Outsourcing Privacy-Preserving Social Networks to a Cloud

Guojun Wang[†], Qin Liu[*†‡], Feng Li[§], Shuhui Yang[¶], and Jie Wu[‡]
[†]School of Information Science and Engineering, Central South University, P. R. China
[‡]Department of Computer and Information Sciences, Temple University, USA
[§]Computer and Information Technology, Indiana University-Purdue University Indianapolis, USA
[¶]Department of Math, Computer Science, and Statistics, Purdue University Calumet, USA
[*]Correspondence to: gracelq628@yahoo.com.cn

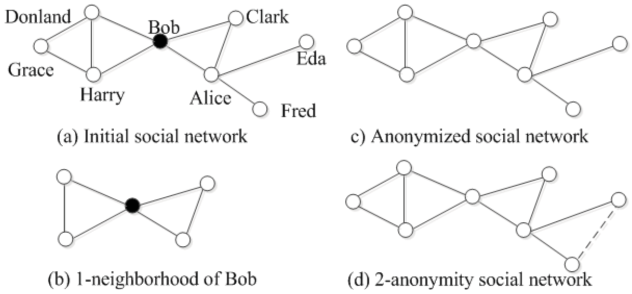
***Abstract*—In the real world, companies would publish social networks to a third party, e.g., a cloud service provider, for marketing reasons. Preserving privacy when publishing social network data becomes an important issue. In this paper, we identify a novel type of privacy attack, termed *1*-neighborhood attack*. We assume that an attacker has knowledge about the degrees of a target's one-hop neighbors, in addition to the target's *1-neighborhood graph*, which consists of the one-hop neighbors of the target and the relationships among these neighbors. With this information, an attacker may re-identify the target from a k-anonymity social network with a probability higher than $1/k$, where any node's 1-neighborhood graph is isomorphic with $k-1$ other nodes' graphs. To resist the 1*-neighborhood attack, we define a key privacy property, *probability indistinguishability*, for an outsourced social network, and propose a heuristic indistinguishable group anonymization (HIGA) scheme to generate an anonymized social network with this privacy property. The empirical study indicates that the anonymized social networks can still be used to answer aggregate queries with high accuracy.**

***Index Terms*—Cloud computing, social networks, privacy, probability indistinguishability.**

I. Introduction

As social networks have developed rapidly, recent research has begun to explore social networks to understand their structure, advertising and marketing, and data mining [1]. Cloud computing [2], [3], as an emerging computing paradigm, is expected to reshape the information technology processes in the near future. Cloud services, which are available in a pay-as-you-go manner, promise ubiquitous 24/7 access at a low cost. Due to the overwhelming merits of cloud computing, e.g., flexibility and scalability, more and more organizations that host social network data choose to outsource a portion of their data to a cloud environment [4]. Preserving privacy when publishing social network data becomes an important issue.

Social networks model social relationships with a graph structure using nodes and edges, where nodes model individual social actors in a network, and edges model relationships between social actors [5]. The relationships between social actors are often private, and directly outsourcing the social networks to a cloud may result in unacceptable disclosures. For example, publishing social network data [6] that describes a set of social actors related by sexual contacts or shared drug injections may compromise the privacy of the social actors involved. Therefore, existing research has proposed to anonymize social networks before outsourcing.



(a) Initial social network
c) Anonymized social network
(b) 1-neighborhood of Bob
(d) 2-anonymity social network

Fig. 1. 1-neighborhood attacks in a social network.

A naïve approach is to simply anonymize the identity of the social actors before outsourcing. However, an attacker that has some knowledge about a target's neighborhood, especially a one-hop neighborhood, can still re-identify the target with high confidence. This attack, termed *1-neighborhood attack*, is proposed by Zhou et al. [7].

Consider a synthetic social network of "co-authors", as shown in Fig. 1-(a), where a node denotes an author, and an edge that links two authors denotes that they previously cooperated on a paper. In the neighborhood attack, an attacker, who knows Bob's one-hop neighbors and the connections between them, i.e., Bob's *1-neighborhood graph*, as shown in Fig. 1-(b), can still re-identify Bob from an anonymized graph, Fig. 1-(c), where all user identities are removed.

This is because Bob's 1-neighborhood graph is unique. To mitigate this attack, Zhou et al. defined a k-anonymity social network, where an attacker, with the knowledge of any target's 1-neighborhood graph, cannot re-identify the target with confidence higher than $1/k$. Their basic idea is to make any node's 1-neighborhood graph isomorphic with at least $k-1$ other nodes' graphs by adding noise edges. Given k isomorphic 1-neighborhood graphs, everyone has a probability of $1/k$ to the target. For example, by adding an edge between Eda and Fred, Fig. 1-(d) becomes a 2-anonymity social network.

In this paper, we identify a novel type of privacy attack, termed *1*-neighborhood attack*, where an attacker is assumed to know the *degrees* of the target's one-hop neighbors, in addition to the structure of the 1-neighborhood graph. We call this kind of background knowledge the *1*-neighborhood* *graph*. This assumption is reasonable, since once the attacker knows the identities of the target's one-hop neighbors, he will be very likely to collect more information about the one-hop neighbors, rather than only collecting the connection information between them. With this assumption, the attacker may re-identify the target from a k-anonymity social network with a probability higher than $1/k$.

To illustrate, let us assume that the attacker knows the degrees of Bob's one-hop neighbors, Alice, Clark, Donland, and Harry, say 4, 2, 3, 3, respectively. In Fig. 1-(d), the degrees of Alice's one-hop neighbors, Bob, Clark, Eda, and Fred, are 4, 2, 2, 2, respectively. Since Ref. [7] only adds edges to make 1-neighborhood graphs isomorphic, Alice can be excluded from the target candidate set, and the probability to re-identify Bob is 1. To deal with the 1*-neighborhood attack, Ref. [7] requires the addition of more edges, so that the degrees of the k isomorphic graphs are the same. For example, by adding edges between Grace and Fred, and between Grace and Eda, the degrees of Alice's one-hop neighbors are the same as that of Bob's. However, as more edges are added, the usage of the social networks will be further compromised.

To permit useful analysis on the social networks, while preserving the privacy of the social actors involved, we define a key privacy property, *probabilistic indistinguishability*, for an outsourced social network. To generate an anonymized social network with such a property, we propose a heuristic indistinguishable group anonymization (HIGA) scheme. Our basic idea consists of four key steps: *Grouping*, we group nodes whose 1*-neighborhood graphs satisfy certain metrics together, and provide a combination and splitting mechanism to make each group size at least equal to k; *Testing*, in a group, we use random walk (RW) [8], [9] to test whether the 1-neighborhood graphs of any pair of nodes approximately match or not; *Anonymization*, we propose a heuristic anonymization algorithm to make any node's 1-neighborhood graph approximately match those of other nodes in a group, by either adding or removing edges [10], [11]; *Randomization*, we randomly modify the graph structure with a certain probability to make sure each 1*-neighborhood graph has a certain probability of being different from the original one.

