Preserving Privacy with Probabilistic Indistinguishability in Weighted Social Networks

Qin Liu, Guojun Wang, Feng Li, Shuhui Yang, *IEEE Member,* and Jie Wu, *IEEE Fellow*

Abstract—The increasing popularity of social networks has inspired recent research to explore social graphs for marketing and data mining. As social networks often contain sensitive information about individuals, preserving privacy when publishing social graphs becomes an important issue. In this paper, we consider the identity disclosure problem in releasing *weighted* social graphs. We identify *weighted 1*-neighborhood attacks*, which assume that an attacker has knowledge about not only a target's one-hop neighbors and connections between them (*1-neighborhood graph*), but also related node degrees and edge weights. With this information, an attacker may re-identify a target with high confidence, even if any node's 1-neighborhood graph is isomorphic with $k-1$ other nodes' graphs. To counter this attack while preserving high utility of the published graph, we define a key privacy property, *probabilistic indistinguishability*, and propose a heuristic indistinguishable group anonymization (HIGA) scheme to anonymize a weighted social graph with such a property. Extensive experiments on both real and synthetic data sets illustrate the effectiveness and efficiency of the proposed scheme.

Index Terms—weighted social networks, weighted 1*-neighborhood attack, probabilistic indistinguishability, privacy.

1 Introduction

The increasing popularity of social networks has stimulated recent research to explore social graph data to understand its structure, advertising and marketing, and data mining. 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. Therefore, publishing the original social network data directly may compromise individuals' privacy, resulting in unacceptable consequences.

A naïve anonymization approach to preserving individuals' privacy simply involves removing nodes' identities before publishing. However, recent studies have shown that an attacker that has background knowledge about a target's degree [1], neighbors [2], or subgraphs [3] can still re-identify the target with high confidence. For example, Zhou et. al. [2] defined a *1-neighborhood graph* as an individual's one-hop neighbors and the connections between them, and also identified *1-neighborhood attacks*, which allow an attacker with knowledge of an individual's *unique* 1-neighborhood graph to re-identify the target from a naïvely anonymized social network. To resist re-identification attacks, the mechanism of *k-anonymity* was adopted by ensuring that each node is indistinguishable from at least $k-1$ others. For example, in a k-neighbor-anonymous social network [2], each node's 1-neighborhood graph is isomorphic to at least $k-1$ other nodes' graphs.

However, existing research is mainly concerned with anonymizing unweighted graphs. In practice, many social networks are intrinsically weighted, where the

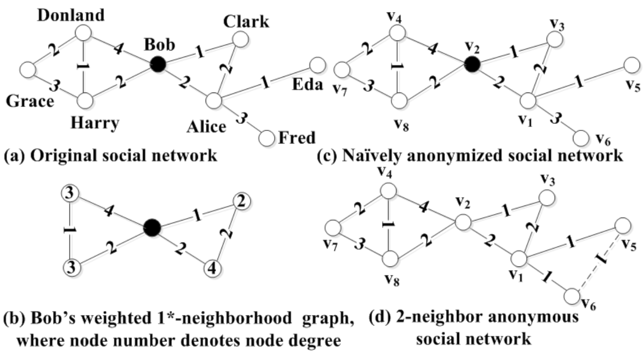

Donland
Bob
Clark
Grace
Harry
Alice
Eda
Fred
(a) Original social network
v_4
v_2
v_3
v_7
v_8
v_1
v_5
v_6
(c) Naïvely anonymized social network
(b) Bob's weighted 1*-neighborhood graph, where node number denotes node degree
(d) 2-neighbor anonymous social network

Fig. 1. Weighted 1*-neighborhood attacks.

