applications include location-aware search (Baidu Maps), E-commerce (Meituan, Dianping), location-based social recommendation (QQ, WeChat), and ordering application and crowdsourcing (Ali) [5].

During this process, users' current or future whereabouts and interests are disclosed to the LBS server through their queries. Access to all submitted information is deemed necessary to best serve users; the LBS server is entrusted with rich information [6–8]. However, many studies have revealed that service providers can be honest but curious, belligerently stockpiling information of profile users, identifying homes, working places, and social relationships, or inferring interests towards commercial purposes [9–12]. Therefore, the concern of LBS is to provide high quality service while the user's location is anonymous to the LBS server. It seems contradictory and challengeable [13].

The topic of LBS privacy has been widely studied. In 2003, Beresford proposed the concept of location privacy [14], which pioneered the research on location privacy protection. Since then, the research on discrete location privacy or trajectory privacy has been published successively. There are two types of privacy issues in LBS: location privacy and query privacy [15]. Location privacy includes users' previous, current, and future location and query privacy is the type of POI he/she is interested in. In addition, the importance of query privacy is greater when the request is sensitive (query for hospitals). In this paper, we propose a novel location privacy protection model for the former, which ensures high quality service while user location is protected.

These approaches regarding the location privacy in LBS are classified into three categories. The first one is to enlarge the user location into a region; the representative is k-anonymity: e.g., an obfuscated region is formed by k users [16]. The second one can be viewed as a dummy-based technique [17]. The dummy location is sent to the LBS server instead of the exact location. Obviously, the user utilizes another position to replace his/her location. The

major limitation of such replacement is that the quality of service degrades significantly if the user chooses a higher level of privacy. The last one is to transform the original query into another problem such that the users' location cannot be inferred [15]. This kind of approaches usually employs cryptographic algorithms and spatial transformation techniques (e.g., Hilbert curve).

Geo-indistinguishability (GeoInd), a formal notion of location privacy introduced in [18], builds on the concept of differential privacy [19] to design user-centric location privacy protection mechanisms. GeoInd guarantees that obfuscated locations are statistically indistinguishable from other locations within a radius around the users' real location. However, [20] illustrates that GeoInd can be misleading in terms of both privacy and utility. Sometimes, GeoInd mechanisms possibly generate an obfuscated location vary far away from the user leading to dissatisfying service quality.

In this paper, a trusted third party (TTP) is added between the user and the server, for collecting users' real location and then sending to the LBS server with the disturbed one. The TTP perturbs the real location with a novel location privacy protection model based on loss of service quality which solves the challenge to ensure high quality service while protecting location privacy. To summarize, our contributions are as follows:

(1) We propose a function of service quality loss ($loss(P_n, P_r)$) based on a real query result, in which P_r is the real query result and P_n is the perturbation

(2) We propose a novel location privacy protection model based on loss of service quality. The TTP calculates a noisy area based on the maximum service quality loss (L_{max}) specified by the user and selects one point randomly to return to the server

(3) We propose a novel traversal method based on a Voronoi diagram, considering the geographic relationships of locations that efficiently reduce the complexity of computation

2. Related Works

Due to the paramount importance of location privacy in LBS services, it has been studied extensively, and several methods have been proposed. We review some of the typical briefly. K-anonymity based on trajectory generalization has been prevailing for its good balance of privacy protection and data availability. In [21], a (k, δ) anonymous algorithm was proposed for trajectory dataset publication. Based on trajectory generalization and k-anonymity, this algorithm generalizes every position in the trajectory to a circle with a radius of δ and makes sure that each circle has at least k points to satisfy k-anonymity, each of which is represented by a cylinder of these circles. Literature [22] proposed a technique by replacing the original data with a logical one.

Differential privacy was quickly applied to the privacy protection of data publishing [23] based on fake data technology to achieve privacy protection by adding noise to the real dataset [24]. In data distribution, differential privacy can achieve different levels of privacy protection and data

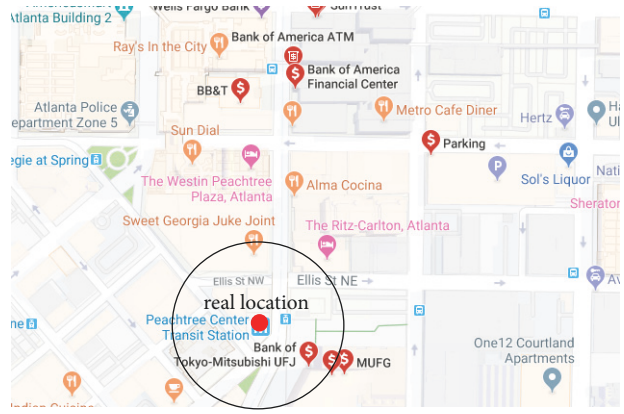


Figure 1: Actual result of banks within 500 m.a

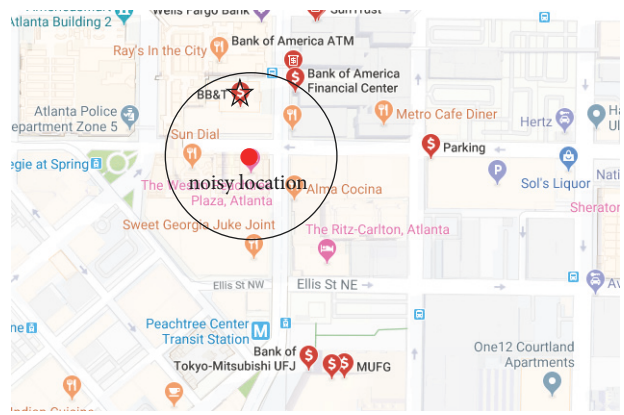


Figure 2: Query result with a noisy location.

publishing accuracy by adjusting the privacy parameter ϵ. In general, the larger the value of ϵ, the lower the level of privacy protection and the higher the accuracy of the published dataset. The first common mechanism for implementing differential privacy is the Laplace mechanism proposed in [25]. This mechanism mainly focuses on numeric queries, by adding random noise obeying the Laplace distribution to the results of real queries [26, 27]. For nonnumeric queries, [28] proposed an exponential mechanism, which is the second universal mechanism to achieve differential privacy.

Since its proposal in 2013, GeoInd has drawn a lot of attention from the research community. It has been widely used based on its core qualitative advantages, regardless of the adversary's background information. However, [21] illustrates that GeoInd is not that great. Sometimes, GeoInd possibly generates an obfuscated location very far away from the user leading to worthless data as shown in Figures 1 and 2.

To rectify this, we propose a novel location privacy protection model to replace the user's real location with an obfuscated one based on loss of service quality.

3. Preliminaries

In this section, the symbols and related definitions used in this paper are given. As mentioned earlier, the quality

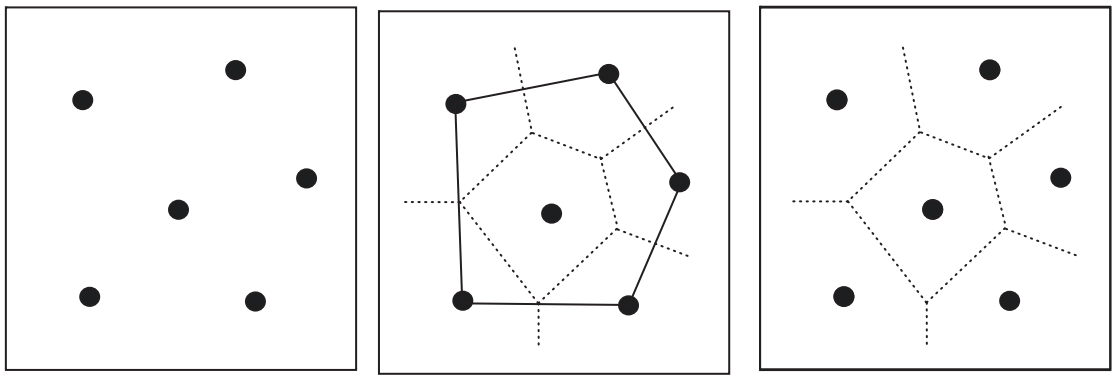


Figure 3: Generation of the Voronoi diagram.

of service declines dramatically after adding the Laplace noise, which means the obfuscated location is far away from the real one. To fix this, we propose a loss of service quality ($loss(P_n, P_r)$) based on the real query result as a novel evaluation index. The obfuscated location is generated randomly from the noisy area, which is calculated according to a specific $loss(P_n, P_r)$.

Real query result: the TTP receives the realistic location l and query radius r of a user, using l as the center within r, the set made up of points of interest (POI) sorted by the distance from l.

LBS query result: the LBS server takes the obfuscated location l' from the TTP, using l' as the center within same r, the set made up of points of interest (POI) sorted by the distance from l'. This article leverages the maximum service quality loss (L_{max}) to constrain the LBS query result.

Loss of service quality ($loss(P_n, P_r)$): regard the change of the real query result as $loss(P_n, P_r)$. According to the statistics about the clickthrough rates of search results released from AOL and IMN [29], ranking and attention were found to be expressed by a power function $y = \lambda a^{-x} (a > 1)$. Therefore, the weight of rank is set to $W(1) = \lambda a^{-1}$, $W(2) = \lambda a^{-2}, \ldots, W(k-1) = \lambda a^{1-k}$, $W(k) = \lambda a^{1-k}$, in which $w(i)$ denotes the weight of rank i, and the last two weights repeat. Given the real query result $P_r =< a, b, c, \ldots)$ and the obfuscated result $P_n =< \ldots, a, \ldots, b, \ldots)$, in which $\text{rank}(a, P_r)$ denotes the index of POI a at P_r, $loss(P_n, P_r)$ is formally defined below: compare the ranking of each POI after added noise; regard the weight difference $w(\text{rank}(x, P_r)) - w(\text{rank}(x, P_n))$ as the loss of x if the ranking drops. We set $w(\text{rank}(x, P_n)) = 0$ if x is not present in the obfuscated result.

$$loss(P_n, P_r) = \sum_{x \in P_r} \Delta_w(x)$$

$$\Delta_w(x) = \begin{cases} w(\text{rank}(x, P_r)) - w(\text{rank}(x, P_n)) & \text{if } \text{rank}(x, P_r) < \text{rank}(x, P_n) \\ 0 & \text{else} \end{cases} \tag{1}$$

$$\text{For } x \notin P_n, \ \text{rank}(P_n) = +\infty, \ w(\text{rank}(x, P_n)) = 0.$$

Maximal tolerance L_{max}: this is the maximal loss of service quality that a user may tolerate. The smaller L_{max} is, the more similar the obfuscated result and the real one are.

Euclidean distance: the Euclidean distance is the shortest distance between two points in space. Given two points in two dimensions (x_1, y_1) and (x_2, y_2), the Euclid distance of two points is defined: $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

The Voronoi diagram [30]: the Voronoi diagram, also known as the Thiessen polygon or Dirichlet diagram, generates a Delaunay triangulation at first and connects the center of the circumcircle of the adjacent triangle. The characteristic is that there is a generator with each V polygon in the graph, and the distance from the inner point of each V to the generator is shorter than other generators. Points on the boundary of two polygons are equidistant from the corresponding generator. The establishment method of the Voronoi diagram is shown in Figure 3.

4. Our Framework of Privacy

In this section we describe our system architecture and the novel method of location privacy protection. We use Baidu Maps API [31] as the trusted third party (TTP) which is added between the user and the server, for collecting the user's real location and then sending to the LBS server with the disturbed one. The TTP perturbs the real location with a novel location privacy protection model based on loss of

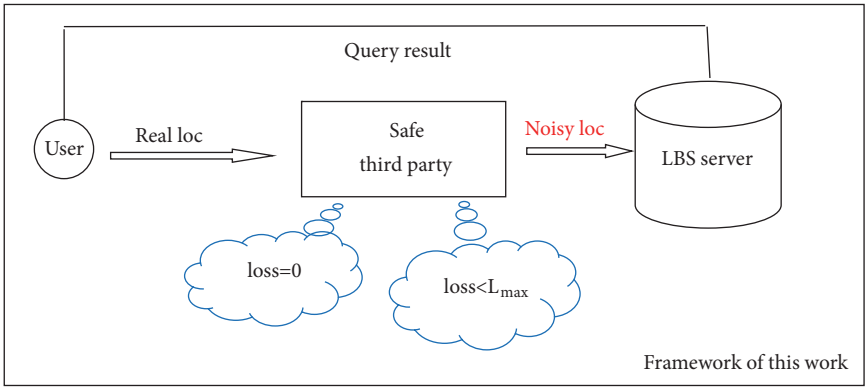


Figure 4: System architecture.

service quality which solves the challenge to ensure high quality service while protecting location privacy. The overall system architecture is shown in Figure 4. The model allows a user to express his/her requirement of service quality by specifying a maximum service quality loss L_{max}, in which the user would tolerate the loss of service quality ($loss(P_n, P_r) \leq L_{max}$). L_{max} can also be set to 0, which means immutable service quality. To guarantee the quality of query service, L_{max} is typically set to a small value.

It can be easier to calculate the distance between points in two dimensions; we convert from latitude and longitude to UTM coordinates [32], also flagged as l.

4.1. Nonlossy Service Quality ($loss(P_n, P_r) = 0$). As mentioned in the previous section, the loss of service quality ($loss(P_n, P_r)$) based on the real query result is a novel evaluation index to measure the difference between the real query result and the obfuscated one. There are two kinds of situations. The first is that the number of POI and ranking stay the same (the last two points of interest are interchangeable). In another story, the number of POI increases while the rankings of points in the real query result are interchangeable. For instance, given real query result *ABCDE*, the obfuscated result could be *ABCDE* or *ABCDE***FG** if $loss(P_n, P_r) = 0$. POI F and G in bold are generated in addition without impacting the loss of service quality. Therefore, to ensure nonlossy service quality, postprocessing would be required on the obfuscated result.

4.1.1. Generation of Obfuscated Region. Given a real location l of a user, the real query result can be obtained by calling Baidu Maps API. Calculate the obfuscated region according to the ranking of the real query result (proximal to distal from l) and the Voronoi diagram. The distance from each point within it to each query result satisfies the true ranking.

Algorithm 1 illustrates the details of the algorithm of obfuscated region generation. Given the real location l of a user, Step 2 obtains the real query result by calling Baidu Maps API. Step 4 executes the Delaunay triangulation algorithm backwards according to the ranking. We compute the overlapped region at each step to find the final obfuscated region κ. The step of the Delaunay triangulation algorithm is as follows:

Input: Real location l, radius r
Output: generated area κ
1. Initialize generated area $\kappa = \phi$
2. $P_r = BaiduAPI.Query(l)$
3. **for** each $x \in P_r$ **do**
4. Generate Delaunay triangulation of x
5. Calculate the Voronoi area of x as κ_x
6. $P_r.remove(x)$
7. **if** $x = top1$ **then**
8. $\kappa = \kappa_x$
9. **else**
10. $\kappa = \kappa \cap \kappa_x$
11. $\kappa = \kappa \cap \text{circle}(P_r.getLast(), r)$
12. **return** κ

Algorithm 1: Obfuscated region generation.