Our contributions are threefold:

1) We identify a novel attack, 1*-neighborhood attack, for outsourcing social networks to a cloud.
2) We define the probabilistic indistinguishability property for an outsourced social network, and propose a heuristic indistinguishable group anonymization scheme (HIGA) to generate social networks with this privacy property.
3) We conduct experiments on both synthetic and real data sets to verify the effectiveness of the proposed scheme.

The remainder of this paper is organized as follows: We describe technique preliminaries in Section II. Then, we provide an overview of our work in Section III and describe the proposed scheme in Section IV. We analyze the privacy and performance of the proposed scheme in Section V, and conduct evaluations in Section VI. Finally, we introduce related work in Section VII, and conclude this paper in Section VIII.

II. Preliminaries

A. System Model

We consider a system that consists of a publisher, a cloud service provider, an attacker, and many users. The publisher, such as Facebook or Twitter, outsources a social network to a cloud. In our system, a social network is modeled as an undirected and unlabeled graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$, where $V(\mathcal{G})$ is a set of nodes, and $E(\mathcal{G}) \subseteq V(\mathcal{G}) \times V(\mathcal{G})$ is a set of edges. The node identities are assumed to be removed.

The attacker has certain background knowledge about the target and he tries to re-identify the target by analyzing the outsourced social network. To protect the privacy of the social actors in the network from the attacker, the publisher anonymizes $\mathcal{G}$ to $\mathcal{G}' = (V(\mathcal{G}'), E(\mathcal{G}'))$ before outsourcing. As in [7], we assume that each node in $\mathcal{G}$ exists in $\mathcal{G}'$, and no fake nodes are added in $\mathcal{G}'$ to preserve the global structure of the social network. As previous work [10], [11], we allow edges $\{(u, v)\} \in E(\mathcal{G})$ to be removed from $E(\mathcal{G}')$.

The cloud service provider, such as Google or Amazon, maintains the cloud infrastructures, which pool the bandwidth, storage space, and CPU power of many cloud servers to provide 24/7 services. We assume that the cloud infrastructures are more reliable and more powerful than personal computers. The users can perform data analysis on the outsourced social networks with more convenience.

The users are assumed to be more interested in aggregate queries on the social networks, e.g., the average number of co-authors. While many types of queries on social networks are important, we are particularly interested in aggregate queries in this paper. We believe that the usability of a social network can be examined in a meaningful way using aggregate queries.

B. Attack Model

In this paper, we assume that the attacker is more interested in the privacy of social actors. Before launching an attack, the attacker needs to collect some background knowledge about the target victim. We assume that an attacker may have background knowledge about the *1*-neighborhood graphs* of some targets. Informally, a target's 1*-neighborhood graph consists of both the 1-neighborhood graph of the target and the degrees of the target's one-hop neighbors. Following the work in [7], for each node $u \in V(\mathcal{G})$, the related 1-neighborhood graph, denoted as G_u, is defined as follows:

1-Neighborhood Graph. *$G_u = (V_u, E_u)$, where V_u denotes a set of nodes $\{v|(u, v) \in E(\mathcal{G}) \vee (v = u)\}$, and E_u denotes a set of edges $\{(w, v)|(w, v) \in E(\mathcal{G}) \wedge \{w, v\} \in V_u\}$.*

For each node $u \in V(\mathcal{G})$, the related 1*-neighborhood graph, denoted as G_u^*, is defined as follows:

1*-Neighborhood Graph. *$G_u^* = (G_u, D_u)$, where G_u is the 1-neighborhood graph of node u, and D_u is a sequence of degrees of u's one-hop neighbors.*

We focus on 1*-neighborhood attack since it tends to be much more difficult for an attacker to collect information

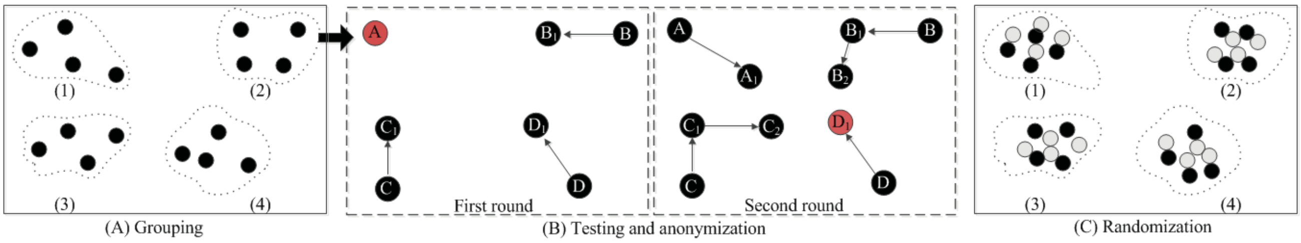


Fig. 2. Analogue of the HIGA scheme.

beyond a one-hop neighborhood [7], and the detailed neighborhood information about a target's directed neighbors is more difficult to collect comparing to their degree information. For example, it may be easy to know that Alice has 100 neighbors, but hard to know the detailed information (ID, name, or age) about these 100 neighbors. For a social actor u, if an attacker can re-identify u from an outsourced social network with a high probability, then the privacy of the social actor is leaked.

C. Random Walk (RW) based Approximate Matching

Inspired by the work in [12], we use random-walk-based approximate matching as the building block of our HIGA scheme. The random walk (RW) [8] is known as a useful tool to obtain the steady state distribution for a graph, referred to as the *topological signatures*, which provide the foundation for the approximate matching. Specifically, given graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$, where $V(\mathcal{G}) = \{u_1, \ldots, u_{|V(\mathcal{G})|}\}$. A RW on $\mathcal{G}$ allows the probability $p_{u_j}(t)$ of a node $u_j \in V(\mathcal{G})$ being located at time t to be computed with Eq. 1:

$$\begin{aligned} p_{u_j}(t) = & \sum_{u_i \in V(\mathcal{G})} \frac{1}{|V(\mathcal{G})|} \cdot (1-d) \cdot p_{u_i}(t-1) \\ & + \sum_{u_i \in N(u_j)} \frac{1}{|N(u_i)|} \cdot d \cdot p_{u_i}(t-1) \end{aligned} \tag{1}$$

where $|V(\mathcal{G})|$ is the number of nodes in $\mathcal{G}$, $|N(u_j)|$ is the number of one-hop neighbors for node u_j, and d is the damping factor which defines the probability of directly jumping or traversing. For Eq. 1, the first part relates to the probability of moving from node u_i to u_j by jumping directly, while node u_i is not a one-hop neighbor of u_j; the second part relates to the probability by traversing the edge from u_i to u_j, while node u_i is a one-hop neighbor of u_j.