- *Qin Liu is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, Hunan Province, P. R. China, 410082. E-mail: gracelq628@hnu.edu.cn*
- *Guojun Wang is with the School of Computer Science and Educational Software, Guangzhou University, Guangzhou, Guangdong Province, P. R. China, 510006. E-mail: csgjwang@gmail.com (corresponding author)*
- *Feng Li is with the Department of Computer and Information Technology, Indiana University-Purdue University Indianapolis, Indianapolis, IN 46202-5160, USA. E-mail: fengli@iupui.edu*
- *Shuhui Yang is with the the Department of Math, Computer Science, and Statistics, Purdue University Calumet, Hammond, IN 46323, USA. E-mail: yang246@purdue.edu*
- *Jie Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA. E-mail: jiewu@temple.edu*

edges have different strengths denoting the affinity between nodes. From both theoretical and practical viewpoints, weighted graphs provide more unique structural information than unweighted graphs, thereby increasing the risk of identity disclosure.

In this paper, we concentrate on publishing weighted social networks in a privacy preserving way. We first identify a *weighted 1*-neighborhood attack*. In addition to the target's 1-neighborhood graph, we assume that an attacker has knowledge about the *degree* of each one-hop neighbor and the *weight* on each edge of the 1-neighborhood graph. We call this kind of information the *weighted 1*-neighborhood graph*, which provides richer background knowledge about the target. Thus, a k-neighbor-anonymous social network still suffers from weighted 1*-neighborhood attacks.

To illustrate this, let us consider a subgraph extracted from a weighted social network, ***actor***, as shown in Fig. 1-(a), where a node denotes an actor, an edge that links two actors denotes previous cooperation between the two in a movie, and the weight on the edge denotes the amount of cooperation between the two actors. Fig. 1-(c) is a naïvely anonymized social network, where node identity is replaced by a random number. If the attacker knows that Bob has co-stared with Alice, Clark, Donland, and Harry and that there have been cooperations between Alice and Clark as well as between Donland and Harry, then Bob's 1-neighborhood graph is exposed. The attacker can re-identify Bob from Fig. 1-(c), where only v_2' neighborhood matches Bob's information. By adding an edge between nodes v_5 and v_6 with the weight of value 1, we have a 2-neighbor-anonymous social network, as shown in Fig. 1-(d), where nodes $\{v_2, v_1\}$, $\{v_4, v_8\}$, and $\{v_3, v_5, v_6, v_7\}$ have isomorphic 1-neighborhood graphs. If an attacker knows only Bob's 1-neighborhood graph, he cannot distinguish Bob from nodes $\{v_2, v_1\}$. However, if the attacker knows that Bob has co-stared with Donland four times and that Donland has cooperated with three actors, Bob's weighted 1*-neighborhood graph (Fig. 1-(b)) is partially exposed, and Bob can be re-identified from Fig. 1-(d) with probability 1.

To resist the weighted 1*-neighborhood attack, we can simply remove all weight information and add more edges to make neighbors' degrees isomorphic. However, the *utility* of the social network will be largely reduced. For example, the trivial solution disables aggregate queries on the social network, such as the average number of cooperations between actors. 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 a weighted 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 main steps: 1) *Node Grouping* groups nodes whose weighted 1*-neighborhood graphs satisfy certain metrics together and provides a combination and splitting mechanism so that each group has an appropriate size; 2) *Approximate Matching Test* determines whether the weighted 1*-neighborhood graphs of any pair of nodes in a group are approximately matching or not, by random-walk-based structural similarity measurement and weight compatibility measurement; 3) *Group Anonymization* utilizes the Graph Approach and Weight Generalization algorithms so that a group of nodes' weighted 1*-neighborhood graphs have similar structures and compatible weights; 4) *Randomization* randomly modifies the graph structure with a certain probability to ensure that each weighted 1*-neighborhood graph has a certain probability of being different from the original one. Steps 1 through 3 enable all nodes in the network to be classified into multiple groups, where a group of nodes' graphs are exactly similar. After Step 4, the nodes in a group still have similar graphs, which have a high probability of deviating from the original structure. In each group from the current weighted 1*-neighborhood graphs of any pair of nodes, the attacker cannot decide whether or not the nodes have the same graphs that they had in the original graph, achieving probabilistic indistinguishability. Our contributions are threefold:

1) We identify a novel weighted 1*-neighborhood attack for publishing privacy preserving weighted social graphs with high utility.
2) We define the probabilistic indistinguishability property for a weighted social network, and we 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.