(1) Construct the Delaunay networks with the discrete points

(2) Calculate the center of the circumcircle of each triangle and take it down

(3) Look for three adjacent triangles whose border is in common with the current triangle

(4) If adjacent triangles are found, connect the circumcenter of each one to the circumcenter of the current triangle. If not, calculate the midperpendicular of the outermost border

4.1.2. Postprocessing of Obfuscated Region. We get the obfuscated region from the previous section satisfying $loss(P_n, P_r) = 0$, like polygon *AOPQR* in Figure 5. A closer inspection would reveal an extra POI K on the certain extension of the query if the obfuscated location were located on A. Besides, the distance from K to A is less than the distance from D to A ($d_2 < d_1$). In this case, K influences the ranking of D leading to $loss \neq 0$. So, to hedge against this, we need to do the postprocessing of our region.

To guarantee $loss(P_n, P_r) = 0$, the distance from K to the obfuscated location l' must be larger than the distance from the last-ranking POI D to l', which is ($dis(K, l') \geq dis(D, l')$).

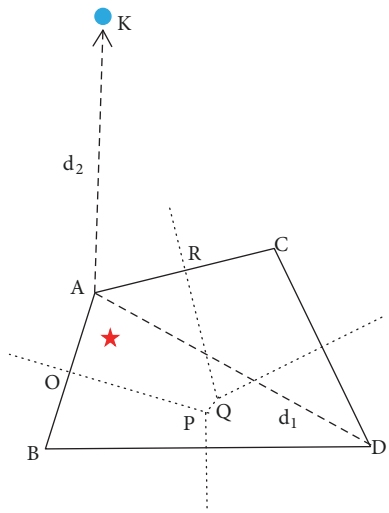


Figure 5: Obfuscated region from Section 4.1.1.

Input: Generated area κ, radius r, Real Rank P_r
Output: Final area κ'
1. Let V=vertex of area κ, $N = \phi$, $\kappa' = \kappa$
2. init A=ϕ
3. **for** each $v \in V$ **do**
4. Circle with radius r as c_v(v=1 to $|V|$)
5. Make the tangent of $\begin{cases} (c_v, c_{v+1}) & v \leq |V| \\ (c_v, c_1) & v = |V| \end{cases}$
6. **end for**
7. A=All enclosed area
8. $QueryResult(Q) = BaiduAPI(A)$
9. if $\exists p \in Q \cap p \notin P_r$
10. $N.add(p)$
11. **for** each $n \in N$ **do**
12. κ'=Area surrounded by perpendicular bisector of $(n, P_r.getLast())$ and κ'
13. **end for**
14. **return** κ

Algorithm 2: Postprocessing step.

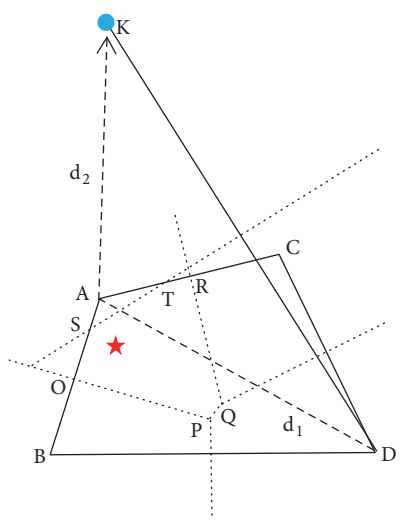


Figure 6: Ultimately obfuscated region.

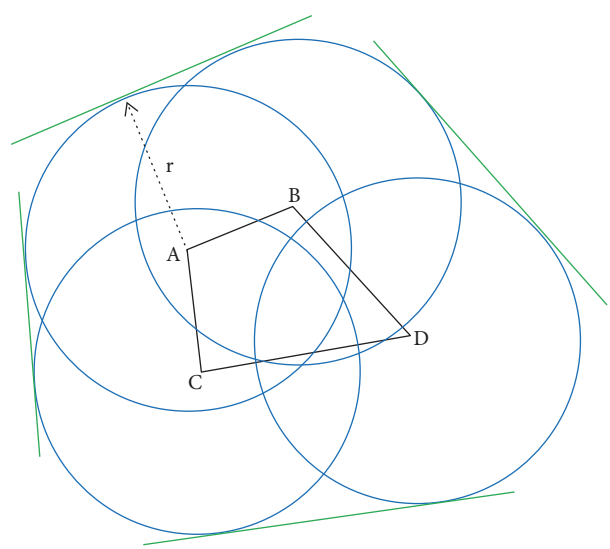


Figure 7: Query ranges.

The vertical bisector of segment KD crosses polygon $AOPQR$ at POI S and T. We get the ultimately obfuscated region just like Figure 6.

Algorithm 2 illustrates the details of the postprocessing algorithm. Given the obfuscated region κ calculated in the previous section, we initialize the set of vertices as V and the set of extra points $N = \phi$. In Steps 4 to 6, we compute the query ranges A with query radius r. The area enclosed by the red line in Figure 7 is the query ranges. Decide whether there come extra points after calling Baidu Maps API again within A and add them to N (Steps 7–9). In Steps 10 to 12, for each point n in the set N, draw the vertical bisector of segment $(n, P_r.getLast())$ crossing κ to form the new κ. That can ensure the distance from the last-ranking to the obfuscated location is less than that of any point in N to the obfuscated location.

4.2. Tolerable Quality of Services ($loss(P_n, P_r) \leq L_{max}$). In this section, we consider the case that service quality might be lossy. Given the maximal tolerance L_{max}, the loss of service quality within it is acceptable. By our definition in Section 3, the loss of service quality is the loss of weight regarding the real query result. The top priority in this section is to find all possible rankings under satisfying constraints, expressed as $P_x \mid loss(P_x, P_r) \leq L_{max}$. The traditional enumeration is complex, so we propose two enhanced enumeration algorithms to reduce the time complexity effectively.

4.2.1. Enumeration with Pruning. The first algorithm is enumeration with pruning. For a certain position i, calculate the upper bound sup L and the lower bound inf L of the queue

```
Input: real ranking P_r, Maximum tolerance L_max
Output: ranking Set P
1. init P = ϕ, sup(loss), inf (loss), C = P_r ∪ χ
2. for each c ∈ C do
3.     calculate sup L(c) and inf L(c)
4.     if (inf L(c) < L_max) then
5.         P.append(c)
6. for i = 2 to |P_r| do
7. init N = ϕ
8.     for each p in P do
9.         for j = 1 to |P_r| + 1 do
10.            if (pcontains(C[j]) ∩ C[j] ≠ χ) then
11.                continue
12.            temp = p + C[j]
13.            calculate sup L(temp) and inf L(temp)
14.            if (inf L(temp) < L_max) then
15.                N.append(temp)
16.     P = N
17. return P
```

Algorithm 3: Enumeration with pruning.

```
Input: real ranking P_r; Maximum tolerance L_max
Output: ranking Set P;
1. init queue q, set of Candidate Candy;
2. list = P_r; 3. for each x ∈ P_r do
4.     Generate Delaunay triangulation of x
5. end for
6. function circulate(list):
7.     for each item ∈ list do
8.         q.add(item);
9.         if sup L(q) ⩽ L_max then
10.            P.add(q);
11.        else if inf L(q) > L_max then
12.            candy.remove(x) rank (x) > rank (item);
13.        else
14.            q.add(item);
15.            calculate the candidate of q;
16.            circulate(Candy);
17. return P;
```

Algorithm 4: Enumeration with the Voronoi diagram.

after POI x joined. The POI x is not allowed in position i ($i \leq |P_r|$) if $\inf L > L_{max}$. Therefore, we can prune the branch of rank $(x) = i$. This is as in Algorithm 3.

Algorithm 3 illustrates the details of the enumeration algorithm with pruning. Given the real ranking P_r, we consider the possibility that each point may be the first. In Steps 6 to 9, we add POI into queue *temp* in turn to calculate $\sup L(temp)$ and $\inf L(temp)$. Each point can appear only once and the extra points may occur several times (Steps 10-11). In Steps 13 to 15, we store the current queue to N if the lower bound $\inf L$ is less than maximum tolerance L_{max}. By analogy, all the rankings that meet the constraints are obtained. This method has no regard for the geospatial and will generate many rankings unable to form a region.

4.2.2. Enumeration with a Voronoi Diagram. The pruning algorithm also has a high time complexity, and it will generate many useless rankings which can not form a region. To solve this, an enumeration method with a Voronoi diagram is given in this paper. The ultimately obfuscated region satisfying L_{max} can be obtained by dividing the polygons continuously. This method operates on the Voronoi diagrams directly, which is intuitive and easier for getting the obfuscated region without postprocessing.

Algorithm 4 illustrates the details of the enumeration with the Voronoi diagram algorithm. Given the real ranking P_r, we generate the Voronoi diagram only once. Step 6 starts the recursive function; calculate the upper bound $\sup L(q)$ and the lower bound $\inf L(q)$ after each addition. If the condition $\sup L(q) \leqslant L_{max}$ is met, add the current queue q into ranking set P. Moreover, if the condition $\inf L(q) > L_{max}$ is met, we remove all the points in candidate set *Candy* that ranked lower than q. Besides, in Steps 13 to 16, we divide the current region into multiple regions and start a new round of recursion. The candidate set creating algorithm is as in Algorithm 5.

Algorithm 5 illustrates the details of the candidate generation algorithm. Given the current queue q, the Delaunay triangulation dt, and the current candidate set *Candy*, the algorithm finds the neighbor POI of each point in queue q and adds them to *Candy*. Then, it sorts them according to the raw ranking. Using Figure 8 as an example, the real ranking is $\langle A, B, C, D\rangle$ and the maximum tolerance $L_{max} = 0.1$. Generate the Voronoi diagram at first, and calculate the likelihood of a given queue with each point on the top, just like $\langle A\cdots\rangle, \langle B\cdots\rangle, \langle C\cdots\rangle, \langle D\cdots\rangle$. The lower bounds of B,C,D $\inf L(B \mid C \mid D)$ are all greater than 0.1; that is why the top one must be A. After that, we partition the V polygon *Aqwry* into smaller polygons; the vertical bisector of segment *BC* crosses *Aqwry* at points t and e. Calculate the upper bound and the lower bound of two polygons, which are $\sup L(AB \mid AC)$ and $\inf L(AB \mid AC)$. The lower bound of polygon *tyre* is beyond L_{max}, while the upper bound of polygon *Aqwet* is under L_{max}. Therefore, we regard the gray area *Aqwet* as the ultimately obfuscated region.

5. Experimental Evaluation

In all experiments, we use the real Harbin Station $(45.768038, 126.644593)$ as the real location l and query the banks within 400 meters. The real result can be obtained by calling Baidu Maps. For the sake of simplicity, we only take the top 10 points of interest into consideration and regard the others as the extra ones. We ran our experiments on a desktop computer with an Intel Core i5-7200 2.50 GHz processor and 8 GB RAM. The real query result and the ranking are as follows.

Since the triangle made of $(110, 110)$, $(120, 120)$, $(130, 130)$ has the same shape as the triangle made of $(10, 10)$, $(20, 20)$, $(10, 30)$, we take the common prefix away to get

```
Input: Queue q; Delaunay triangulation dt; Set of Candidate Candy;
Output: set of Candidate Candy;
1. for each x ∈ q do
2.     for each Simplex t ∈ dt do
3.     if t.contains(x) then
4.         l=t.vertices;
5.         for each v ∈ l do
6.             if v ∉ Candy then
7.                 Candy.add(v);
8. Sort(Candy) with ranking;
9. return Candy;
```

Algorithm 5: Generate the candidate of a queue.

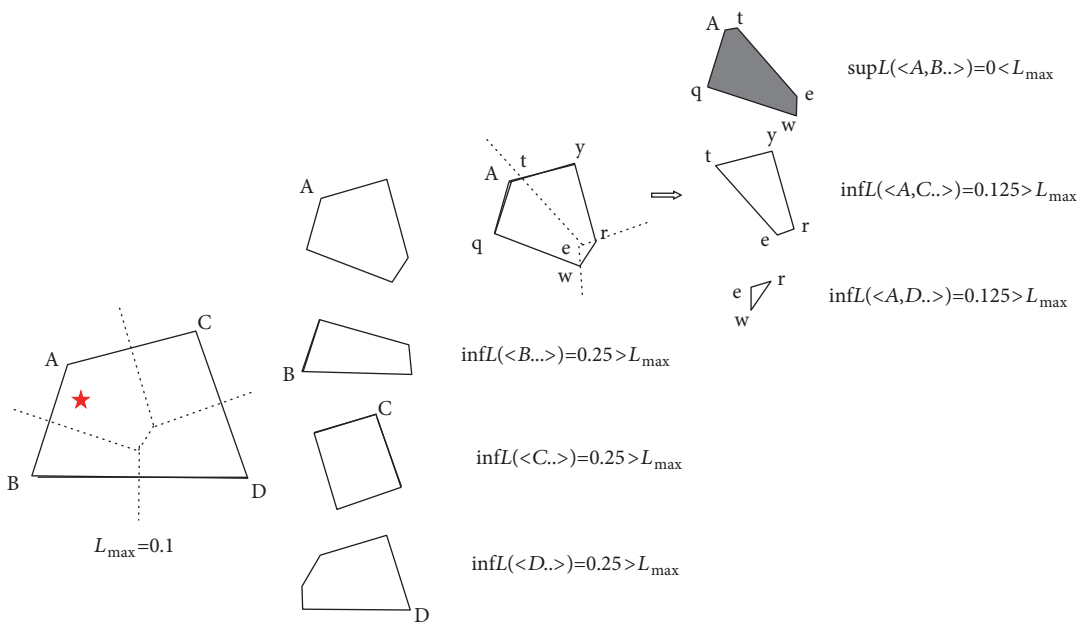


Figure 8: Calculate the obfuscated region.

a smaller coordinate value for computing triangulation conveniently. We change the top one POI (14097372.321867, 5711495.970734) to (372.321867,495.970734) and then do the same thing for the others as in Table 1. The change of coordinates will also induce the change of the distance between the real location and each query result. We regard the distance between the last one POI transformed (710.269230,3.976522) and the real location transformed (433.443102926,417.242938906) as the query radius, which is $r_{max} = 500$.

5.1. Performance. For the first situation which is $loss(P_n, P_r) = 0$, as defined in Section 3, the last two POI are interchangeable. We got the ultimately obfuscated region in terms of that. In order to realize the tolerable quality of services ($loss(P_n, P_r) \leq L_{max}$), given L_{max}, the key question is how to get all the ranking results. A useful lemma combining the classical triangulation is shown as follows.