Therefore, the probability distribution on all nodes in $\mathcal{G}$, denoted as a vector $\mathbf{p}(t) = [p_{u_1}(t), \ldots, p_{u_{|V(\mathcal{G})|}}(t)]$, can be calculated with Eq. 2:

$$\mathbf{p}(t) = \frac{(1-d)}{N} \cdot \mathbb{I} + d \cdot \mathbf{W} \cdot \mathbf{p}(t-1) \tag{2}$$

where $\mathbf{W} = (\theta \mathbf{A})'$, $\mathbf{A}$ being the adjacency matrix of the graph and θ the diagonal matrix whose diagonal element is $\frac{1}{|N(u_i)|}$; $\mathbb{I}$ is a vector whose entries are all equal to one.

The steady state distribution of the RW, defined by Eq. 2, can be expressed as:

$$\mathbf{p}^\star = \frac{(1-d)}{|V(\mathcal{G})|} \cdot \sum_{k=0}^{\infty} d^k W^k \cdot \mathbb{I} \tag{3}$$

where $\sum_{k=0}^{\infty} d^k W^k$ is used to denote $(1 - d\mathbf{W})^{-1}$. In the our anonymization scheme, the testing step is based on the approximate matching using the above topological signatures.

D. Design Goals

The main design goal of our work is to reduce the probability of a social actor being re-identified while publishing social networks to a cloud. Specifically, given a social graph $\mathcal{G}$, we want to generate an anonymized graph $\mathcal{G}'$, so that the following requirements are satisfied:

- **Privacy**. Given any target's 1-*neighborhood graph, the attacker cannot re-identify the target from an anonymized social network with confidence higher than a threshold.
- **Usability.** The anonymized social networks can be used to answer aggregate queries with high accuracy.

III. Scheme Overview

A. Definitions

To preserve privacy, previous research proposed to make any node's 1-neighborhood graph be isomorphic with at least $k-1$ others. In many cases, isomorphism is a strong condition that is not necessary for anonymizing the graph. In this paper, we define the concept of *probabilistic indistinguishability*, which can preserve privacy at a lower anonymization cost.

Let G_u^* and $G_u'^*$ denote the 1*-neighborhood graph of node u in the original social network $\mathcal{G}$ and in the anonymized social network $\mathcal{G}'$, respectively. Probabilistic indistinguishability can be defined in a hierarchical way as follows:

Node Indistinguishability. *Nodes u and v are indistinguishable if an observer cannot decide whether or not $G_u^* \neq G_v^*$ in the original graph $\mathcal{G}$, by comparing $G_u'^*$ and $G_v'^*$ in an anonymized graph $\mathcal{G}'$.*

Here, "cannot decide" means that the observers' confidence level will be below a pre-determined threshold. In our scheme, we achieve node indistinguishability by introducing the randomness into the published graph. If there is no limit on the randomness, node indistinguishability is easy to achieved. However, we hope to preserve the usability of the published graph, and hence, need to minimize the *anonymization cost*. Thus, we need to have a more sophisticated design.

Group Indistinguishability. *For a group of nodes $g = \{v | v \in V(\mathcal{G})\}$ and $|g| \geq k$ if for each pair of nodes $\{\langle u, v\rangle | u, v \in g\}$, u and v are indistinguishable in the published graph $\mathcal{G}'$, group g is an indistinguishable group.*

Probabilistic Indistinguishability. *A published social network $\mathcal{G}'$ achieves probabilistic indistinguishability, if all nodes*

$\{v|v \in V(\mathcal{G}')\}$ *can be classified into* $m \geq 1$ *groups, where each group has the property of* group indistinguishability.

Problem Definition. *Given a network graph* $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ *and a positive integer* k*, derive an anonymized graph* $\mathcal{G}' = (V(\mathcal{G}'), E(\mathcal{G}'))$ *to be published, such that (1)* $V(\mathcal{G}') = V(\mathcal{G})$*; (2)* $\mathcal{G}'$ *is probabilistic indistinguishable with respect to* $\mathcal{G}$*; (3) the anonymization from* $\mathcal{G}$ *to* $\mathcal{G}'$ *has minimal anonymization cost.*

The problem of generating a social network with above three properties is NP-hard. The proof lies in reducing the NP-complete problem of Partition into Triangles [13]. As limited by space, we omit the details here.

B. Intuition

The basic idea of the heuristic indistinguishable group anonymization (HIGA) scheme consists of four steps:

- **Step 1: Grouping.** We classify nodes whose 1*-neighborhood graphs satisfy certain metrics into groups, where each group size is at least equal to k.
- **Step 2: Testing.** We use random walk (RW) to test whether the 1-neighborhood graphs of nodes in a group approximately match or not.
- **Step 3: Anonymization.** We use a heuristic anonymization algorithm to make the 1-neighborhood graphs of nodes in each group approximately match.
- **Step 4: Randomization.** We randomly modify the graph with certain probability to make each node's 1*-neighborhood graph be changed with certain probability.

We use the example shown in Fig. 2 as the analogue of the HIGA scheme: We first classify the nodes that are the *closest* into groups with size at least equal to k, so that the moving distance in the third step can be reduced, e.g., in Fig. 2-(A), nodes are classified into four groups. Then, we test whether the *distance* between each pair of nodes in a group is shorter than a threshold value α. If not, in each group, we choose a random node as the seed and move other nodes toward the seed, until it passes testing. For example, in Fig. 2-(B), we first choose node A as the seed, and then choose node D as the seed for group 2, where B_1, C_1, and D_1 are the positions of nodes B, C, D after the first round, and A_1, B_2, and C_2 are the positions of nodes A, B, C after the second round. In the last step, we randomly move each node for a short distance. In Fig. 2-(C), the black nodes and the pale nodes denote node positions before and after randomization, respectively.

Probabilistic indistinguishability is achieved as follows: Steps 1 through 3 enable all nodes in the network to be classified into multiple groups, where any pair of nodes are very close. After Step 4, each pair of nodes in a group are still close, but they will deviate from the original position with a high probability. In a group, from the current positions of any pair of nodes, the attacker cannot decide whether or not they are in the same position as they were in the original graph, i.e., node indistinguishability is achieved. Since each group size is at least equal to k, group indistinguishability is achieved.