2 Preliminaries

2.1 Definitions and Notations

In this paper, a social network is modeled as an undirected and weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V}$ is a set of nodes, $\mathcal{E}$ is a set of edges, and $\mathcal{W}$ is a set of weights on the edges. The node identities are anonymized with random numbers. The cardinalities of $\mathcal{V}$ and $\mathcal{E}$, $||\mathcal{V}||$ and $||\mathcal{E}||$, denote the number of nodes and edges in $\mathcal{G}$, respectively. We assume that $||\mathcal{V}|| = n$ and $||\mathcal{E}|| = m$. The nodes of the graph, $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$, denote meaningful entities from the real world such as individuals, organizations, communities and so on. $e_{i,j} \in \mathcal{E}$ is an edge between nodes $v_i, v_j \in \mathcal{V}$, denoting the relationship between a pair of nodes, such as friendship, partnership, co-authorship, and so on. Each edge $e_{i,j}$ is associated with a weight $w_{i,j} \in \mathcal{W}$ which denotes the affinity between nodes v_i and v_j such as the communication frequency between individuals, the similarity between two communities, and so on.

We assume that the attacker has certain background knowledge about the target and that he tries to re-identify the target by analyzing the published social network. To protect the social actors in the network from the re-identification attacks, the social network graph $\mathcal{G}$ will be anonymized to $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{W}')$ before publishing. As in [2], we assume $||\mathcal{V}|| = ||\mathcal{V}'||$ to preserve the global structure of the social network. As in previous work [4], we assume that edge addition and edge deletion are allowed for generating $\mathcal{G}'$.

2.2 Background Knowledge

We assume that an attacker may have background knowledge about the *weighed 1*-neighborhood graphs* of some targets. Following the work in [2], we provide the following definitions:

1-Neighborhood Graph. *$G(v) = (V(v), E(v))$, where $V(v) = v \cup \{u|e_{v,u} \in \mathcal{E}\}$, and $E(v) = \{e_{w,u}|w, u \in V(v) \wedge e_{w,u} \in \mathcal{E}\}$.*

Weighted 1*-Neighborhood Graph. *$G^*(v) = (G(v), D(v), W(v))$, where $G(v)$ is node v's 1-neighborhood graph, $D(v) = [dv_1, \dots, dv_{||V(v)||}]$ is the degree sequence with dv_i denoting the degree of the i-th node in $V(v)$, and $W(v) = [w_1, \dots, w_{||E(v)||}]$ is the weight sequence with w_j denoting the weight on the j-th edge in $E(v)$.*

We focus on the weighted 1*-neighborhood attack since it tends to be much more difficult for an attacker to collect information beyond a one-hop neighborhood [2]. The reason is that the diameter of social networks is small due to the small-world characteristic. Therefore, the attacker has to collect information about many nodes to initiate d-neighborhood attacks, for $d > 1$. It is worth noticing that socialbots [5] can be used as a tool to harvest private user data. The countermeasures proposed in [5] can be employed to alleviate this situation. Furthermore, the detailed neighborhood information about a target's directed neighbors is more difficult to collect than their degree information. For example, it may be easy to know that Bob has a very close friend Alice, who has 100 friends, but it is hard to know detailed information, e.g., ID, or age, regarding these 100 friends.

2.3 Random-Walk-Based Similarity Measurement

Inspired by the work in [6], we use random walk (R-W) [7] as a tool for structural similarity measurement. The basic idea is to perform RW on two graphs to obtain the *steady state distribution* (*topological signature*) for each graph. Then, we calculate the *distance* between topological signatures and determine that two graphs are approximately matching when the distance is smaller than a threshold value.