Table 1

Rank	Geographic	Horizontal
1	126.644040,45.768543	14097372.321867,5711495.970734
2	126.643815,45.768693	14097347.451385,5711519.173524
3	126.643240,45.769379	14097283.918655,5711626.604204
4	126.646863,45.766884	14097684.462267,5711240.031558
5	126.647147,45.769253	14097716.036524,5711617.643764
6	126.648564,45.768321	14097872.775613,5711473.096577
7	126.648580,45.768274	14097874.543330,5711465.660886
8	126.648690,45.767788	14097886.686469,5711388.631129
9	126.641381,45.766047	14097078.182499,5711090.967867
10	126.647097,45.765396	14097710.269230,5711003.976522

Lemma 1. *If the ranking of the current generator (vertex) v is r, C represents the set of vertices within the triangulations that*

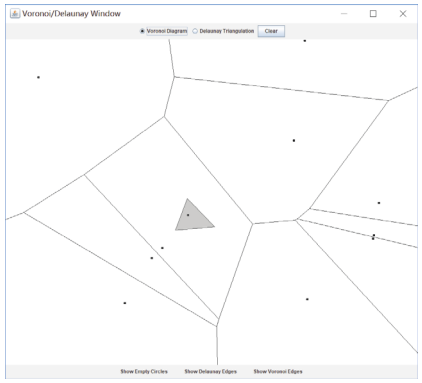


Figure 9: Obfuscated region with nonlossy service quality ($\lambda = 1, a = 2$).

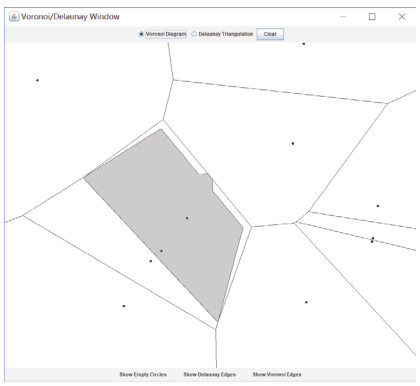


Figure 10: Obfuscated region with tolerance ($\lambda = 1, a = 2, L_{max} = 0.2$).

contain r. Then all the possible vertices in ranking $r + 1$ are expressed as $v' \in C$ ($v' \neq v$).

Proof. The Delaunay triangulation (TIN) gathers the 3 nearest neighbors, and each generator (vertex) has a public edge with the others in the Voronoi diagram [33]. Since $dis(v', v) \forall v \in C < dis(v', others)$, the vertices which can be ranking in $r + 1$ must have a public edge with the current generator (vertex). □

As shown in Figure 9, any points in the gray area satisfy the nonlossy service quality. The user's real location is protected while receiving the highest quality of service. We utilize the theory of the Voronoi triangulations instead of simple enumeration to slump the time complexity of the ranking calculation. As shown in Figure 10, we set the weight of rank and the maximal tolerance as $a = 2$ and $L_{max} = 0.2$. The distance between each point of the obfuscated region and any one query result is within the maximum query ranges, which can be expressed as $dis(v, l') < r_{max}$.

5.2. Effect of L_{max}. We studied the scalability of our method by varying L_{max} in the range of 0.1 to 0.25. The weighting parameter a was 2. Figure 11 presents the obfuscated region when L_{max} increases from 0.1 to 0.25. As can be observed, the obfuscated region increases constantly as L_{max} increases, which realizes more protection of the user's location. It is slow growth when L_{max} increases from 0.1 to 0.2, but when we set $L_{max} = 0.25$, the obfuscated region nearly doubles on account of the change of the higher rankings.

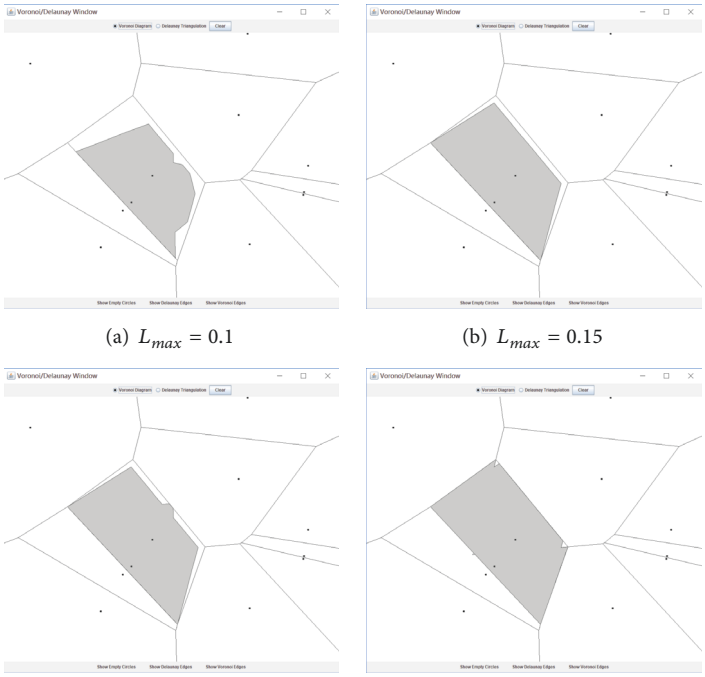
5.3. Effect of the Weighting Parameter a. We studied the scalability of our method by varying a in the range of 2 to 4 and λ in the range of 1 to 3. L_{max} was 0.2. Figure 12 presents the obfuscated region when the weighting parameter ($\lambda a^{(} - 1)$) increases from 1/2 to 3/4. As can be observed, the obfuscated region shrinks constantly as $\lambda a^{(} - 1)$ increases, which realizes better quality of service. It is significant change when weighting parameter increases from 1/2 to 2/3, and there is no obvious change from 2/3 to 3/4.

5.4. Errors of Perturbation. We now compare the perturbation errors of our scheme with the Laplace noises from 10 experiments. We set the parameter of global sensitivity as Δf=300 and vary ϵ from 0.5 to 1 and L_{max} from 0 to 0.15. Figure 13 depicts the comparison results and we can see that ϵ decreasing and L_{max} increasing can lead to the perturbation error increases. Besides, our scheme achieves fewer errors than the Laplace noise.

5.5. Comparison of Ranking Calculating Time. We then look at the ranking calculating time of our enumeration algorithms: enumeration with pruning and enumeration with the Voronoi diagram. We set $L_{max} = 0.1$ and perform 10 experiments. The results are shown in Figure 14. We can see that the time cost of enumeration with the Voronoi diagram is approximately 30% of enumeration with pruning. Experiments prove the enumeration with the Voronoi diagram can effectively reduce time complexity.

6. Conclusion

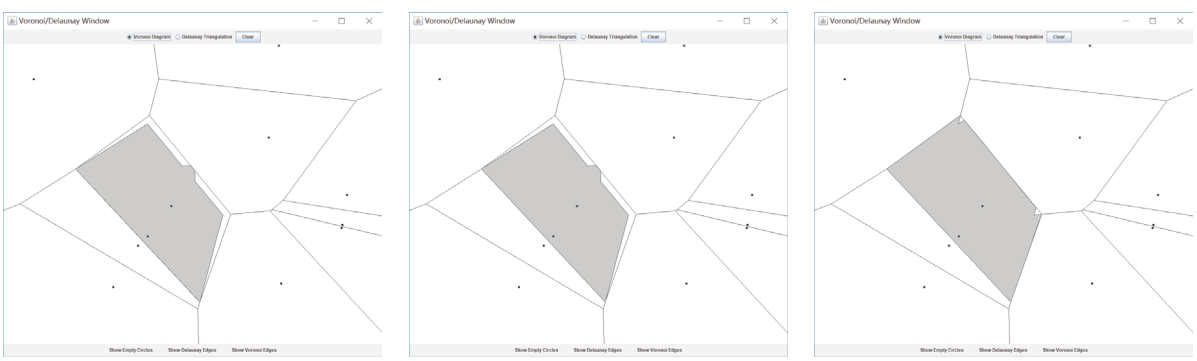
In this paper, we propose a novel location privacy protection model based on the loss of service quality. A trusted third party (TTP) is added between the user and the server, for collecting the user's real location and then sending to the LBS



(a) $L_{max} = 0.1$ (b) $L_{max} = 0.15$

(c) $L_{max} = 0.2$ (d) $L_{max} = 0.25$

Figure 11: Effect of L_{max}.



(a) $\lambda = 1, a = 2$ (b) $\lambda = 2, a = 3$ (c) $\lambda = 3, a = 4$

Figure 12: Effect of the weighting parameter a and λ.

server with the disturbed one. The model allows a user to express his/her requirement of service quality by specifying a maximum service quality loss L_{max}, which the user would tolerate. The loss of service quality L_{max} can also be set to 0. Find all possible rankings under satisfying constraints to get the final obfuscated region, and then select one point at random as the obfuscated location. In order to ensure the excellent service, L_{max} is usually set to a smaller one.

In this paper, we only see the user's location for privacy and a novel strategy based on the Voronoi diagram is used to generate the noisy location in order to improve the service quality. Since the query privacy is also a key privacy concern, the query interests incur potential damage to personal privacy and even to individual safety. How to ensure the query privacy is another area we would like to investigate further.

Data Availability

The data used in our paper is just longitude and latitude of points of interest (POI), and these are public and can be obtained by anyone. And the third party of our work is Baidu Maps API, which is also public.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This article is partly supported by the National Natural Science Foundation of China under Grant No. 61370084

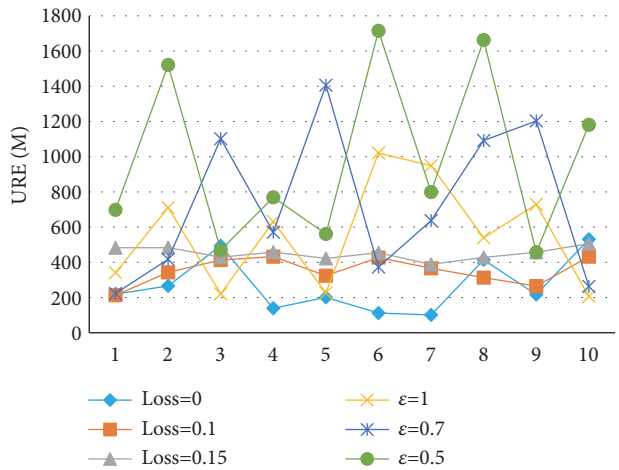
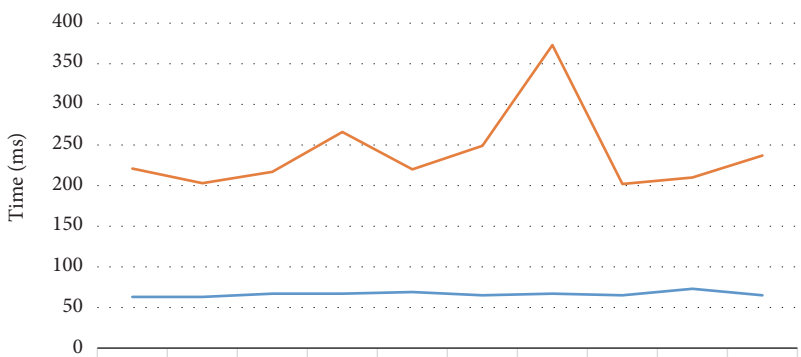


Figure 13: Errors of perturbation.



	1	2	3	4	5	6	7	8	9	10
Voronoi	63	63	67	67	69	65	67	65	73	65
Pruning	221	203	217	266	220	249	373	202	210	237

Figure 14: Comparison of ranking calculating time.

and No. 61872105, the Experimental Verification of the Basic Commonness and the Key Technical Standards of the Industrial Internet Network Architecture, the Key Technology of Home-Based Care Service System (2016RAXXJ013), and Fundamental Research Funds for the Central Universities (Grant No. 3072019CF0602).

References

[1] Q. Han, S. Liang, and H. Zhang, "Mobile cloud sensing, big data, and 5G networks make an intelligent and smart world," *IEEE Network*, vol. 29, no. 2, pp. 40–45, 2015.

[2] K. Zhang, Q. Han, Z. Cai, and G. Yin, "RiPPAS: a ring-based privacy-preserving aggregation scheme in wireless sensor networks," *Sensors*, vol. 17, no. 2, p. 300, 2017.

[3] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart internet of things systems: a consideration from a privacy perspective," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 55–61, 2018.

[4] M. F. Mokbel, "Privacy in location-based services: state-of-the-art and research directions," in *Proceedings of the 8th International Conference on Mobile Data Management (MDM)*, p. 228, IEEE, Mannheim, Germany, 2007.

[5] Z. Cai and Z. He, "Trading private range counting over big iot data," in *Proceedings of the 39th IEEE International Conference on Distributed Computing Systems*, 2019.

[6] Y. Liang, Z. Cai, J. Yu, Q. Han, and Y. Li, "Deep learning based inference of private information using embedded sensors in smart devices," *IEEE Network*, vol. 32, no. 4, pp. 8–14, 2018.

[7] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.

[8] Z. Cai, X. Zheng, and J. Yu, "A differential-private framework for urban traffic flows estimation via taxi companies," *IEEE Transactions on Industrial Informatics*, 2019.

[9] S. Gambs, M.-O. Killijian, and M. N. del Prado Cortez, "Show me how you move and I will tell you who you are," *Transactions on Data Privacy*, vol. 4, no. 2, pp. 103–126, 2011.

[10] J. Krumm, "Inference attacks on location tracks," in *Proceedings of the Pervasive Computing, 5th International Conference (PERVASIVE)*, pp. 127–143, Toronto , Canada, 2007.

[11] Y. Matsuo, N. Okazaki, K. Izumi et al., "Inferring long-term user properties based on users location history," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2159–2165, Hyderabad, India, 2007.

[12] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, 2018.

[13] Y. Huo, X. Fan, L. Ma, X. Cheng, Z. Tian, and D. Chen, "Secure communications in tiered 5G wireless networks with cooperative jamming," *IEEE Transactions on Wireless Communications*, vol. 18, no. 6, pp. 3265–3280, 2019.

[14] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive Computing*, vol. 2, no. 1, pp. 46–55, 2003.

[15] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pp. 121–132, Vancouver, BC, Canada, 2008.

[16] P. Samarati and L. Sweeney, "Generalizing data to provide anonymity when disclosing information (abstract)," in *Proceedings of the 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, p. 188, Seattle, Wash, USA, 1998.

[17] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *Proceedings of the 21st International Conference on Data Engineering Workshops (ICDEW)*, p. 1248, Tokyo, Japan, 2005.

[18] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: differential privacy for location-based systems," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pp. 901–914, ACM, Berlin, Germany, 2013.

[19] C. Dwork, "Differential privacy," in *Proceedings of the in Automata, Languages and Programming, 33rd International Colloquium (ICALP)*, pp. 1–12, Venice , Italy, 2006.

[20] S. Oya, C. Troncoso, and F. Pérez-González, "Is geo-indistinguishability what you are looking for?" in *Proceedings of the 2017 on Workshop on Privacy in the Electronic Society*, pp. 137–140, Dallas, Tex, USA, 2017.