TABLE I
EFFECTIVENESS OF METRICS FOR GROUPING

Nodes	Edges	Percentage
$100 \sim 200$	$1,000 \sim 2,000$	47%
$500 \sim 1,000$	$5,000 \sim 10,000$	61%

IV. HEURISTIC INDISTINGUISHABLE GROUP ANONYMIZATION SCHEME

A. Grouping

We group nodes by using the following metric: *number of one-hop neighbors*, *in-degree sequence*, *out-degree sequence*, *total number of edges*, and *betweenness*. Although other metrics, e.g., closeness centrality and local clustering coefficient, also can be used for grouping, we only consider the above metrics. The concepts of "number of one-hop neighbors" and "total number of edges" are easily understood. Therefore, we only provide the definitions for the other metrics. For a node $v \in V(\mathcal{G})$, whose 1*-neighborhood graph $G_v^* = (G_v, D_v)$, where $G_v = (V_v, E_v)$, we have the following definitions:

In-degree sequence. $I_v = \{|E_u^+|\}_{u \in V_v}$*, where* $E_u^+ = \{(u,w)|w \in V_v\}$*, and* $|E_u^+|$ *is the number of edges in* E_u^+*.*

Out-degree sequence. $O_v = \{|E_u^-|\}_{u \in V_v}$*, where* $E_u^- = \{(u,w)|w \notin V_v\}$*, and* $|E_u^-|$ *is the number of edges in* E_u^-*.*

Betweenness. $B_v = |V_v^*|/|V_v^+|$*, where* $V^* = \{\langle u,w\rangle|u,v \in V_v \wedge (u,w) \notin E_v\}$*, and* $V_v^+ = \{\langle u,w\rangle|u,v \in V_v\}$*.*

For a node v, the in-degree sequence is a sequence of in-degrees for v's one-hop neighbors, where the in-degree for v's one-hop neighborhood u is the number of edges that are connected between u and other v's one-hop neighbors; the out-degree sequence is a sequence of out-degrees for v's one-hop neighbors, where the out-degree for v's one-hop neighborhood u is the number of edges that are connected between u and the nodes outside v's 1-neighborhood graph; the betweenness is the ratio of the number of paired nodes whose shortest path must go through node v to the total number of node pairs in v's 1-neighborhood graph.

To test the effectiveness of the metrics for grouping, we first randomly generate a set of graphs where the node degree follows the power-law distribution, and we use the above metrics to classify nodes into groups. Then, we calculate the percentage of graphs that are isomorphic in a group.

In Table I, in social networks with $100 \sim 200$ nodes and $1,000 \sim 2,000$ edges, the percentage of isomorphic 1*-neighborhood graphs is about 47%. As the scale of the social networks increases to $500 \sim 1,000$ nodes and $5,000 \sim 10,000$ edges, the percentage increases to around 61%. Therefore, these metrics are helpful for grouping.

B. Testing

In the testing step, we analyze each pair of nodes u and v by computing the steady states of their 1-neighborhood graphs $G_u = (V(G_u), E(G_u))$ and $G_v = (V(G_v), E(G_v))$ with the

RW tool and Eq. 3. We determine the approximate matching of G_u and G_v by performing bipartite graph matching as follows:

If $|V(G_u)| < |V(G_v)|$, we should add virtual nodes in $V(G_u)$ to make $|V(G_u)| = |V(G_v)|$, and vice versa. Let $cost(x, w)$ be the cost of matching nodes $x \in V(G_u)$ and $w \in V(G_v)$, $\mathbb{V}$ be the set of virtual nodes, and $\mathbf{p}_x^\star$ and $\mathbf{p}_w^\star$ denote the steady states of nodes x and w, respectively. If $x, w \notin \mathbb{V}$, the cost function calculates the *Euclidean distance* between the topological signatures of the nodes as follows:

$$cost(x, w) = \sqrt{(\mathbf{p}_x^\star - \mathbf{p}_w^\star)^2} \tag{4}$$

If either x or w is a virtual node, $cost(x, w)$ is set to a fixed value β. The cost of matching $V(G_u)$ and $V(G_v)$, which is the sum of the costs of matching all nodes in $V(G_u)$ and $V(G_v)$, can be calculated with Eq. 5:

$$cost(G_u, G_v) = \sqrt{\sum_{x,w \notin \mathbb{V}} (\mathbf{p}_x^\star - \mathbf{p}_w^\star)^2 + (|\mathbb{V}| * \beta)} \tag{5}$$

We define graph approximate matching as follows:

Approximate matching. *Let $G_u = (V(G_u), E(G_u))$ and $G_v = (V(G_v), E(G_v))$ be two graphs. G_u and G_v approximately match, denoted as $G_u \approx G_v$, if an optimal bipartite graph matching exists between $V(G_u)$ and $V(G_v)$, such that the $cost(G_u, G_v)$ is smaller than a threshold value α.*

Therefore, given a pair of nodes u and v, if their 1-neighborhood graphs approximately match, we say that node u and node v approximately match, denoted as $u \approx v$. The key problem lies in determining the threshold value α. If this value is too large, then the two dissimilar graphs will be deemed approximately matching. If this value is too small, then approximate matching may be equal to isomorphic. Since α relates to the number of nodes in G_u and G_v, we find a reasonable threshold value by conducting experiments in Section VI-(A). To enrich the topological signatures, we can compute the steady states with different damping factors. Suppose that we choose n distinct damping factors $D = [d_1, \ldots, d_n]$, the topological signatures associated with node w can be defined as $\mathbf{p}_w^\star = [\mathbf{p}_w^\star(d_1), \ldots, \mathbf{p}_w^\star(d_n)]$.

C. Anonymization

Suppose that there are m groups, $g_1, \ldots, g_m$, where each group size is assumed to be at least equal to k. For each group, if any pair of nodes are not approximately matching, we use a heuristic anonymization algorithm (Alg. 1) to make the 1-neighborhood graphs approximately match as follows:

Initially, the candidate group set (CGS) consists of m groups. We sort groups in descending order of the number of neighbors, pick the first one as the *processing group*, and remove it from CGS. For each node in the processing group, we construct its 1-neighborhood graph, and use RW to calculate related topological signatures. Then, for any pair of nodes u and v, we use Eq. 5 to calculate the cost of matching their 1-neighborhood graphs. For any pair of nodes, if this cost is smaller than a threshold value α, we choose the next grouping in CGS as the *processing group* and do it again.