Specifically, consider graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \dots, v_n\}$. A RW on $\mathcal{G}$ allows the probability $p_v(t)$ of v, located at time t, to be computed with Eq. 1:

$$\begin{aligned} p_v(t) = \textstyle\sum_{u\in\mathcal{V}} p(v|u,j)\cdot(1-df)\cdot p_u(t-1) \\ +\textstyle\sum_{u\in N(v)} p(v|u,l)\cdot df \cdot p_u(t-1) \end{aligned} \quad (1)$$

where $N(v)$ denotes the one-hop neighbors of node $v \in \mathcal{V}$, df is the damping factor which defines the probability of directly jumping or traverse, and $p(v|u,j)$ and $p(v|u,l)$ are the probabilities of moving from u to v by performing jumping or by traversing the edge $e_{u,v}$, respectively, under the requirements of $\forall u \in \mathcal{V}$, $\sum_{v\in\mathcal{V}} p(v|u,j) = 1$ and $\sum_{v\in N(u)} p(v|u,l) = 1$.

In an unlabeled graph, we select the node for a jump over all n nodes in the graph, and for a traversing over all one-hop neighbors, with a uniform probability distribution. Therefore, $p(v|u,j) = 1/n$ and $p(v|u,l) = 1/||N(u)||$. In a labeled graph, we define the tendency of following an edge and jumping from node u to node v as functions f_l and f_j of two nodes labeled $\mathcal{L}_u$ and $\mathcal{L}_v$, respectively. The above tendency functions can be normalized with Eq. 2:

$$p(v|u,l) = \frac{f_l(L_v, L_u)}{\sum\limits_{\omega\in N(u)} f_l(L_\omega, L_u)}; p(v|u,j) = \frac{f_j(L_v, L_u)}{\sum\limits_{\omega\in\mathcal{V}} f_j(L_\omega, L_u)} \quad (2)$$

The probability distribution on all nodes in $\mathcal{G}$, denoted as a vector $\mathbf{p}(t) = [p_{v_1}(t), \dots, p_{v_n}(t)]$, can be described in matrix form as:

$$\mathbf{p}(t) = (\mathbf{\Sigma}\cdot\mathbf{D}_j)'\mathbf{p}(t-1) + (\mathbf{\Delta}\cdot\mathbf{D}_l)'\mathbf{p}(t-1) \quad (3)$$

where $\mathbf{\Sigma}, \mathbf{\Delta}$ are $n\times n$ matrices collecting the probabilities $p(u|v,j)$ and $p(u|v,l)$, respectively, and $\mathbf{D}_j, \mathbf{D}_l$ are $n \times n$ diagonal matrices, with diagonal values $(1-df)$ and df, respectively. The entry (u,v) of matrix $\mathbf{\Delta}$ is not null only when the corresponding entry of the graph adjacency matrix $\mathbf{A}$ is equal to 1, i.e., if the nodes u and v are linked by an edge $e_{u,v}$.

By defining the transition matrix as $\mathbf{T} = (\mathbf{\Sigma}\cdot\mathbf{D}_j + \mathbf{\Delta}\cdot\mathbf{D}_l)'$, Eq. 3 can be written as Eq. 4. The signature of the graph is obtained by considering the steady state distribution $\mathbf{p}^\star$ of the Markov chain defined in Eq. 4.

$$\mathbf{p}(t) = \mathbf{T}\cdot\mathbf{p}(t-1) \quad (4)$$

3 Problem Formulation

To preserve identity privacy, previous research advocated k-anonymity, where any node is 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 with a lower information loss.

Let $G^*(u)$ and $G'^*(u)$ denote the weighted 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 observer's confidence level will be below a pre-determined threshold. In our scheme, we achieve node indistinguishability by introducing randomness into the published graph. If there is no limit on the randomness, node indistinguishability is easily achieved. However, we hope to preserve the usability and utility of the published graph and hence, we need to minimize the information loss. Thus, we need to have a more sophisticated design.