[21] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: uncertainty for anonymity in moving objects databases," in *Proceedings of the IEEE 24th International Conference on Data Engineering (ICDE)*, pp. 376–385, IEEE, Cancun, Mexico, 2008.

[22] Q. Han, D. Lu, K. Zhang, X. Du, and M. Guizani, "Lclean: a plausible approach to individual trajectory data sanitization," *IEEE Access*, vol. 6, pp. 30110–30116, 2018.

[23] Q. Han, B. Shao, L. Li, Z. Ma, H. Zhang, and X. Du, "Publishing histograms with outliers under data differential privacy," *Security and Communication Networks*, vol. 9, no. 14, pp. 2313–2322, 2016.

[24] X. Zheng, Z. Cai, J. Li, and H. Gao, "Location-privacy-aware review publication mechanism for local business service systems," in *Proceedings of the IEEE INFOCOM - Conference on Computer Communications*, pp. 1–9, IEEE, Atlanta, Ga, USA, 2017.

[25] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proceedings of the 3rd Theory of Cryptography Conference (TCC)*, pp. 265–284, Springer, New York, NY, USA, 2006.

[26] Q. Han, Q. Chen, L. Zhang, and K. Zhang, "HRR: a data cleaning approach preserving local differential privacy," *International Journal of Distributed Sensor Networks*, vol. 14, no. 12, 2018.

[27] Q. Han, Z. Xiong, and K. Zhang, "Research on trajectory data releasing method via differential privacy based on spatial partition," *Security and Communication Networks*, vol. 2018, Article ID 4248092, 14 pages, 2018.

[28] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in *Proceedings of the 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, p. 1592, Allerton Park & Retreat Center, Monticello, IL, USA, 2013.

[29] https://www.internetmarketingninjas.com/.

[30] https://en.wikipedia.org/wiki/Voronoi_diagram/.

[31] http://lbsyun.baidu.com/.

[32] J. A. O'Keefe, "The universal transverse mercator grid and projection," *The Professional Geographer*, vol. 4, no. 5, pp. 19–24, 1952.

[33] M. Mostafavi, G. Abolfazl, Christopher., and D. Maciej, "Delete and insert operations in voronoi/delaunay methods and applications," *Computers and Geosciences*, vol. 29, no. 4, pp. 523–530, 2003.

Hindawi
Security and Communication Networks
Volume 2019, Article ID 4074272, 15 pages
https://doi.org/10.1155/2019/4074272

Research Article

Towards Supporting Security and Privacy for Social IoT Applications: A Network Virtualization Perspective

Jian Sun,[1] **Guanhua Huang**,[1] **Arun Kumar Sangaiah**,[2] **Guangyang Zhu**,[1] **and Xiaojiang Du**[3]

[1] *Key Lab of Optical Fiber Sensing and Communications (Ministry of Education), UESTC, Chengdu, China*
[2] *Vellore Institute of Technology, Tamil Nadu, India*
[3] *Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA*

Correspondence should be addressed to Jian Sun; sj@uestc.edu.cn

Received 16 January 2019; Accepted 25 February 2019; Published 14 March 2019

Guest Editor: Houbing Song

Network function virtualization (NFV) is a new way to provide services to users in a network. Different from dedicated hardware that realizes the network functions for an IoT application, the network function of an NFV network is executed on general servers, and in order to achieve complete network functions, service function chaining (SFC) chains virtual network functions to work together to support an IoT application. In this paper, we focus on a main challenge in this domain, i.e., resource efficient provisioning for social IoT application oriented SFC requests. We propose an online SFC deployment algorithm based on the layered strategies of physical networks and an evaluation of physical network nodes, which can efficiently reduce bandwidth resource consumption (OSFCD-LSEM) and support the security and privacy of social IoT applications. The results of our simulation show that our proposed algorithm improves the bandwidth carrying rate, time efficiency, and acceptance rate by 50%, 60%, and 15%, respectively.

1. Introduction

In traditional service provider networks, network functions (NFs) (e.g., intrusion detection systems (IDSs), gateways, load balancers, network address translators (NATs), and firewalls [1, 2]) for social IoT applications [3–6] are implemented by specialized hardware devices, and it is expensive to incorporate some new devices into an existing service network. With an increasing number of social IoT applications, the demand of corresponding NFs for each user necessitates a significant amount of hardware resources. Furthermore, security and privacy preservation in social IoT applications are also important issues that need to be solved [7–10]. To address these problems, network function virtualization (NFV) technology has been put forward. In NFV networks, NFs are implemented in the form of software instead of dedicated hardware and separately run on different virtual machines (VMs) [11, 12] and thus guarantee the security and privacy requirements. As shown in Figure 1, NFs that run in NFV networks are called virtual network functions (VNFs). Different VNFs are arranged in a prescribed order to form a service function chain (SFC) to fulfill the communication requirements [13–15].

NFV is convenient for social IoT service providers to deploy VNFs on commercial servers and manage the underlying network [16–18]. Using NFV technology, the social IoT service provider can flexibly deploy virtual NFs on commercial servers [19–21], which are traditionally fulfilled on dedicated hardware. Furthermore, with an increasing social IoT service demand [22–25], common commercial servers can provide support to multiple VNFs, which significantly reduces the vacancy rate of physical resources [26–28] and saves the cost of purchasing new dedicated hardware. The network operators can flexibly deploy and chain the VNFs according to the social IoT users' requests and practical network topology. This approach decreases the use of inapplicable functions and the corresponding deployment costs. Moreover, this approach can customize the social IoT services for the clients, introduce a broad business outlook, and promote its commercialization. Regarding

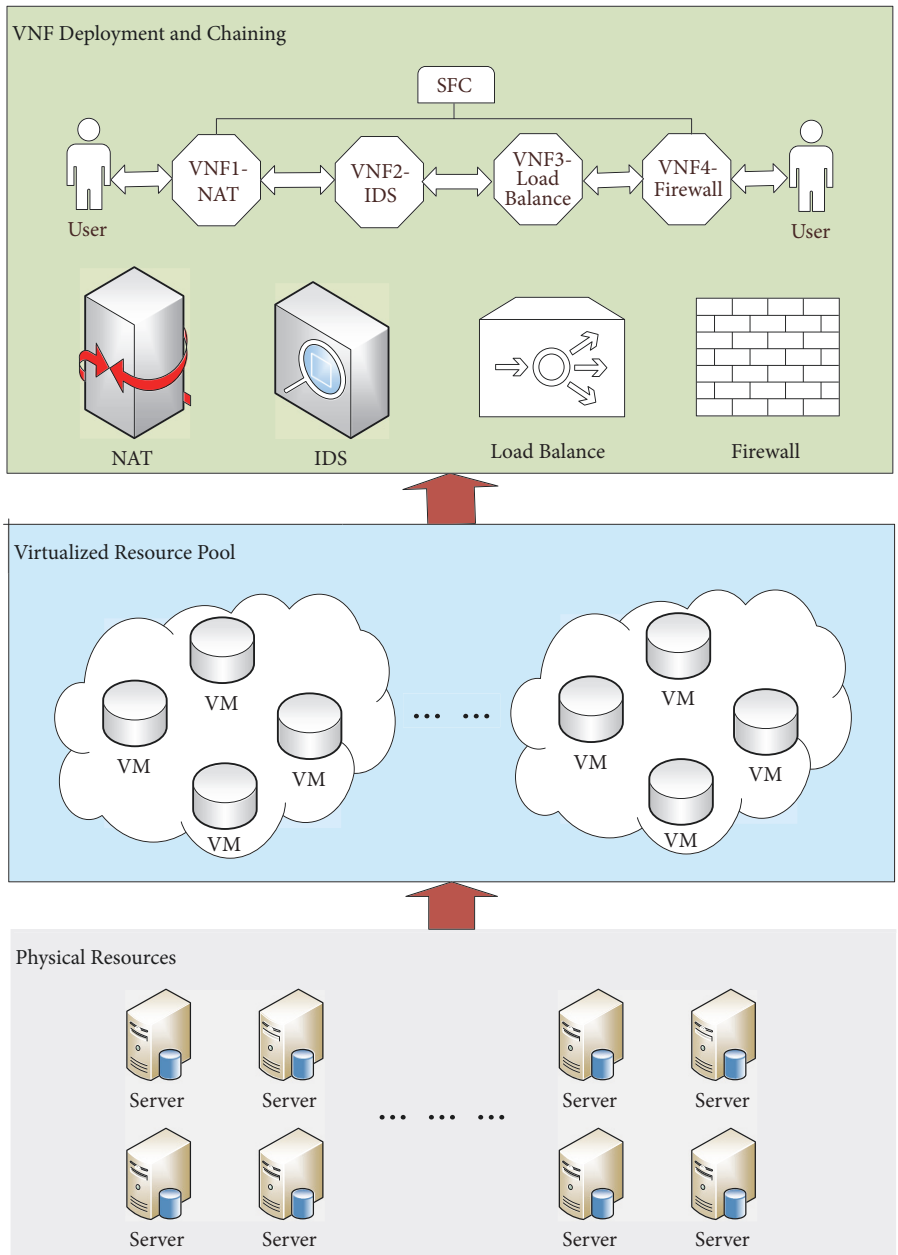


Figure 1: SFC organization by virtualizing commercial servers.

traditional NFs, VNFs increase policy compliance capabilities [29] and promote the emergence of more novel techniques and the development of NFV for social IoT application.

NFV can bring additional benefits to social IoT service providers, i.e., optimize their operational expenditure (OPEX) and capital expenditure (CAPEX) [11, 30–32] and improve the quality of service (QoS) by decreasing latency, reducing bandwidth consumption and increasing adaptation. NFV introduces many benefits to both users and social IoT service providers, but there are still many research limitations that need to be dealt with. For example, the bandwidth consumption of the SFC is an important research topic. Bandwidth consumption represents the resource cost of communication among users/clients [33, 34]. The network operator seeks to achieve low bandwidth consumption to complete the deployment of the current SFC request and save resources for the next SFC request to improve the accommodating capacity of the network [35]. With an increasing diversity of service requirements and additional pressure from massive information transmission, bandwidth resources have become progressively scarce [36]. Furthermore, low bandwidth consumption can provide the client/user a good experience due to the excellent performance of the network operator. In previous research, several algorithms have aimed to reduce bandwidth consumption. In [17], the authors

focused on bandwidth consumption optimization and provided a framework for their studied problem. Thus, instead of the existing hardware environment, targeted research of NFV can produce increased benefits and reduce the energy and space consumption of various middle boxes [19, 37].

In this paper, we study the problem of how to provision social IoT oriented SFC requests while minimizing the bandwidth resource consumption as well supporting the security and privacy requirements. The studied problem has been proven to be an NP-hard problem [35, 37, 38].

Thus, we propose an efficient algorithm, called OSFCD-LSEM, with layered strategies for physical networks. It not only efficiently solves the SFC deployment problem but also makes the physical network more robust. Our OSFCD-LSEM algorithm evaluates the nodes in the physical network to optimize the bandwidth consumption and thus achieves a higher acceptance ratio and shorter response time (i.e., latency) for user demands than the exiting work. In addition, we can make the physical network more robust using our OSFCD-LSEM algorithm. The main contributions of this paper are as follows:

(i) We develop a model to evaluate the physical network and extend it precisely to its "*weak points*" and make the physical network more robust.

(ii) We propose an efficient algorithm (i.e., OSFCD-LSEM) for social IoT oriented SFC deployment that is based on the layered strategies of a physical network and evaluation of a physical network node to optimize the consumption of bandwidth resources and guarantee the security and privacy.

(iii) We conduct extensive simulations to verify and evaluate our algorithm in this paper. The results show that our proposed algorithm can efficiently address the online SFC deployment problem.

The remainder of this paper is organized as follows. Section 2 provides an overview of related works. Section 3 provides the problem descriptions and formulation. In Section 4, we describe our SFC deployment algorithm based on the layered strategies of physical networks and evaluation of physical network nodes. Section 5 shows the simulation results and analysis. Section 6 summarizes this paper.

2. Related Work

NFV aims to satisfy client demands with minimal resource consumptions (e.g., bandwidth consumption and computing core consumption) and high performance (e.g., low latency and high throughput) [20]. An increasing number of studies have focused on the more efficient deployment of social IoT oriented SFC requests in various scenarios.

The research in [39] focused on the suitability of different architectures of data center for a resilient SFC and the placement of VNFs with high availability constraints. In [40], the joint VNF placement and path selection problem were studied. The authors proposed a chain deployment algorithm to balance the chain length and reuse factor; its target was to serve as many demands as possible under limited resources. The research in [41] designed a heuristic algorithm to address the NFV-RA problem in a coordinated manner by splitting the SFC request. In [42], the author made a BCMP mixed queuing network to replace the SFC model and proposed the convex optimization problem to shorten the acceptance interval of the service chain. The simulation results shown the algorithm in [42] can effectively adapt the resource usage to the network dynamics and serve more demands than other existing algorithms. The authors of [43] studied SFC deployment in a multi-domain network and proposed a vertex-centric distributed computing algorithm to find all feasible SFC deployment methods of user requests in the distributed orchestration framework; then, the algorithm selects the most appropriate scheme to deploy the SFC to achieve efficient performance. In [44], the researcher discussed the phenomenon that the VNFs may sprawl across the network due to inefficient mapping during the SFC orchestration process and designed an efficient greedy heuristic algorithm to solve the problem. The author in [45] studied the service availability constraints in a data center and designed a heuristic algorithm to deploy the SFC. In the situation of distributed NFVs, the authors in [46] studied the security level of a network security framework. The authors proposed a new optimization algorithm to avoid a single point of failure in a bottleneck problem and obtained reasonable performance relative to that of a common device. Li et al. [47] presented a real-time resource-distributing system for NFV, which integrates timing analysis with other algorithms, such as timing abstraction, SFC consolidation, and linear programming algorithm, to efficiently distribute network resources while considering latency constraints.

Many researchers have studied the deployment of the social IoT oriented VNF or SFC, and a few of them have attached importance to the total bandwidth consumption. Ye et al. [17] focused on the joint topology design and SFC deployment problem to minimize the total consumption of bandwidth resources. Their main idea was to reduce bandwidth usage and cost by prioritizing VNF deployment. In [48], the authors proposed and analyzed two deployment strategies, HSD and VSD, whose performance was analyzed in terms of cost. They claimed that HSD had better performance in terms of an even distribution of the load over all servers, access links, and ToR switches. The authors in [49] combined NFV with cloud computing and designed a bandwidth-guaranteed SFC-placing algorithm; then, the authors tested the performance of their algorithm in a data center. The simulation results showed that the heuristic algorithm had excellent performance. In [38], the authors designed a forecast-assisted online SFC deployment algorithm that included the prediction of future VNF requirements. The simulation results showed that the algorithm in [38] could reduce the blocking probability of SFC requests and effectively improve the profit of the service provider from SFC deployment.