Algorithm 1 Heuristic Anonymization Algorithm

```
{Given $m$ groups $g_1, \ldots, g_m$ as CGS}
Sort CGS in descending order of the number of neighbors
while CGS is not empty do
    Choose the first group in CGS as the processing group
    $g_*$ and remove $g_*$ from CGS
    for each node $u$ in $g_*$ do
        Construct 1-neighborhood graph $G_u$
        Use Eq. 3 to calculate $G_u$'s topological signatures
    for each pair of nodes $(u, v)$ in $g_*$ do
        Use Eq. 5 to calculate cost of matching $G_u$ and $G_v$
    while exists a cost larger than $\alpha$ do
        Randomly choose a node $u \in g_*$ as the group seed
        for each node $v \in g_*$ do
            if $cost(G_u, G_v) > \alpha$ then
                Approach $G_u$ to $G_v$ with probability $q$
                Approach $G_v$ to $G_u$ with probability $1 - q$
```

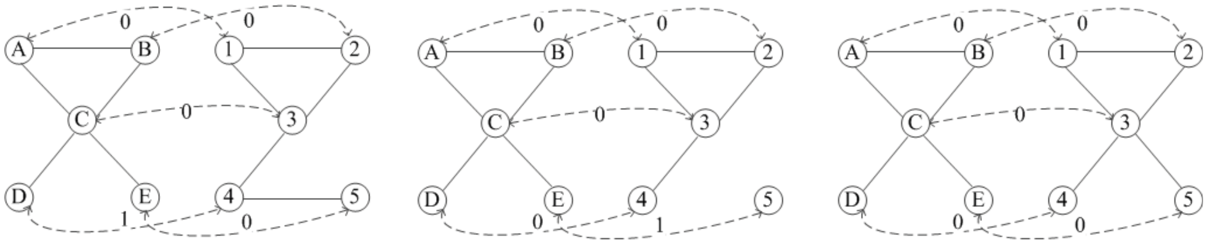
Otherwise, we modify 1-neighborhood graphs of the nodes in the processing group as follows: We first choose a random node u in the group as the *group seed*. For any other node v in this group, if the related cost $cost(G_u, G_v)$ is larger than α, we approach the structure of G_u to that of G_v with probability q, and approach the structure of G_v to that G_u with probability $1-q$. This process will continue until, for any pair of nodes in the processing group, the cost for matching their 1-neighborhood graphs is equal to or smaller than α. The anonymization process may be recursive, since some changes may impact the groups that have been processed previously. However, due to the power-law node distribution, and the small world phenomenon [14], this process will rapidly stop.

To approach the structure of $G_v = (V_v, E_v)$ to that of $G_u = (V_u, E_u)$, we first obtain the optimal matching of nodes in two graphs with the method discussed in Section IV-(B). In the optimal matching, for any pair of nodes $x \in V_v$ and $w \in V_u$, if $cost(x, w) > \alpha$, we make u's connections the same as those of v. During the approaching process, we make sure that the structure of G_u will not be modified.

For example, in Fig. 3-(B), the optimal matching for graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ is A1, B2, C3, D4, E5. Suppose $\alpha = 0$, we approach $\mathcal{G}_1$ to $\mathcal{G}_2$. For all pairs of nodes, only the cost of matching $D \in \mathcal{G}_1$ and $4 \in \mathcal{G}_2$ is larger than 0. Therefore, we change the connection of node 4 to the same as that of node D. In the first round, node D only connects to node C in $\mathcal{G}_1$, but node 4 connects to both nodes 3 and 5. Therefore, we remove the edge between nodes 4 and 5, and test the cost again. In this round, $cost(E, 5) > \alpha$, and node E connects to C in $\mathcal{G}_1$, but node 5 has no connections. Therefore, we add an edge between nodes 3 and 5, and test the cost again. In this round, the related cost of matching $\mathcal{G}_1$ and $\mathcal{G}_2$ is equal to α, and the anonymization completes.

D. Randomization

Consider a graph $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ and a randomization probability p. We first randomly remove $p(|E(\mathcal{G})|)$ edges from $\mathcal{G}$, and then for two nodes that are not linked, we add an



(a) Original graphs (b) First round (c) Second round

Fig. 3. Working process of the heuristic approach algorithm. The righthand graph is $\mathcal{G}_1$ and the lefthand graph is $\mathcal{G}_2$.

edge with probability p. This randomization process closely follows that in [15]. The key problem lies in determining p to randomize the graph. To obtain a reasonable parameter, we conduct experiments in Section VI-(A).

E. Combining and Splitting Groups

In the previous section, we simply assume that the size of each group is equal to k. However, during the empirical study, we found that the group size also follows the power-law distribution: most groups contain only one or two members, and a small number of the groups have thousands of members. Therefore, we provide a combining and splitting mechanism to make each group have an appropriate size.

We first sort groups in descending order of the number of neighbors associated with its members. The sorted groups are denoted as $g'_1, g'_2, \ldots, g'_m$. For $i > m$, if g'_i's size is smaller than k, we combine the members in g'_{i+1} into g'_i as follows: if $|g'_i| + |g'_{i+1}| < k$, we simply combine two groups; otherwise, we should group $k - |g'_i|$ nodes in g'_{i+1} to $|g'_i|$. If the size of the last group is equal to or larger than k, then let it be. Otherwise, we combine it to the previous group. Therefore, each group, except for the last one, has exactly k members, and the last group may have $[k, 2k)$ members.

V. Analysis

A. Anonymization Strength

Theorem 1. *From the anonymized graph $\mathcal{G}'$, an attacker, with the knowledge of any target's 1*-neighborhood graph, cannot re-identify the target with confidence higher than $1/k$.*

Proof: The attacker will try to re-identify a target from the published graph $\mathcal{G}'$ by using the target t's 1*-neighborhood graph G_t^* in the original graph (attacker's knowledge). There are two possible consequences after searching $\mathcal{G}'$:

- **Case 1.** Attacker found at least an exact match of target.
- **Case 2.** Attacker cannot find an exact match of target.

Here, we assume an intelligent attacker who knows the uniform random noise probability p. We also assume that the intelligent attacker will not give up even if the exact match cannot be found.

For Case 1, after the exact matching, the attacker has two possible strategies: 1. Consider an exact match u as the re-identified target; 2. Consider other nodes as the re-identified target t. The latter strategy will be combined in the discussion of Case 2. Let us first consider Case 1 with the first strategy. Based on the uniform random noise, the probability that the target t's 1*-neighborhood graph G_t was not changed is $P(G_u^* = G_t^* \wedge G_u'^* = G_t'^*) = (1-p)^{|E(G_t^*)|}$.