Group Indistinguishability. *For a group of nodes, $g = \{v | v \in \mathcal{V}'\}$ 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 $\{v | v \in \mathcal{V}'\}$ can be classified into $M \geq 1$ groups, where each group has the property of* group indistinguishability.

The information loss is a critical measure in quantifying the utility and usability of an anonymized graph, which will be discussed in Section 7.2. Let $C_{\mathcal{G}\rightarrow\mathcal{G}'}$ denote the information loss incurred in our HIGA scheme. We formally define the probabilistic indistinguishability problem as follows:

The probabilistic indistinguishability problem. *Given a network graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ and a positive integer k, derive an anonymized graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{W}')$ to be published, such that (1) $||\mathcal{V}|| = ||\mathcal{V}'||$; (2) $\mathcal{G}'$ is probabilistically indistinguishable with respect to $\mathcal{G}$; (3) the anonymization from $\mathcal{G}$ to $\mathcal{G}'$ has minimal information loss $C_{\mathcal{G}\rightarrow\mathcal{G}'}$.*

The problem of generating a k-neighbor-anonymous graph is NP-hard [2]. The proof lies in reducing the k-Dimensional Perfect Matching problem [8]. The unweighted graph is a special case of the weighted graph. Therefore, the problem of generating a weighted social network with the above properties is NP-hard.

4 Node Grouping

4.1 Group Formation

For a given weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, we group nodes $\{v\} \in \mathcal{V}$ by using the following metrics: $||V(v)||$, $||E(v)||$, local clustering coefficient, in-degree sequence, out-degree sequence, and weight volume. Here, $||V(v)||$ and $||E(v)||$ denote the numbers of nodes and edges in $G^*(v)$, respectively. The definitions for the other metrics are provided as follows:

Local clustering coefficient. *$C_v = \lambda_G(v)/\tau_G(v)$, where $\lambda_G(v)$ and $\tau_G(v)$ are the numbers of triangles and triples in $G^*(v)$, respectively. Thus, C_v is the proportion of edges between the nodes within v's direct neighbourhood divided by the number of edges that may exist between them.*

In-degree sequence. *$I_v = \{E_u^+\}_{u \in V(v)}$, where E_u^+ is the in-degree of node $u \in V(v)$ denoting the number of edges connected between u and the nodes in $G^*(v)$.*

TABLE 1
Effectiveness of metrics for grouping

Nodes	Edges	Percentage	Nodes	Edges	Percentage
5,000	9,988	75%	5,000	24,970	54%
10,000	19,993	79%	10,000	121,251	58%

Out-degree sequence. *$O_v = \{E_u^-\}_{u \in V(v)}$, where E_u^- is the out-degree of node $u \in V(v)$ denoting the number of edges connected between u and the nodes outside $G^*(v)$.*

Weight volume. *$WV_v = \sum_{w \in W(v)} w$, where WV_v is the sum of the weights on the edges in $G^*(v)$.*

We group nodes together if $||V(v)||$, $||E(v)||$, C_v, I_v, and O_v are equal and the differences of their weight volumes are within a pre-defined scope in their weighted 1*-neighborhood graphs. To test the effectiveness of the above metrics, we use Barab*á*si-Albert (B-A) algorithm [9] to generate synthetic data sets. Then, we use the above metrics to classify nodes into groups and calculate the percentage of graphs that are *isomorphic* in a group. In Table 1, for a social network with $5,000$ nodes and $9,988$ edges, the percentage of isomorphic weighted 1*-neighborhood graphs is about 75%. As the scale of the social networks increases to $10,000$ nodes and $19,993$ edges, this percentage increases to 79%. Therefore, these metrics are helpful for grouping.

4.2 Group Reshaping

Although the Group Formulation process groups the nodes with similar weighted 1*-neighborhood graphs together, not all groups have a size greater than or equal to k. In reality, during the empirical study, we find 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 Group Reshaping mechanism so that each group has an appropriate size, e.g., $[k, 2k)$.