Overall, bandwidth is the most prominently expensive resource that directly affects the number of demands that the

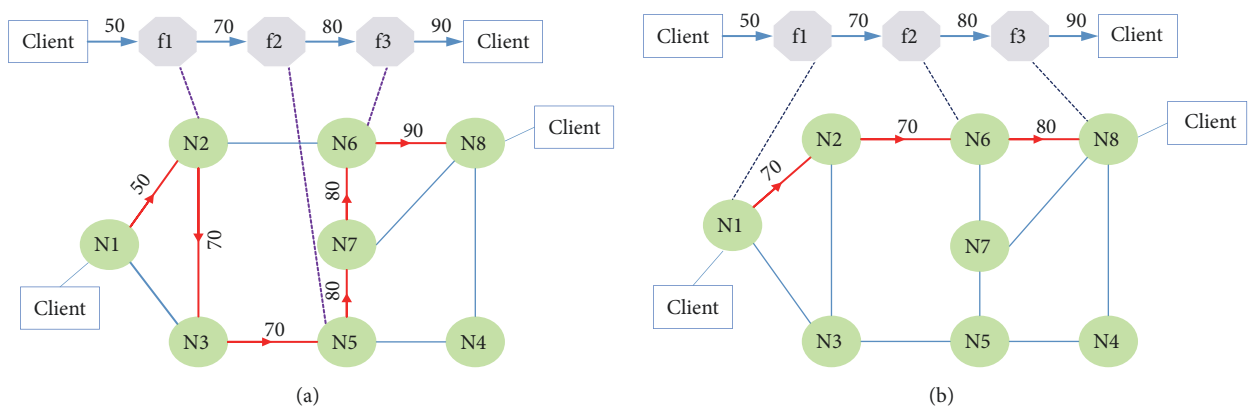


Figure 2: Two examples of SFC deployment.

network can satisfy online and the capacity of the physical network. Thus, in this research, we focused on the bandwidth consumption and designed an efficient algorithm to provision social IoT oriented SFC requests.

3. Problem Formulation

In this paper, we studied the online social IoT oriented SFC deployment problem. We considered a situation in which each SFC request has a pair of start and terminal nodes attached to the given physical network nodes and a specific sequence of VNFs that make up continuous functions. We have to deploy these VNFs onto the corresponding physical nodes and then organize the VNFs to form an SFC. To reduce the bandwidth resource consumption, we need to use fewer nodes to shorten the length of the SFC as much as possible.

A user request can be denoted as $S = (F_S, E_S)$, where $F_S = \{f_1, f_2, \ldots, f_m\}$ is the set of VNFs and $E_S = \{e_1, e_2, \ldots, e_q\}$ represents the virtual link after SFC deployment. The real physical underlying network can be modeled as $G = (N, L)$, where G is an undirected weighted graph, $N = \{N_1, N_2, \ldots, N_y\}$ is the set of physical nodes, and $L = \{L_1, L_2, \ldots, L_k\}$ is the set of real links in the physical underlying network. We use P to denote the physical path that holds the SFC request. We define C_B^T to denote the total bandwidth consumption, which is defined in

$$C_B^T = \sum_{L_{i\_e_j} \in P\_\{S\}} C_B^{L_{i\_e_j}} \tag{1}$$

where L_{i_ej} denotes that virtual link, e_j is deployed on the link L_i, and $C_B^{L_{i\_ej}}$ is the bandwidth consumption of virtual link e_j, which is deployed on the link L_i. We define $R_C^{N_i}$ as the available computing resources of the physical node N_i and $C_N^{N_{i\_fj}}$ as the computing resource requirements of VNF f_j, which would be deployed on the node N_i. $R_B^{L_i}$ is the available bandwidth resources of the physical link L_i.

To deploy an SFC, we must map the VNFs to some node and the virtual links E_S to some real links in the physical network; the path $P\_\{S\}$, which would hold the SFCs, must have sufficient nodes and computing resources to deploy the corresponding VNFs, and the physical links must have sufficient available bandwidth for communication among those VNFs. In addition, the available bandwidth resources must satisfy the requirements of the E_S in the corresponding user request. Similar to [10, 20], this paper assumes that each VNF from the same SFC needs to use different physical network nodes. Then, the number of VNFs (denoted as $\|Fs\|$) should be less than the number of nodes of physical path $P\_\{S\}$ (denoted by $\|P\_\{S\}\|$). Then, the problem of SFC deployment can be formulated as follows:

$$\begin{aligned} \min \quad & \sum_{L_{i\_e_j} \in P\_\{S\}} C_B^{L_{i\_e_j}} \\ s.t. \quad & \forall L_{i\_e_j} \in P\_\{S\} \\ & R_B^{L_i} - C_B^{L_{i\_e_j}} \geq 0 \\ & \forall N_{i\_f_j} \in P\_\{S\} \\ & R_C^{N_i} - C_N^{N_{i\_f_j}} \geq 0 \\ & \|P\_\{S\}\| - \|Fs\| \geq 0 \end{aligned} \tag{2}$$

Formula (2) is used to model the SFC deployment problem, which can minimize the bandwidth consumptions while finishing SFC deployment. There must be sufficient available computing resources to deploy corresponding SFCs, and the bandwidth must be sufficient to satisfy the communication demands among the corresponding VNFs. In addition, there must be sufficient nodes in the physical path P to deploy the VNFs of the SFC request.

Figure 2 shows two different examples of provisioning an SFC request, both of which successfully deploy the SFC request and satisfy the clients' demands. Nodes *N1* and *N8* are the requested client node and destination client node, respectively. There are three VNFs (i.e., f_1, f_2, and f_3) in the SFC request. Client node *N1* needs to transmit 50 units

of traffic to VNF f_1. Between VNFs f_1 and f_2, 70 units of information need to be transmitted. Between VNFs f_2 and f_3, 80 units of traffic need to be transmitted. There is a 90-unit transmission demand from VNF f_3 to the destination client node *N8*. As shown in Figure 2(a), we deploy the VNFs f_1, f_2, and f_3 onto the physical nodes *N2*, *N5* and *N6*, respectively. Then, we find the path $P = \{N1\text{-}N2, N2\text{-}N3, N3\text{-}N5, N5\text{-}N7, N7\text{-}N6, N6\text{-}N8\}$ to host the SFC. In this deployment scheme, the total bandwidth consumption is 440 units. Although this approach successfully deploys the SFC request, it wastes bandwidth resources. In Figure 2(b), the VNFs f_1, f_2, and f_3 are deployed onto the physical nodes *N1*, *N6*, and *N8*, respectively. Then, we find a short path $P = \{N1\text{-}N2, N2\text{-}N6, N6\text{-}N8\}$ (shown as the red line in Figure 2(b)) which can also deploy all VNFs from the user requests. However, the total consumption of bandwidth resources of this scheme is only 220 units, which is approximately half of the consumption of the deployment in Figure 2(a). In Figure 2(b), path P is the shortest path to deploy this SFC request, and this scheme can reduce more bandwidth consumption. Mapping the SFC to the path in Figure 2(b), the service network can deploy more SFC with other links, which is impossible in Figure 2(a).

Therefore, different deployment schemes significantly affect the bandwidth consumptions and thus affect the available network capacity. An efficient algorithm is important and urgently needed to better deploy SFC requests and reduce bandwidth consumption.

4. Algorithm Design

Since the optimal SFC deployment problem is NP-hard [35, 37], we design an efficient algorithm for solving our research problem in this section. Our proposed algorithm employs the layered strategies of physical networks and evaluation of physical network nodes to optimize the consumption of bandwidth resources, which is denoted by OSFCD-LSEM. The basic idea of OSFCD-LSEM is to find the shortest paths to deploy SFC requests while saving as many bandwidth resources as possible to satisfy more user requests. When a user demand arrives, the OSFCD-LSEM algorithm begins to handle it. First, calling Algorithm 2, the OSFCD-LSEM algorithm layers the physical underlying network and obtains the information from the nodes and links in each layer of the physical network. Then, it calls Algorithm 3 to perform an evaluation of nodes in the network and select some nodes that are most suitable to host the VNFs of this SFC request. Finally, it connects the deployed VNFs by using a shortest path to fulfill an SFC. By layered strategies and selection of the nodes for VNF deployment, the OSFCD-LSEM algorithm can deploy VNFs in the most suitable nodes and deploy the SFC requests in appropriate simple paths to save more bandwidth resources. The physical path must contain the start node N_r and the terminal node N_a. Furthermore, the nodes in the physical path must have sufficient resources to host the VNFs of the user request.

In the proposed OSFCD-LSEM algorithm, we provide some definitions of variables. G_L is the physical network after layering, and V_X is used to denote the set of nodes in the *X-th* layer. The *X-th* layer is denoted as *L.X*. G_L^X is the inner layered network in the *X-th* layer (*L.X*), and we used *L.Y* to denote the *Y-th* layer in G_L^X. $V_{(X,Y)}^i$ is the set of nodes in the *L.Y* of G_L^X, which includes the node N_i. E_X is the set of links that connect the nodes from *L.X* and *L.X*-1. $E_{(X,Y)}^i$ denotes the corresponding links that connect the nodes in the *L.Y*-1 about node N_i, and L_{MAX}^i is the corresponding maximal layer of node N_i. L_{MAX} is the layer number of G_L. N_T is the node number of the physical network, and L_T is the link number of the physical network G. The OSFCD-LSEM algorithm is shown in Algorithm 1.

Next, Algorithm 2 is responsible for handling hierarchical physical networks in our algorithm. Algorithm 2 layers the entire network to obtain the layering information of the nodes and links in the network and outputs the results to other algorithms. Therefore, Algorithm 2 is the basis of our SFC request deployment scheme and physical network node evaluation. The physical network layering can be formulated as follows:

$$G_L = \sum_{X=1}^{L_{MAX}} (V_X, E_X) + \sum_{X=2}^{L_{MAX}} G_L^X \tag{3}$$

$$G_L^X = \sum_{V_i \in V_X} \sum_{Y=1}^{L_{MAX}^i} \left(V_{(X,Y)}^i, E_{(X,Y)}^i\right) \tag{4}$$

$$\sum_{X=1}^{L_{MAX}} V_X - N_T \geq 0 \tag{5}$$

$$\sum_{X=2}^{L_{MAX}} E_X + \sum_{X=2}^{L_{MAX}} \sum_{V_i \in V_X} \sum_{Y=2}^{L_{MAX}^i} E_{(X,Y)}^i - L_T = 0 \tag{6}$$

In (3), two parts make up G_L: one part is the inner layer network G_L^X about the X-*th* layer (*L.X*) and the other part is the overall layered network. The layering process begins from the request node N_r; thus, $V_1 = N_r$, $E_1 = \emptyset$, and $G_L^1 = \emptyset$. Equation (4) shows that, to make G_L closer to the physical network G, each layer except layer *L*.1 should obtain its inner layer node-related information. The OSFCD-LSEM algorithm can obtain a more precise evaluation of the physical network and more efficiently deploy the corresponding SFC requests. In (5), the equation describes that all nodes in the layered physical network must be in the corresponding layer. In (6), the equation shows that each link should be in the corresponding layer or inner layer.

In Figure 3, we give an example of layering a network. Figure 3(a) shows the topology of a physical network, and it has a total of 10 nodes. Figure 3(b) provides the detailed processing procedure for layering the network topology. We assume that the start node N_r in the request is node *N1* and that the terminal node N_a in the request is node *N9*. First, our algorithm places the start node *N1* into *L*.1 (*N1* is the only node in V_1 of *L*.1) and places nodes *N2*, *N3*, and *N4* into L.2 because they directly have links to node *N1*. Then, our algorithm places nodes *N5*, *N6*, and *N9*, all of which have links to some nodes in the *L*.2 (i.e., nodes *N2*, *N3*, and *N4*),

Input: (1) SFC request. (2) physical network G;
Output: SFC deployment scheme.
(1) Receive a user request;
(2) Initialize $Path = []$;
(3) $N_a \longrightarrow Path$; $N_L = N_a$;
(4) Layer the topology: Algorithm 2(N_r; N_L; G);
(5) Get L_A: the layers that destination client belongs to;
(6) **while** $L_S > \max(L_A) + \sum_{X=1}^{L_{MAX}} \max(L_{MAX}^i)\ \forall N_i \in V_X$ **do**
(7) **if** $\max\{L_A\} = L_{MAX}$
(8) N_{TEMP} = Algorithm 3(; ; true; $\max\{L_A\}$);
(9) $N_{TEMP} \longrightarrow Path$;
(10) $N_L = N_{TEMP}$;
(11) **else**
(12) N_{TEMP} = Algorithm 3(; ; false; $\max\{L_A\}$);
(13) $N_{TEMP} \longrightarrow Path$;
(14) $N_L = N_{TEMP}$;
(15) **end if**
(16) $L_S = L_S - 1$;
(17) VNF $\longrightarrow N_{TEMP}$;
(18) Algorithm 2(N_r; N_L; G);
(19) **Update** L_A;
(20) **end while**
(21) **if** $L_S \leq \max(L_A)$
(22) Select min $L.X \in L_A$ && $L.X > L_S$;
(23) **while** $N_r \notin Path$ **do**
(24) N_{TEMP} = Algorithm 3 (; ; true; $L.X$);
(25) $N_{TEMP} \longrightarrow Path$;
(26) $N_L = N_{TEMP}$;
(27) $L.X = L.X - 1$;
(28) VNF$\longrightarrow N_{TEMP}$;
(29) **end while**
(30) **end if**
(31) SFC deployment scheme is *Path*.

Algorithm 1: OSFCD-LSEM algorithm.

into $L.3$ (we specify that all nodes can only belong to one layer except the terminal node *N9*; thus, node *N4* cannot be part of $L.3$, even though it connects with node *N3*, which is in $L.2$).

In our layered network, except for the destination client node *N9*, the nodes in one layer must have links to the nodes in the previous layer. Thus, nodes *N7*, *N8*, and *N10* directly have links to the nodes in $L.3$ (i.e., nodes *N5*, *N6*, and *N9*), whereas *N10* connects only with destination client node *N9* among the three nodes in $L.3$. Node *N10* should not be placed in layer $L.4$. Thus, we place only nodes *N7* and *N8* into $L.4$. Nodes *N9 and N10* connect with nodes *N7 and N8* in $L.4$; thus, we place nodes *N9* and *N10* in layer $L.5$, and because *N9* connects with node *N10*, our algorithm places node *N9* in $L.6$. When ten nodes in G belong to the corresponding layers, the overall network-layered processing finishes. Thus, the OSFCD-LSEM algorithm can guarantee finding a path without loops. For each $L.X$, we also need to layer and obtain the inner layer G_L^X. In the example shown by Figure 3(b), the layer $L.2$ has an inner layer G_L^2 that includes two layers. For G_L^2 in Figure 3(b), each layer $X \leq L_{MAX}$ and each node $N_i \in V.X$ should be set as the start node N_r; let $N_a = \emptyset$. Then, we obtain their inner layer information. In G_L^2, both L_{MAX}^{N3} and L_{MAX}^{N4} are equal to 2, whereas L_{MAX}^{N2} is equal to 1.