Besides this factor, we also know that the exact match of G_t^* must belong to an indistinguishable group g with $|g| \geq k$. For each node v in g, we know that an observer cannot decide whether or not $G_v^* \neq G_t^*$ in the original graph $\mathcal{G}$ by comparing $G_v'^*$ and $G_t'^*$ in $\mathcal{G}'$. Therefore, we have $P(G_v^* \neq G_t^* \wedge G_v'^* = G_t'^*) \geq (1-p)^{|E(G_t^*)|}$ (otherwise, v and t will violate the indistinguishability requirement in g).

Let $\tau = (1-p)^{|E(G_t^*)|}$. Therefore, under Case 1 with the first strategy, the probability that an exact match node u is the correctly identified target t is:

$$P(u=t) = \frac{\tau}{\tau + \sum\limits_{v \in g, v \neq u} P(G_v^* \neq G_t^* \wedge G_v'^* = G_t'^*)} \qquad (6)$$

Since $P(G_v^* \neq G_t^* \wedge G_v'^* = G_t'^*) \geq \tau$ and $|g| \geq k$, it is clear that $P(u=t) \leq \tau/(k \cdot \tau) = 1/k$.

Case 2 and the remaining part of Case 1 can be proven in a similar manner. ■

Corollary 1. *The anonymization strength of the heuristic indistinguishable group anonymization scheme $\geq$ k-anonymity social network defined in [7].*

Proof: The proof of Corollary 1 is obvious. First, the k-anonymity social network defined in [7] assumes that the attacker only knows a target's 1-neighborhood, which contradicts the reality since the attacker usually collects more information about the one-hop neighbors, rather than only collecting the connection information between them. With the 1*-neighborhood knowledge, the attacker can further narrow down the target in the blend-in group and re-identify it.

Second, even if we assume that the k-anonymity social network can be extended to the 1*-neighborhood case (it will significantly increase the anonymization cost due to the exact matching), the k-anonymity social network only guarantees that the attacker cannot identify the target with confidence higher than $1/k$. According to Theorem 1, our scheme will produce equal or greater anonymization strength. ■

B. Anonymization Cost

Our solution anonymizes a graph by adding and removing edges, which will lead to some information loss. Consider a social network graph $\mathcal{G}$. We first classify the nodes into m groups $g_1, \ldots, g_m$. For each group g_i, we transfer a set of original 1*-neighborhood graphs $\{G_u^* | u \in g_i\}$ to a set of anonymized 1*-neighborhood graphs $\{G_u'^* | u \in g_i\}$.

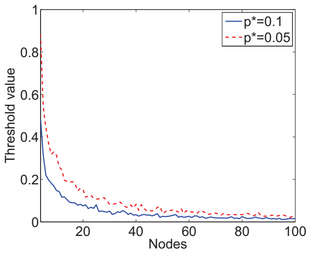


Fig. 4. Threshold Values α.

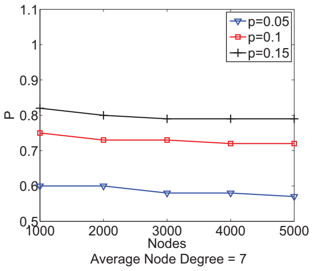


Fig. 5. Probability p.

Let m_i and n_i denote the numbers of added edges and removed edges. The related anonymization cost $C_{\mathcal{G}(g_i)\to\mathcal{G}'(g_i)}$ for transferring g_i to g'_i can be calculated with Eq. 7:

$$C_{\mathcal{G}(g_i)\to\mathcal{G}'(g_i)} = am_i/L_i + bn_i/L_i \quad (7)$$

where a and b are the weights associated with each component, and L_i is the number of edges in $\{G_u|u \in g_i\}$. The cost consists of two parts: the first part is the normalized cost measuring the information loss of adding edges, and the second part is the normalized cost measuring the information loss of removing edges. Therefore, given m groups of graphs, the total cost is:

$$C_{\mathcal{G}\to\mathcal{G}'} = \sum_{i=1}^{m} C_{\mathcal{G}(g_i)\to\mathcal{G}'(g_i)} \quad (8)$$

VI. Evaluation

In this section, we evaluate our anonymization method on both synthetic and real data sets. Experiments are conducted with MATLAB R2010a running on the local machine with an Intel Core 2 Duo E8400 3.0 GHz CPU and 8 GB Linux RAM.

A. Parameter Setting

The following parameters need to be determined before conducting experiments:

- The threshold value α that determines whether or not two 1-neighbor graphs approximately match.
- The probability p for randomizing the graph.

To obtain a reasonable threshold value, we conduct experiments with respect to different sizes of 1-neighbor graphs as follows: Given N, we first randomly generate a 1-neighborhood graph G_v with N nodes, and then generate a similar graph G'_v by randomly modifying p^* percentage of edges. Finally, with n different damping factors $D = [d_1, \dots, d_n]$, we calculate the cost for optimal matching G_v and G'_v, denoted as $cost(G_v, G'_v)$. The above process will be conducted for multiple rounds, and the average value $\widetilde{c}$ is used as the threshold value α.

In our experiments, we set $p^* = [0.1, 0.05]$ and $\beta = 0.1/N$ for matching nodes in the virtual node set, and choose damping factors $D = [0.7, 0.8, 0.9]$. Fig. 4 shows the threshold value α with different p^* values, while N ranges from 4 to 100. We know that α will decrease as N increases, e.g., when $N = 4$, $\alpha = 0.22$, and when $N = 100$, $\alpha = 0.0158$.

To obtain a reasonable randomization probability p, we first randomly generate a graph with N nodes and M edges. Then, we randomize the graph with different p values, and calculate the percentage P of 1*-neighborhood graphs being changed in the randomized graph. This process will be done multiple times, and the average percentage P will be used to measure the impact of p. Fig. 5 shows the value of P with respect to different p under a graph with average node degree 7. We know that $p = 0.05$ is appropriate for a network with such settings. For a network with average node degree 3, the changing percentage P is about 47% when $p = 0.05$.

B. Synthetic Data Set

We use the Barabái-Albert algorithm [16] (B-A algorithm for short) to generate synthetic data sets. The basic idea of the B-A algorithm is to first generate a network of a small size, and then use that network as a *seed* to build a larger-sized network, continuing this process until the actual desired network size is reached. The node degree follows the power-law distribution. In our experiments, the initial seed size contains 5 interconnected nodes, and the generated networks contain 1,000, 2,000, 3,000, 4,000, and 5,000 nodes.