The Group Reshaping mechanism can be classified in two main steps: *Group Combination* and *Group Splitting*. Group Combination first sorts groups in descending order of the maximal node degree of the group members. The sorted groups are denoted as $g_1^1, g_2^1, \ldots, g_{M_1}^1$. For each group g_i^1 of a size smaller than k, we incorporate the members in g_i^1 into g_{i+1}^1 directly. Therefore, after the Group Combination step, all groups will have a size larger than k.

Suppose that there are M_2 groups after combination, denoted as $g_1^2, \ldots, g_{M_2}^2$. Then, we perform the Group Splitting step to enable each group to have $[k, 2k)$ members as follows: For each group g_i^2 of size larger than $2k$, we choose $c = \lceil ||g_i^2||/k \rceil$ nodes in g_i^2 as the group seeds, denoted as $s_1, \ldots, s_c$. The candidate member set, CMS, consists of all the members in g_i^2 except for the c seeds. For each seed s_j, we construct a former member set from CMS, denoted as FMS_j, recording s_j's group members before combination. Then, we classify nodes in CMS that are *far from the seed* in one group, so that the modification of a node's

Algorithm 1 Structure Similarity Measurement

```
Input: $G^*(u)$, $G^*(v)$, $DF = [df_1, \ldots, df_{\mathcal{N}}]$
OutPut: $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*)$
1: Construct structure graphs $\widetilde{G}(u)^*$ and $\widetilde{G}(v)^*$
2: for each node $\omega \in V(u)$ and $\mu \in V(v)$ do
3:     Calculate Eq. 4 for the steady states of $\omega$, $\mathbf{p}^\star_\omega = [\mathbf{p}^\star_\omega(df_1), \ldots, \mathbf{p}^\star_\omega(df_{\mathcal{N}})]$, in $\widetilde{G}(u)^*$, and $\mu$, $\mathbf{p}^\star_\mu = [\mathbf{p}^\star_\mu(df_1), \ldots, \mathbf{p}^\star_\mu(df_{\mathcal{N}})]$, in $\widetilde{G}(v)^*$, under different damping factors, respectively
4: Construct bipartite graph $G_B$
5: Find the optimal matching of $G_B$
6: Calculate the cost of optimal matching with Eq. 6
```

weighted 1*-neighborhood graph has less impact on other nodes' graphs in a group, thereby reducing the information loss during anonymization.

Specifically, for seed s_j, we preferentially choose group members from FMS_j, then from CMS, since the nodes in FMS_j are most similar to s_j. In each selection, we first calculate the *penalty* for selecting node v as follows: 1) $PT = 0$ if v is not the neighbor of existing group members; 2) $PT = 1$ if v is not the neighbor of the seed; 3) $PT = INF$ if v is the neighbor of the seed. Then, we choose the node with the minimal penalty as the group member, and we remove it from CMS and from related $FMSs$. The main idea is based on the classical heuristic clustering algorithm, *K*-means [10]. This process will be run for multiple rounds and in each round, the sum of the penalties will be recorded. At the end of this algorithm, the optimal result, $g_1, \ldots, g_M$, will be the output.

5 Approximate Matching Test

5.1 Structure Similarity Measurement

Alg. 1 determines whether the structures of two n-odes' weighted 1*-neighborhood graphs are similar or not. Let $G^*(u) = (G(u), D(u), W(u))$ and $G^*(v) = (G(v), D(v), W(v))$ be the weighted 1-*neighborhood graphs of nodes u and v in graph $\mathcal{G}$, respectively. The first step is to construct the *structure graphs* for nodes u and v. Specifically, for node u, the structure graph $\widetilde{G}(u)^*$ is first initialized with its 1-neighborhood graphs $G(u)$. Then, for each of u's one-hop neighbors, ω, we add $o_{u\omega}$ *dummy nodes* to $\widetilde{G}(u)^*$ and add edges between ω with all the dummy nodes, where $o_{u\omega}$ is the out-degree of ω. For example, for a given weighted 1*-neighborhood graph as shown in Fig. 1-(b), the structure graph is shown in Fig. 2-(a).