Overall, the physical network is layered into six layers with two inner layer networks associated with nodes *N3* and *N4*. The start node N_r is the only one in the first layer, and the terminal node N_a is in three layers, including $L.3$, $L.5$, and $L.6$. Thus, we obtain three paths that can be used between N_r and N_a. L_P is used to denote the length of path, which is equal to the number of VNFs that the path can hold. L_S is the length of an SFC, which is equal to the VNF number of the SFC request.

Algorithm 3 evaluates the nodes in the physical network and selects a node which is most suitable to deploy the required VNF. After obtaining the layering information, Algorithm 3 makes the decision regarding whether the multiple links can satisfy the user request. When the maximal layer number in L_A of the terminal nodes N_a, which is obtained from the sum of all inner layers, is smaller than the SFC length L_S in the user request, the physical network cannot satisfy the user request. For example, if we need to deploy an SFC into the physical network in Figure 3(a), the start and terminal nodes are *N1* and *N9*. The maximum layer number in L_A is

Input: (1) N_r; (2) N_a; (3) physical network G.
Output: G_L;
(1) $N_r \longrightarrow V_1$; $L_{MAX} = L.1$;
(2) **for** $V_{L_{MAX}} \neq \emptyset$; $N_m \neq N_a$; **do**
(3) **for** each $N_n \in G$; **do**
(4) **if** $N_m \longleftrightarrow N_n$ && $N_n \notin \sum_1^{L_{MAX}} V_X$
(5) $N_n \longrightarrow V_{L_{MAX+1}}$;
(6) **else if** $N_m \longleftrightarrow N_n$ && $N_n \in \sum_1^{L_{MAX}} V_X$&&$N_n = N_a$
(7) $N_n \longrightarrow V_{L_{MAX+1}}$;
(8) **end if**
(9) **end for**
(10) L_{MAX} ++;
(11) **end for**
(12) **for** $L.X \leq L_{MAX}$; **do**
(13) **for** $N_m \in V_X$; **do**
(14) $N_m \longrightarrow V_{(X,1)}$;
(15) $L^m{}_{MAX} = L^m.1$;
(16) **for** $V_{(X,L^m{}_{MAX})} \neq \emptyset$; **do**
(17) **if** $N_n \in V_X$&&$N_m \longleftrightarrow N_n$&&$N_n \notin \sum_1^{L^m{}_{MAX}} G_L^X$
(18) $N_n \longrightarrow V_{(X,L^m{}_{MAX})}$;
(19) **end if**
(20) $L^m{}_{MAX}$ ++;
(21) **end for**
(22) **end for**
(23) **end for**
(24) **return** G_L

Algorithm 2: Network layered processing.

Input: (1) SFC request;
(2) G_L;
(3) **bool** *direction*;
(4) **X:** $N_L \in V_X$;
Output: the node N_C which has the minimum value of δ;
(1) Temp = $+\infty$
(2) int i= 0;
(3) **if** *direction* is true
(4) i= X-1;
(5) **end if**
(6) **if** *direction* is false
(7) i= X+1;
(8) **end if**
(9) **for** $N_m \in V_i$ **do**
(10) **if** $N_m \longleftrightarrow N_L$
(11) **if** $B_{si\_m} > B_{ri\_m}$&&$B_{se\_m} > B_{re\_m}$ && $C_{s\_m} > C_r$;
(12) Compute δ according to Equation (7);
(13) **if** $\delta <$ Temp
(14) Temp = δ;
(15) $N_C = N_m$;
(16) **end if**
(17) **end if**
(18) **end if**
(19) **end for**
(20) **return** N_C.

Algorithm 3: Evaluation of related nodes.

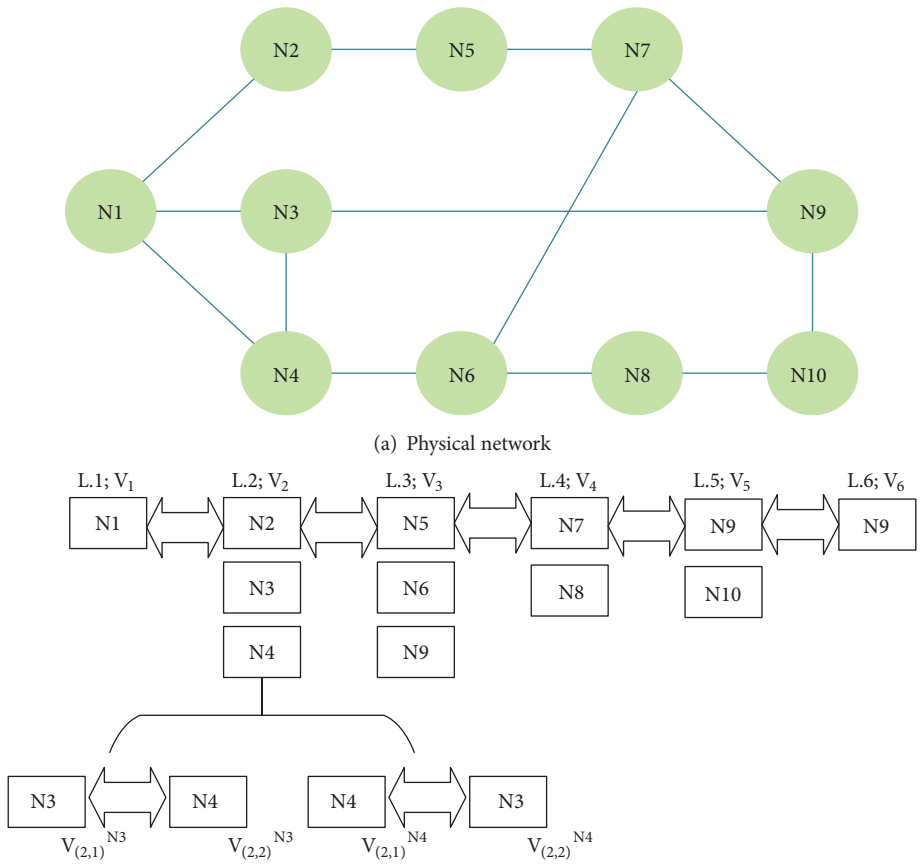


Figure 3: An example of a layered physical network.

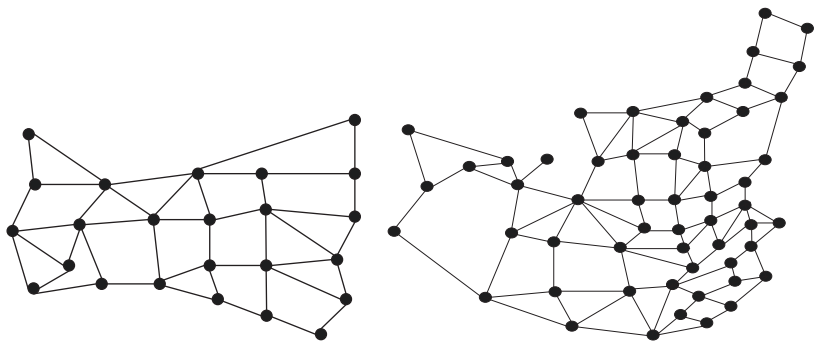
6, and there is an inner layer in layer $L.2$. The total sum is 7, so this network topology can satisfy only an SFC request whose length is no more than 7. An SFC request whose number of VNFs is more than 7 is too large for this network. However, as long as the number of VNFs is not greater than the number of nodes in the network, our OSFCD-LSEM algorithm can try to find more paths to deploy the longer SFC. It may need additional time and capacity of bandwidth resources because there are too many VNFs that need to be deployed. The proposed Algorithm 3 can search nodes in both positive and negative directions. When addressing an SFC that is longer than the abovementioned sum, Algorithm 3 commonly finds a suitable node in the layer V_{L_A+1} instead of in the upper layer V_{L_A-1}; then, the algorithm can increase the length of the path (L_P). However, an extreme situation should be considered, such as when the node is in the last layer $L.L_{MAX}$. For this situation, our algorithm will find a node in the upper layer $V_{L_{MAX}-1}$ and run Algorithm 2 again.

Finally, we must evaluate the nodes from each layer of the layered network and select some nodes to map the VNFs. Algorithm 3 uses the abovementioned strategy to find a path from N_a to N_r to host an SFC request and satisfy the user demand. Algorithm 3 selects the nodes from each layer using (7) and (8). The selected node must directly link to the node in the next layer V_N and must have sufficient resources to deploy the VNFs and satisfy the corresponding function requirements.

$$\delta = \min\left(\frac{B_{si\_m} - B_{ri\_vnf_{i,j}}}{B_{si\_m}}, \frac{B_{se\_m} - B_{re\_vnf_{i,j}}}{B_{se\_m}}\right) \times \frac{C_{s\_m} - C_{r\_vnf_{i,j}}}{C_{s\_m}} \tag{7}$$

$$C_{s\_m} = C_{total\_m} - \sum_{S_i \in SFC\_online} \sum_{vnf_{i,j} \in S_i} C_{r\_vnf_{i,j}} \times \alpha_{vnf_{i,j}\_m} \tag{8}$$

In (7), we use δ to denote the node's appropriateness for the VNF in the user request. B_{si_m} is the idle bandwidth of all links that connect node N_m with the nodes in the next layer V_N, and $B_{ri\_vnf_{i,j}}$ is the requested bandwidth resource from $vnf_{i,j}$ to the $vnf_{i,j+1}$, where $vnf_{i,j}$ is the j-th VNF in the S_i (i.e., the i-th SFC request). B_{se_m} is the idle bandwidth of the path that connects node N_m with the nodes in the upper layer V_U, and $B_{re\_vnf_{i,j}}$ is the requested bandwidth resource for making $vnf_{i,j}$ connect with the last VNF. C_{s_m} represents the idle computing resources of node N_m, and $C_{r\_vnf_{i,j}}$ represents the requested computing resources of the $vnf_{i,j}$. In (8), C_{total_m} represents the total computing resources of node N_m, SFC_{online} is the set of online SFC requests, and



(a) US-Net topology (b) China-Net topology

Figure 4: Two real network topologies used in our simulations.

$\alpha_{vnf_{i,j}\_m}$ denotes whether $vnf_{i,j}$ is deployed on node N_m. We can evaluate whether a physical node is suitable to deploy the corresponding VNF according to the value of δ. After physical node evaluation, we select the node with the minimal value of δ to hold the corresponding VNF.

5. Simulation Results and Analysis

To evaluate the performance of our algorithm, we compare our algorithm with two existing algorithms, i.e., closed-loop with critical mapping feedback (CCMF) [17] and key-VNF deploy first (KVDF) [38]. CCMF deploys the VNFs that have more resource requirement priorities to optimize the total consumption of bandwidth resources. KVDF focuses on the relation between different VNFs in the same SFC and the relation among different SFCs. According to the relation and influence, KVDF prioritizes the deployment of the key-VNF and key-SFC.

To evaluate the performance of algorithms in different network scenarios, we evaluate the performance of compared algorithms in moderate-scale and large-scale networks, the US-Network [50] (shown in Figure 4(a)), and the China-Network [51] (shown in Figure 4(b)). We use GT-ITM [52] to generate the moderate-scale and large-scale network topologies. In moderate-scale network topologies, there are approximately 100 nodes and 400 links. In large-scale network topologies, there are approximately 300 nodes and 1000 links. The US-Net topology has 24 nodes and 43 links. Additionally, the China-Net topology has 55 nodes and 103 links. In these network topologies, the bandwidth resource of all links is uniformly distributed at 100~200 units and the computing resource of all nodes is set as 10 units.

In our simulations, we set the following parameters according to the existing work [49]. The computer randomly generates user requests with lengths (i.e., Ls) from 5 to 14, and these online SFC requests arrive as a Poisson process. For each physical network and for a given L_s, we randomly generate 10,000 SFC requests with the request client and destination client nodes, which are randomly assigned to the physical nodes. The computing resource demand of each VNF follows a uniform distribution U (1, 3), and the bandwidth resource demand of each virtual link follows a uniform distribution U (20, 50).

With the increasing number of users and SFC requests, the deployment of SFC in a static network becomes increasingly challenging. Thus, the network scalability must be improved. For a network-aware scaling strategy, it is better to extend the network instead of changing the network. In network G, we define the evaluation of information G^S in

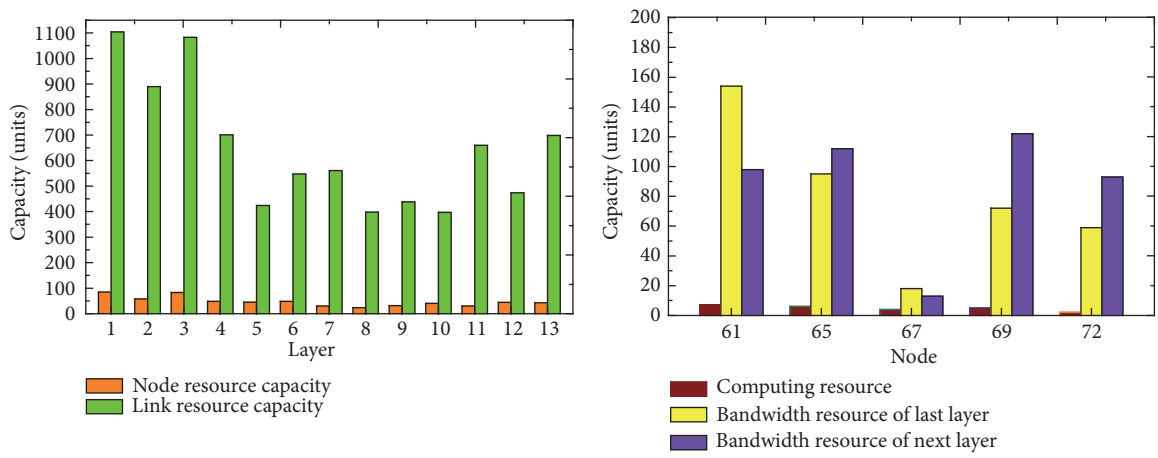
$$G^S = \sum_{X=1}^{L_{MAX}} \sum_{N_i \in V_X} \left(C_s + B_{si} + B_{se}\right) \tag{9}$$

The OSFCD-LSEM algorithm layers the physical network, obtains the layer which has minimum resources, analyzes its inner layer information, and obtains the "*weak nodes* or *links* that influence the capacity of the network. Then, the OSFCD-LSEM algorithm extends their resources to secure a more robust network.

We obtained the simulation results by averaging the results of multiple simulations. We used an Ubuntu virtual machine running on a computer with a 3.7 GHz Intel Core i3-4170 and 4 GB of RAM to run the simulations. And the algorithms were coded in Java programming language.