Figs. 6 and 8 show the percentage of modified edges with respect to different k values and different graph settings. We know that as k increases or p^* decreases, the number of modified edges increases, e.g., when $k = 5$ and $p^* = 0.1$, for a network that has 5,000 nodes with average node degree 5, the HIGA scheme needs to change 350 edges to pass testing, but as k increases to 20, the number of modified edges increases to 559; when $k = 10$ and $p^* = 0.1$, for a network that has 5,000 nodes with average node degree 5, the HIGA scheme needs to change 420 edges to pass testing, but as p^* decreases to 0.05, the number of modified edges increases to 680.

Figs. 7 and 9 show the related anonymization costs calculated with Eq. 8, where parameters a and b are set to 0.6 and 0.4. We consider that removing edges will lose most of the information, and treat this setting default in the remainder of experiments. We know that as the number of nodes in a network increases, the anonymization cost decreases; as k increases, the anonymization cost increases.

C. Real Data Set

To validate the effectiveness of our anonymization method, we conduct experiments on a real social network, Arxiv ASTRO-PH (Astro Physics) collaboration network[1], which is from the e-print arXiv and covers scientific collaborations between authors, submitted papers to Astro Physics category, in the period from January 1993 to April 2003, and contains 18,772 nodes and 396,160 edges. In Astro Physics collaboration network, if an author i co-authored a paper with author j, the graph contains an undirected edge from i to j.

First, we classify nodes, whose 1*-neighborhood graphs satisfy certain metrics, into different groups. After grouping, nodes are classified into 7,029 groups, where the group size ranges from 1 to 1,632, and the average group size is 3. Therefore, we execute a splitting and combining process to make each group size at least equal to k.

[1] http://snap.stanford.edu/data/ca-AstroPh.html

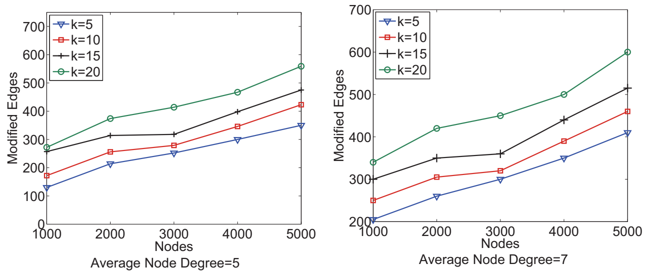


Fig. 6. Number of modified edges on synthetic data sets. $p^* = 0.1$.

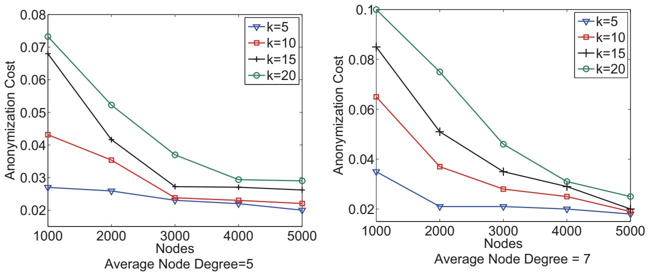


Fig. 7. Anonymization costs on synthetic data sets. $p^* = 0.1$.

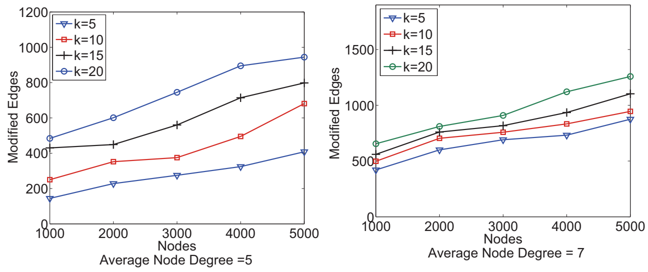


Fig. 8. Number of modified edges on synthetic data sets. $p^* = 0.05$.

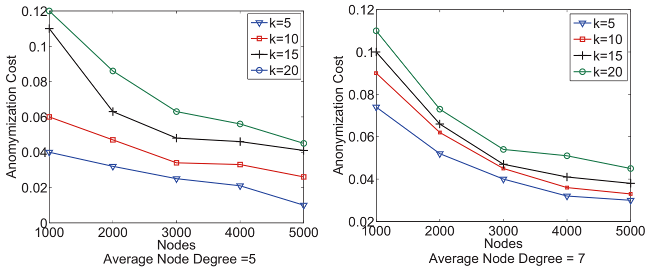


Fig. 9. Anonymization costs on synthetic data sets. $p^* = 0.05$.

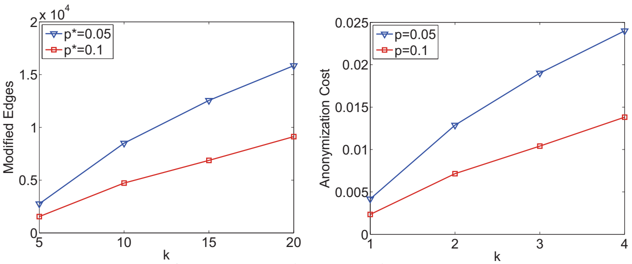


Fig. 10. Results on Real Data Set.

TABLE II
USABILITY OF THE ANONYMIZED SOCIAL NETWORK

	Max	MIN	AVE	Error Rate
Original	505	2	22.1	0
k=5	505	2	22.4	2.9%
k=10	496	2	22.6	6.4%
k=15	485	2	22.9	8.1%
k=20	476	2	23.3	8.3%

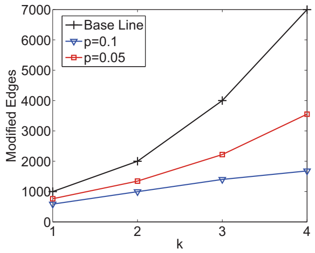


Fig. 11. Comparison with the results in [7].

To obtain reasonable threshold value α and randomization probability p, we conduct experiments as in Section VI-(A). For α, we choose $D = [0.7, 0.8, 0.9]$ and $p^* = [0.05, 0.1]$. For p, we find that 97% of 1*-neighborhood graphs are changed when we set $p = 0.05$, which increases to 99% as we set $p = 0.15$. Therefore, $p = 0.05$ is enough for randomization.

Fig. 8 shows the percentage of modified edges and the anonymization costs with respect to different k values and p^* values. We know that as k increases or as p^* decreases, the number of modified edges increases. However, even when $k = 20$ and $p^* = 0.05$, the number of modified edges is only 15,802, which is less than 4% of the total number of edges.

To test the usability of the anonymized social network, we conduct aggregate queries on both the original social network and the anonymized one. The anonymized graph is generated with parameter $p^* = 0.05$ and $p = 0.01$. We first test the differences between two social networks on the maximal node degree (MAX), the minimal node degree (MIN), and the average node degree (AVE). From Table II, we know that even when $k = 20$, the results are still useful.