We label non-dummy nodes and dummy nodes with $\widetilde{L}$ and $\overline{L}$, respectively. Then, we provide the tendency functions $\{f_l(L_u, L_v)\}$ and $\{f_j(L_u, L_v)\}$, where $L_u, L_v \in \{\widetilde{L}, \overline{L}\}$. The probabilities of following an edge or jumping from node u with $\mathcal{L}_u$ to node v with $\mathcal{L}_v$ can be calculated with Eq. 2. For each *non-dummy* node $\omega \in \widetilde{G}(u)^*$, we obtain its topological

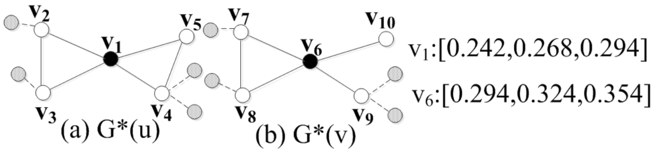


Fig. 2. Sample structure graph. The solid lines connect the target and its neighbors. $DF = [0.7, 0.8, 0.9]$.

signature with the RW tool. To enrich the topological signatures, we can compute the steady states with different damping factors. Suppose that we choose $\mathcal{N}$ distinct damping factors $DF = [df_1, \ldots, df_{\mathcal{N}}]$; node ω's topological signature can be defined as $\mathbf{p}^\star_\omega = [\mathbf{p}^\star_\omega(df_1), \ldots, \mathbf{p}^\star_\omega(df_{\mathcal{N}})]$. Given two graphs' topological signatures $\mathbf{p}^\star(\widetilde{G}(u)^*)$ and $\mathbf{p}^\star(\widetilde{G}(v)^*)$, we determine the structural similarity of $\widetilde{G}(u)^*$ and $\widetilde{G}(v)^*$ by performing bipartite graph matching as follows.

We first add *virtual nodes* to the structure graph with a smaller node cardinality so that two graphs have equal numbers of nodes. Then, we create a bipartite graph $G_B = (V_B, E_B)$ by setting $V_B = V(u) \cup V(v)$ and $E_B = V(u) \times V(v)$. A $cost(\omega, \mu)$ is associated with the edge $e_{\omega,\mu}$ to represent the cost of matching nodes $\omega \in V(u)$ and $\mu \in V(v)$. A match in G_B is a set of edges $E^\diamond \subseteq E_B$ such that each node is associated with only one edge. The cost of matching is the sum of the cost of edges in $E^\diamond$. An optimal match will cause the minimum cost match in G_B under the constraint that node u is matched with node v. We apply the Monte-Carlo algorithm [11] to determine the optimal matching on a bipartite graph in polynomial time.

The cost function will impact the result of the optimal match. Let $\mathbb{V}$ denote a set of virtual nodes. If either ω or μ is a virtual node, $cost(\omega, \mu)$ is set to a fixed value β. If $\omega, \mu \notin \mathbb{V}$, the cost function can be defined as the Hellinger Distance between probability distributions $\mathbf{p}^\star_\omega$ and $\mathbf{p}^\star_\mu$ with Eq. 5:

$$H(\mathbf{p}^\star_\omega, \mathbf{p}^\star_\mu) = \frac{1}{2}\sum_{i=1}^{\mathcal{N}} \left(\sqrt{\mathbf{p}^\star_\omega(df_i)} - \sqrt{p^\star_\mu(df_i)}\right)^2 \tag{5}$$

Therefore, the cost of matching the bipartite graph, which is the sum of the costs of matching all nodes in $V(u)$ and $V(v)$, can be calculated with Eq. 6:

$$cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*) = \sum_{\omega,\mu \notin \mathbb{V}} H(\mathbf{p}^\star_w, \mathbf{p}^\star_\mu) + (||\mathbb{V}|| \cdot \beta) \tag{6}$$