From Figure 5, we can see the simulation results of our OSFCD-LSEM algorithm in a moderate-scale network. Figure 5(a) shows the information of the physical network. We can see that $L.8$ limits the network capacity and will affect the deployment of SFC. Figure 5(b) shows the information of the physical nodes in $L.8$. In $L.8$, node-67 has the minimal amount of bandwidth, and node-72 has the minimal amount of computing resources. Both node-67 and node-72 are the "*weak points*" in the physical network. If we can increase their corresponding resources, the capacity of the physical network will be enhanced. For network operators, increasing the corresponding resources to the corresponding nodes and links is necessary and highly beneficial; moreover, they can also obtain a more robust network.

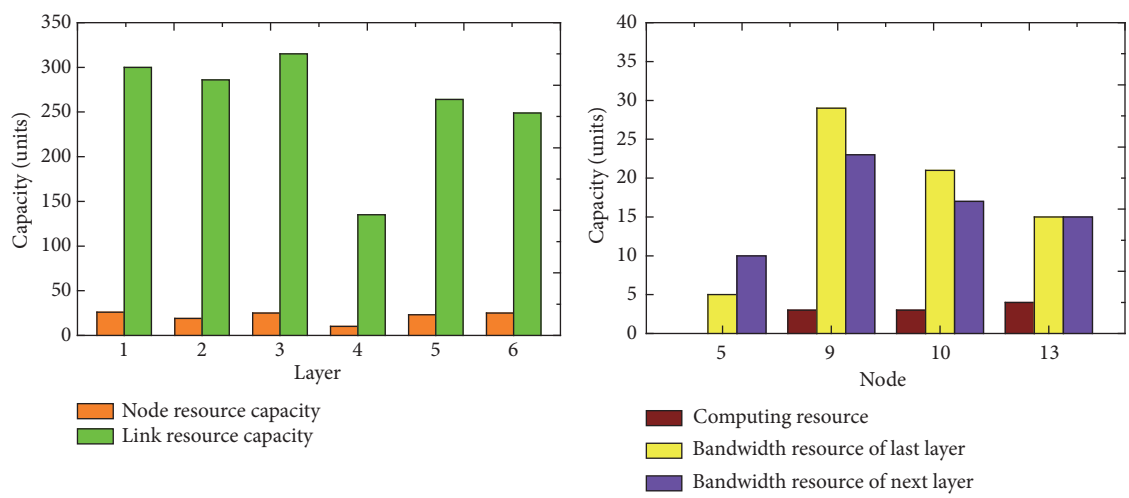
Figure 6 shows the simulation results on the US-Net network. Figure 6(a) shows the information about each layer in the overall network, and we find that $L.4$ may be the "*weak point*" of the physical network because of the lack of computing and bandwidth resources. Figure 6(b) shows



(a) The resource capacity of each layer

(b) The information of the physical nodes in $L.8$

Figure 5: Simulation results for the moderate-scale network.



(a) The resource capacity of each layer

(b) The detailed information of nodes in $L.4$

Figure 6: Simulation results for the US-Net network.

the nodes' detailed information about $L.4$. In the evaluation result, node 5 has no computing resources to carry any VNF. This means that node 5 is a "*dead point*" in the network, and it significantly compromises network performance. Moreover, there are a few bandwidth resources in node 5 for connecting the nodes in the last layer and the next layer. It is necessary and urgent to supplement the corresponding bandwidth resources to make the network more robust. With respect to node 13, though there are enough computing resources, there are insufficient bandwidth resources. Thus, increasing bandwidth resources in node 5 to connect the nodes in $L.3$ and $L.4$ will significantly improve the ability of the physical network to provision more SFC requests and make the network more robust.

For network operators and our OSFCD-LSEM algorithm, those "*weak points*" serve as the focal points for the extension of the physical network. Relative to an approach that blindly extends the physical network to all nodes and links without focus and direction considerations, the extension implemented by the OSFCD-LSEM algorithm is more accurate and efficient. Our OSFCD-LSEM algorithm can be used to supply resources to the nodes that require the most resources, and it avoids wasting resources while improving the total physical network capacity.

Figure 7 shows the simulation results of the acceptance ratios of the OSFCD-LSEM algorithm and the other two algorithms. Figures 7(a), 7(b), 7(c), and 7(d) show the comparable results in the moderate-scale, large-scale, US-Net, and China-Net networks, respectively. The OSFCD-LSEM algorithm has a higher SFC request acceptance ratio than the CCMF and KVDF algorithms in all networks. As a result, the OSFCD-LSEM can find the appropriate nodes for VNF deployment based on the client's direction with the help of the network layering algorithm. Compared with the other

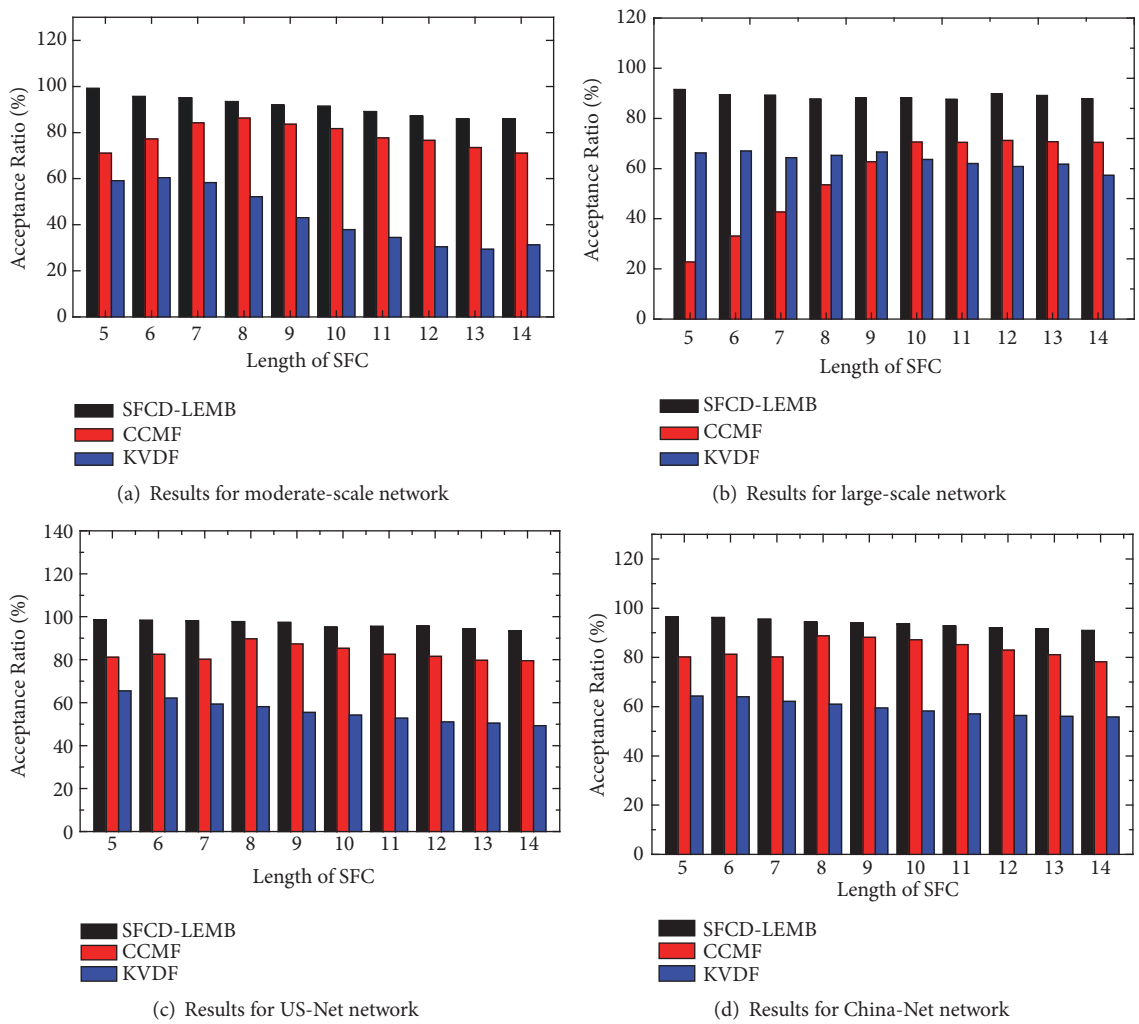
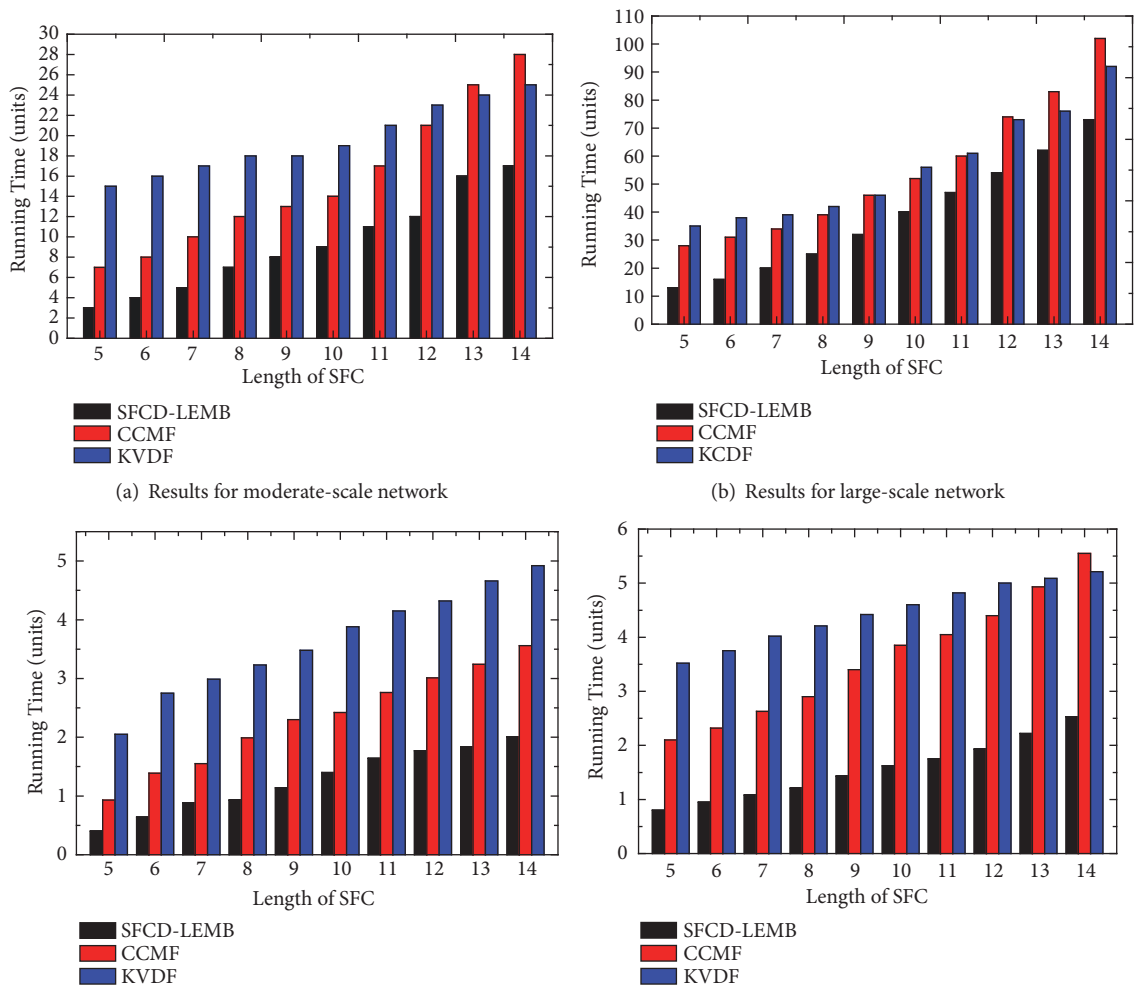


Figure 7: Acceptance ratios in different physical networks.

two algorithms, OSFCD-LSEM always finds the shortest and most appropriate path to deploy SFCs; thus, it has the highest acceptance ratio among the algorithms. Furthermore, the OSFCD-LSEM algorithm has a relatively stable acceptance ratio in arbitrary scale networks and different L_s because our OSFCD-LSEM algorithm evaluates the network after layering part of the network and can appropriately finish the SFC deployments. In addition, the OSFCD-LSEM algorithm performs better in both networks that are generated by GT-ITM and in the real networks. The excellent performance is achieved because it is based on the evaluation of the layered network.

Figure 8 shows the simulation results of the running time of the OSFCD-LSEM algorithm and the other two algorithms. Figures 8(a), 8(b), 8(c), and 8(d) reveal the comparable results in the moderate-scale, large-scale, US-Net, and China-Net networks, respectively. Among the three algorithms, the running time of the OSFCD-LSEM algorithm for performing the deployment is shortest because it can find nodes for VNF deployment based on the appropriate search direction rather than on random searching. Therefore, the OSFCD-LSEM can find the path towards the client node within fewer searching steps and shorter searching time compared with the other algorithms. Moreover, the optimization of the OSFCD-LSEM algorithm makes its running time slowly increase with the increasing length of SFC (L_s). As L_s increases, OSFCD-LSEM requires only a few searching steps, and the running time slowly increases due to the directional search advantage.

Figures 9(a), 9(b), 9(c), and 9(d) show the simulation results of the consumptions of bandwidth resources in the moderate-scale, large-scale, US-Net, and China-Net networks, respectively. Figure 9 shows that the OSFCD-LSEM algorithm can provision SFC requests with less bandwidth consumption than the other two algorithms for the same SFC requests. Since it employs the network layering strategy, the shortest paths can be found by the OSFCD-LSEM algorithm to deploy the SFC based on an efficient search direction. Then, compared with the other two algorithms, SFC communication via the shortest paths consumes less bandwidth resources. With the increase in L_s and network size, the OSFCD-LSEM algorithm shows outstanding



(a) Results for moderate-scale network

(b) Results for large-scale network

(c) Results for US-Net network

(d) Results for China-Net network

Figure 8: Running time in different physical networks.

performance in saving bandwidth resources because it obtains the layering information of the nodes and links in the network by layered strategies, which is one of the main contributions of OSFCD-LSEM. Importantly, since saving bandwidth resources is the goal of this study, the simulations of different network scenarios and SFC lengths (L_s) show that the OSFCD-LSEM algorithm obtains the expected results.

In summary, the OSFCD-LSEM algorithm layers the physical network and is superior to the other two compared algorithms in terms of the acceptance ratio, running time, and bandwidth consumption. These achievements can be attributed to the manner in which the OSFCD-LSEM algorithm obtains overall evaluation information of the physical network.

6. Conclusion

Under NFV technology, the network functions can be migrated from dedicated hardware and be deployed onto commercial servers in any necessary location. NFV can remarkably enhance the flexibility and reduce the resource consumption in the physical network. With the benefit of NFV, the clients' experience and network performance are both improved.