Then, we conduct experiments to test whether the anonymized graph can be used to answer shortest distance queries. We randomly choose a pair of nodes u and v, and test the shortest distances between them in the original graph $\mathcal{G}$ and the anonymized graph $\mathcal{G}'$, denoted as $dist_1$ and $dist_2$, respectively. If $dist_1 \neq dist_2$, then we consider this is a query error. The above process will be repeated multiple times, and the ratio of the number of query errors to the total number of queries will be used as the error rate. From Table II, we know that the error rate is relatively small, even when $k = 20$.

To compare our results with those of [7], we also conduct experiments on KDD cup 2003 co-authorship data set. To fairly perform comparisons, we extract 120,640 edges from the data set, so that our experiment setting is the same as theirs. For the ease of comparison, the results of [7] are denoted as the *base line*. In Fig. 10, our scheme outperforms [7] in terms of the number of modified edges while keeping the same level of privacy. Therefore, our scheme allows less information loss, and hence, high usability. Our current implementation does require more computation time for edge modification (about one magnitude higher). However, the process of edge modification usually is done off-line. As part of our future extensions, we will improve the efficiency of the implementation.

VII. Related Work

Our work is on outsourcing privacy-preserving social networks to a cloud environment. Research in this area is still in its infancy. To the best of our knowledge, the work by [4] is the first to address this problem. The work that is closest to our work can be found in publishing privacy-preserving social networks [7], [13], [15], [17]–[22].

As a pioneering work, Backstrom et al [17] discussed two re-identification attacks in naïve anonymized social networks. In active attacks, an attacker intentionally embeds a subgraph into a social network before publishing and uses such kinds of background knowledge to re-identify nodes and edges in the published network. In passive attacks, an attacker with the knowledge of a target's subgraph can infer the identity of nodes in the published network. However, they do not provide a solution to counter these attacks.

To defend the re-identification attacks, the work in [18] advocated the k-anonymity model, where every node should be indistinguishable with at least k other nodes in terms of both the attributes and the associated structural information, such as neighborhood and node degree. To preserve the scale and the local structures of the original graph, existing anonymization approaches [7], [13], [15], [19]–[22] try to locally modify the graph structure to achieve the privacy preservation requirement. For example, the work in [13] proposed the guarantee of k-anonymity on node degrees, so that for every node v, there are at least k other nodes that have the same node degrees as v. The work in [7] provided a heuristic solution against the 1-neighborhood attack. The work in [15] quantified the privacy risks associated with different kinds of attacks on social networks. The work in [19] anonymized the data graph by adding edges and nodes so that the resulting graph is k-automorphic. The work in [20] identified two realistic targets of attacks, NodeInfo and LinkInfo, and proposed a solution to form k pairwise isomorphic subgraphs.

Most of the above work aims at node re-identification, e.g., in the published data the attacker cannot re-identify any individual to a node with a high confidence. Most of the solutions target k-anonymity. In our work, we identify a more realistic attack model, and our solution targets probabilistic indistinguishability.

VIII. Conclusion

In this paper, we identify a novel 1*-neighborhood attack. To resist this attack, we define a key property, probabilistic indistinguishability for outsourced social networks, and we propose a heuristic anonymization scheme to anonymize social networks with this property. The empirical study indicates that the anonymized social networks can still be used to answer aggregate queries with high accuracy. For our future work, we will conduct a thorough theoretical study of risks on the outsourcing social networks to a cloud, and try to introduce other privacy mechanisms to our scheme, e.g., by combining with l-diversity, we enable the nodes in a group to be associated with at least l different attributes. Furthermore, the average node degree is 22 in the evaluation. However, in many social networks, the average node degree is much higher which may make the proposed anonymization scheme inefficient. Therefore, we will conduct more experiments on larger social graphs with higher node density.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant numbers 61272151 and 61073037, the Ministry of Education Fund for Doctoral Disciplines in Higher Education under grant number 20110162110043; and by NSF grants ECCS 1231461, ECCS 1128209, CNS 1065444, CCF 1028167, and CNS 0948184.

References

[1] L. Getoor and C. Diehl, "Link mining: A survey," *ACM SIGKDD Explorations Newsletter*, 2005.
[2] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, 2010.
[3] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in *Proceedings of ACM CCS*, 2010.
[4] J. Gao, J. Yu, R. Jin, J. Zhou, T. Wang, and D. Yang, "Neighborhood-privacy protected shortest distance computing in cloud," in *Proc. of ACM COMAD*, 2011.
[5] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM SIGKDD Explorations Newsletter*, 2008.
[6] J. Potterat, L. Phillips-Plummer, S. Muth, R. Rothenberg, D. Woodhouse, T. Maldonado-Long, H. Zimmerman, and J. Muth, "Risk network structure in the early epidemic phase of HIV transmission in Colorado springs," *Sexually Transmitted Infections*, 2002.
[7] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *Proc. of IEEE ICDE*, 2008.
[8] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Tech. Rep., 1999.
[9] M. Diligenti, M. Gori, and M. Maggini, "A unified probabilistic framework for web page scoring systems," *IEEE Transactions on Knowledge and Data Engineering*, 2004.
[10] E. Zheleva and L. Getoor, "Preserving the privacy of sensitive relationships in graph data," in *Proc. of ACM PinKDD*, 2007.
[11] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proc. of IEEE ICDM*, 2009.
[12] M. Gori, M. Maggini, and L. Sarti, "Exact and approximate graph matching using random walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.
[13] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proc. of ACM SIGMOD*, 2008.
[14] J. Scott, "Social network analysis," *Sociology*, 1988.
[15] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava, "Anonymizing social networks," Tech. Rep., 2007.
[16] A. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, 1999.
[17] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *Proc. of ACM WWW*, 2007.
[18] A. Campan and T. Truta, "A clustering approach for data and structural anonymity in social networks," in *Proc. of PinKDD*, 2008.
[19] L. Zou, L. Chen, and M. Özsu, "K-automorphism: A general framework for privacy preserving network publication," in *Proc. of the VLDB*, 2009.
[20] J. Cheng, A. Fu, and J. Liu, "K-isomorphism: privacy preserving network publication against structural attacks," in *Proc. of ACM COMAD*, 2010.
[21] X. Ying and X. Wu, "On link privacy in randomizing social networks," *Knowledge and Information Systems*, 2011.
[22] C. Tai, P. Yu, D. Yang, and M. Chen, "Privacy-preserving social network publication against friendship attacks," in *Proc. of ACM KDD*, 2011.