In this paper, we study the efficient online social IoT application oriented SFC provision problem. We propose an SFC deployment algorithm, OSFCD-LSEM, which layers the physical network to obtain the layering information of the nodes and links in the network. Moreover, it also selects the most suitable node to host the VNFs by evaluating some nodes in the physical network. The simulation results show that the OSFCD-LSEM algorithm has better performance in terms of time efficiency, acceptance ratio, and bandwidth consumption for provisioning social IoT application oriented SFC requests. Furthermore, to satisfy the increasing social IoT demands, we can use the layering information to appropriately extend the physical network.

In the future work, we can introduce artificial intelligence algorithm to the existing framework to improve the accuracy rate, or to study the algorithm to run efficiently in a more complex physical network environment.

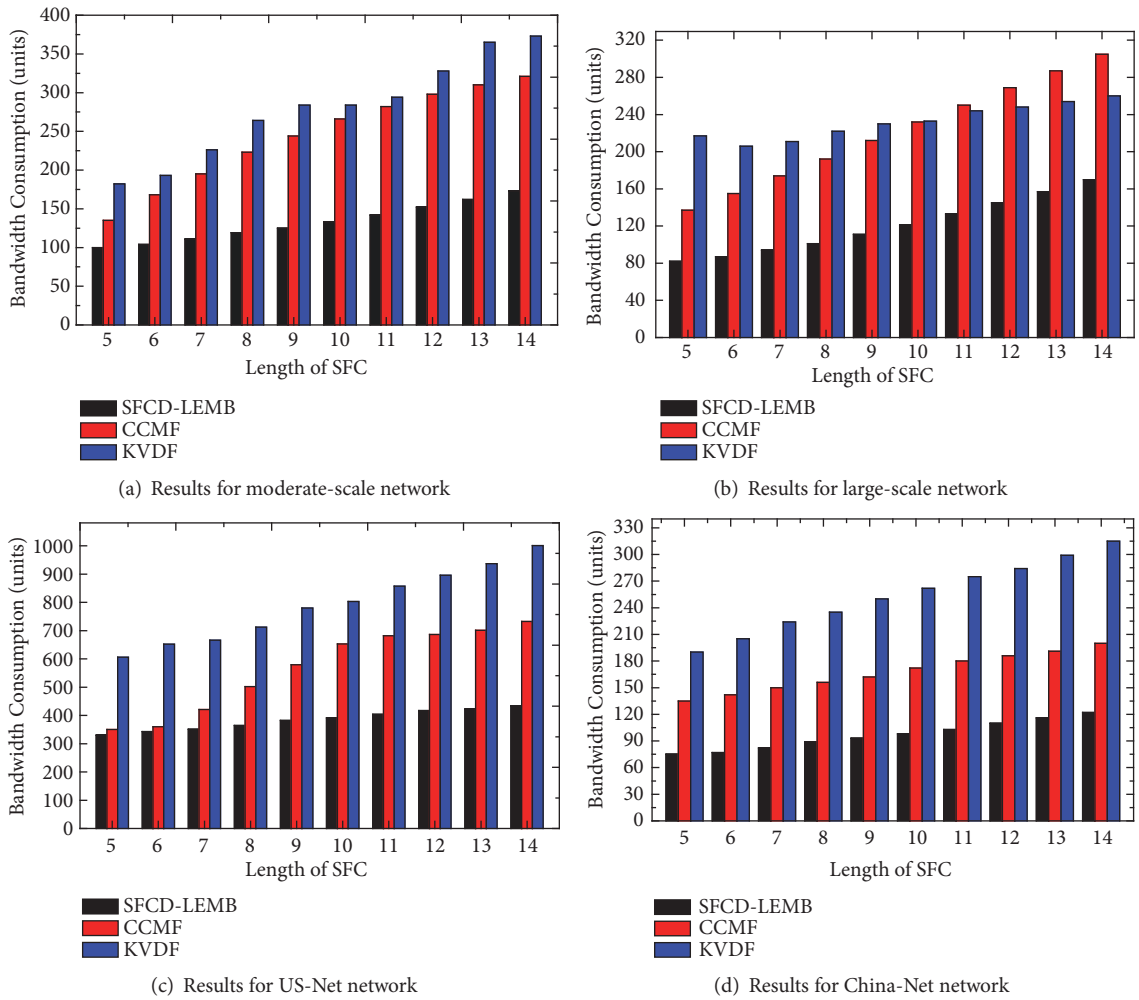


Figure 9: Bandwidth consumptions in different physical networks.

Data Availability

The data supporting the results reported in this work is generated in our simulation experiments in our study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was partially supported by the National Natural Science Foundation of China (61571098), Sichuan Science and Technology Program (2019YFG0206), and the 111 Project (B14039).

References

[1] M. S. Yoon and A. E. Kamal, "NFV resource allocation using mixed queuing network model," in *Proceedings of the IEEE Global Communications Conference*, pp. 1–7, 2016.

[2] G. Sun, S. Sun, J. Sun, H. Yu, X. Du, and M. Guizani, "Security and privacy preservation in fog-based crowd sensing on the internet of vehicles," *Journal of Network and Computer Applications*, vol. 134, pp. 89–99, 2019.

[3] G. Sun, V. Chang, M. Ramachandran et al., "Efficient location privacy algorithm for Internet of Things (IoT) services and applications," *Journal of Network and Computer Applications*, vol. 89, pp. 3–13, 2017.

[4] Q. Du, H. Song, and X. Zhu, "Social-feature enabled communications among devices toward the smart iot community," *IEEE Communications Magazine*, vol. 57, no. 1, pp. 130–137, 2019.

[5] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, "Joint computation offloading and user association in multi-task mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 12, pp. 12313–12325, 2018.

[6] G. Sun, D. Liao, H. Li, H. Yu, and V. Chang, "L2P2: A location-label based approach for privacy preserving in LBS," *Future Generation Computer Systems*, vol. 74, pp. 375–384, 2017.

[7] G. Sun, L. Song, D. Liao, H. Yu, and V. Chang, "Towards privacy preservation for "check-in" services in location-based social networks," *Information Sciences*, vol. 481, pp. 616–634, 2019.

[8] H. Song, G. A. Fink, and S. Jeschke, *Security and Privacy in Cyber-Physical Systems: Foundations, Principles and Applications*, Wiley-IEEE Press, 2017.

[9] X. Huang, Y. Lu, D. Li, and M. Ma, "A novel mechanism for fast detection of transformed data leakage," *IEEE Access*, vol. 6, pp. 35926–35936, 2018.

[10] G. Sun, Y. Xie, D. Liao, H. Yu, and V. Chang, "User-defined privacy location-sharing system in mobile online social networks," *Journal of Network and Computer Applications*, vol. 86, pp. 34–45, 2017.

[11] R. Cohen, L. Lewin-Eytan, J. S. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *Proceedings of the IEEE Conference on Computer Communications*, pp. 1346–1354, 2015.

[12] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *Proceedings of the IEEE 4th International Conference on Cloud Networking*, pp. 171–177, 2015.

[13] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On Dynamic service function chain deployment and readjustment," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 543–553, 2017.

[14] S. Mehraghdam, M. Keller, and H. Karl, "Specifying and placing chains of virtual network functions," in *Proceedings of the IEEE 3rd International Conference on Cloud Networking*, pp. 7–13, 2014.

[15] X. Wang, C. Wu, F. Le et al., "Online VNF scaling in datacenters," pp. 1-9, 2016.

[16] G. Sun, G. Zhu, D. Liao, H. Yu, X. Du, and M. Guizani, "Cost-efficient service function chain orchestration for low-latency applications in NFV networks," *IEEE Systems Journal*, pp. 1–13.

[17] Z. Ye, X. Cao, J. Wang, H. Yu, and C. Qiao, "Joint topology design and mapping of service function chains for efficient, scalable, and reliable network functions virtualization," *IEEE Network*, vol. 30, no. 3, pp. 81–87, 2016.

[18] S. Clayman, E. Maini, A. Galis et al., "The dynamic placement of virtual network functions," *IEEE Network Operations and Management Symposiu*, pp. 1–9, 2014.

[19] G. Sun, V. Chang, G. Yang, and D. Liao, "The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion," *Information Sciences*, vol. 432, pp. 495–515, 2018.

[20] P. S. Khodashenas, B. Blanco, M.-A. Kourtis et al., "Service mapping and orchestration over multi-tenant cloud-enabled RAN," *IEEE Transactions on Network and Service Management*, vol. 14, no. 4, pp. 904–919, 2017.

[21] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[22] X. Du and H. H. Chen, "Security in wireless sensor networks," *IEEE Wireless Communications Magazine*, vol. 15, no. 4, pp. 60–66, 2008.

[23] X. Du, Y. Xiao, M. Guizani, and H.-H. Chen, "An effective key management scheme for heterogeneous sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 24–34, 2007.

[24] H. Song, R. Srinivasan, T. Sookoor et al., *Smart Cities: Foundations, Principles and Applications*, Wiley, 2017.

[25] X. Du, M. Guizani, Y. Xiao et al., "A routing-driven elliptic curve cryptography based key management scheme for heterogeneous sensor networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 3, pp. 1223–1229, 2009.

[26] L. Liu, C. Chen, T. Qiu, M. Zhang, S. Li, and B. Zhou, "A data dissemination scheme based on clustering and probabilistic broadcasting in VANETs," *Vehicular Communications*, vol. 13, pp. 78–88, 2018.

[27] X. Li and C. Qian, "Low-complexity multi-resource packet scheduling for network function virtualization," in *Proceedings of the IEEE Conference on Computer Communications*, pp. 1400–1408, 2015.

[28] G. Sun, Y. Li, D. Liao, and V. Chang, "Service function chain orchestration across multiple domains: A full mesh aggregation approach," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1175–1191, 2018.

[29] G. Xiong, Y.-X. Hu, L. Tian, J.-L. Lan, J.-F. Li, and Q. Zhou, "A virtual service placement approach based on improved quantum genetic algorithm," *Frontiers of Information Technology and Electronic Engineering*, vol. 17, no. 7, pp. 661–671, 2016.

[30] G. Sun, Y. Li, Y. Li, D. Liao, and V. Chang, "Low-latency orchestration for workflow-oriented service function chain in edge computing," *Future Generation Computer Systems*, vol. 85, pp. 116–128, 2018.

[31] G. Sun, D. Liao, D. Zhao, Z. Xu, and H. Yu, "Live migration for multiple correlated virtual machines in cloud-based data centers," *IEEE Transactions on Services Computing*, vol. 11, no. 2, pp. 279–291, 2018.

[32] V. Eramo, E. Miucci, M. Ammar, and F. G. Lavacca, "An approach for service function chain routing and virtual function network instance migration in network function virtualization architectures," *IEEE/ACM Transactions on Networking*, vol. 25, no. 4, pp. 2008–2025, 2017.

[33] R. Mijumbi, J. Serrat, J.-L. Gorricho et al., "Network function virtualization: state-of-the-art and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2015.

[34] W. Cerroni and F. Callegati, "Live migration of virtual network functions in cloud-based edge networks," in *Proceedings of the IEEE International Conference on Communications*, pp. 2963–2968, 2014.

[35] Y. Sang, B. Ji, G. R. Gupta, X. Du, and L. Ye, "Provably efficient algorithms for joint placement and allocation of virtual network functions," in *Proceedings of the IEEE Conference on Computer Communications*, pp. 829–837, 2017.

[36] J. Batallé, J. F. Riera, E. Escalona, and J. A. García-Espín, "On the implementation of NFV over an OpenFlow infrastructure: Routing function virtualization," *Software Defined Networks for Future Networks and Services*, pp. 1–6, 2013.

[37] W. Ma, C. Medina, and D. Pan, "Traffic-aware placement of NFV middleboxes," in *Proceedings of the IEEE Global Communications Conference*, pp. 1–6, 2015.

[38] Q. Sun, P. Lu, W. Lu, and Z. Zhu, "Forecast-assisted NFV service chain deployment based on affiliation-aware vNF placement," in *Proceedings of the IEEE Global Communications Conference*, pp. 1–6, 2017.

[39] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Transactions on Communications*, vol. 64, no. 9, pp. 3746–3758, 2016.

[40] S. Herker, X. An, W. Kiess, S. Beker, and A. Kirstaedter, "Datacenter architecture impacts on virtualized network functions service chain embedding with high availability requirements," in *Proceedings of the IEEE Global Communications Conference*, pp. 1–7, 2015.

[41] T.-W. Kuo, B.-H. Liou, K. C.-J. Lin, and M.-J. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," in *Proceedings of the IEEE Conference on Computer Communications*, pp. 1–9, 2016.

[42] H. Jin, D. Pan, J. Xu et al., "Efficient VM placement with multiple deterministic and stochastic resources in data centers," in *Proceedings of the IEEE Global Communications Conference*, pp. 2505–2510, 2012.

[43] M. T. Beck and J. F. Botero, "Coordinated allocation of service function chains," in *Proceedings of the IEEE Global Communications Conference*, pp. 1–6, 2015.

[44] Q. Zhang, X. Wang, I. Kim, P. Palacharla, and T. Ikeuchi, "Vertex-centric computation of service function chains in multi-domain networks," in *Proceedings of the IEEE Conference on Network Softwarization*, pp. 211–218, 2016.

[45] T. Wen, H. Yu, G. Sun et al., "Network Function Consolidation in Service Function Chaining Orchestration," in *Proceedings of the IEEE International Conference on Communications*, pp. 1–6, 2016.

[46] T. Park, Y. Kim, J. Park, H. Suh, B. Hong, and S. Shin, "QoSE: Quality of security a network security framework with distributed NFV," in *Proceedings of the IEEE International Conference on Communications*, pp. 1–6, 2016.

[47] Y. Li, L. T. X. Phan, and B. T. Loo, "Network functions virtualization with soft real-time guarantees," in *Proceedings of the IEEE Conference on Computer Communications*, pp. 1–9, 2016.

[48] F. Z. Yousaf, P. Loureiro, F. Zdarsky, T. Taleb, and M. Liebsch, "Cost analysis of initial deployment strategies for virtualized mobile core network functions," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 60–66, 2015.

[49] G. Sun, Y. Li, H. Yu, A. V. Vasilakos, X. Du, and M. Guizani, "Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks," *Future Generation Computer Systems*, vol. 91, pp. 347–360, 2019.

[50] A. Haque and P.-H. Ho, "A study on the design of survivable optical virtual private networks (O-VPN)," *IEEE Transactions on Reliability*, vol. 55, no. 3, pp. 516–524, 2006.

[51] X. Xie, N. Hua, and X. Zheng, "Dynamic routing, wavelength and core allocation in multi-core fiber based optical networks considering inter-core crosstalk," in *Proceedings of the Signal Processing in Photonic Communications (pp.JM3A-4)*, Optical Society of America, 2015.

[52] K. Calvert, J. Eagan, S. Merugu, A. Namjoshi, J. Stasko, and E. Zegura, "Extending and enhancing GT-ITM," in *Proceedings of the ACM Sigcomm Workshop on Models*, pp. 23–27, 2003.