For example, given the topological signatures as shown in Fig. 2, the cost for matching nodes v_1 and v_6 is $H(\mathbf{p}^\star_{v_1}, \mathbf{p}^\star_{v_6}) = 0.5((\sqrt{0.242} - \sqrt{0.294})^2 + (\sqrt{0.268} - \sqrt{0.324})^2 + (\sqrt{0.294} - \sqrt{0.354})^2) = 0.004$.

Structure Similarity. *For a given threshold α, two nodes u and v are structurally similar if an optimal bipartite graph matching exists between $V(u)$ and $V(v)$, such that $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*) \leq \alpha$.*

5.2 Weight Compatibility

The weight on an edge may be a concrete value or a range. Let w_i denote the value of weight on the i-th

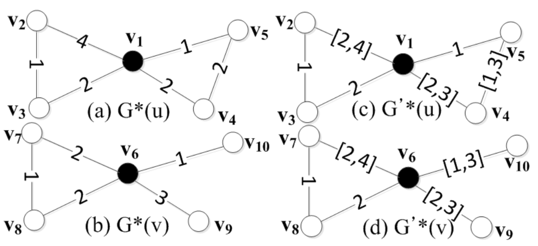


Fig. 3. Sample of possible candidates.

edge. We consider w_i compatible with w_j, denoted as $w_i \preceq w_j$, if $w_i == w_j$ or $w_i \subseteq w_j$. For example, if $w_1 = 1$, $w_2 = 1$, and $w_3 = [1,5]$, we have $w_1 \preceq w_2$, $w_2 \preceq w_1$, $w_1 \preceq w_3$, and $w_2 \preceq w_3$. Let $G^*(v) = (G(v), D(v), W(v))$ and $G'^*(v) = (G'(v), D'(v), W'(v))$ be node v's weighted 1-*neighborhood graphs in the original graph $\mathcal{G}$ and the published graph $\mathcal{G}'$, respectively. Inspired by the work in [12], [13], we provide the following definitions:

Possible Candidate. *$G^*(u)$ is a possible candidate of $G'^*(v)$ if there is a mapping $\phi : E(u) \to E'(v) \bigcup \emptyset$, such that either $e_{\omega,x} \in E(u)$ is mapped to $\emptyset$ or $e_{\omega,x} \in E(u)$ is mapped to $e_{\mu,y} \in E'(v)$ and $w_{\omega,x} \preceq w_{\mu,y}$, where $E(u) \in G(u)$ and $E'(v) \in G'(v)$.*

For example, the weight sequences of $G^*(u)$, $G'^*(u)$, $G'^*(v)$ are $[4, 2, 2, 1, 1, 2]$, $[[2, 4], 2, [2, 3], 1, 1, [1, 3]]$, and $[[2, 4], 2, [2, 3], [1, 3], 1]$ in Fig. 3, respectively . For $G^*(u)$ and $G'^*(u)$, we have $4 \preceq [2, 4]$, $2 \preceq 2$, $2 \preceq [2, 3]$, $1 \preceq 1$, $1 \preceq 1$, and $2 \preceq [1, 3]$. For $G^*(u)$ and $G'^*(v)$, we have $4 \preceq [2, 4]$, $2 \preceq 2$, $2 \preceq [2, 3]$, $1 \preceq [1, 3]$, and $1 \preceq 1$, with the expectation that $e_{4,5} \in E(u)$ is mapped to $\emptyset$. Therefore, $G^*(u)$ is a possible candidate of $G'^*(u)$ and $G'^*(v)$. That is to say, the attacker with knowledge of $G^*(u)$'s weight sequence in $\mathcal{G}$ cannot distinguish node u from node v in $\mathcal{G}'$ with high confidence.

Weight Compatibility. *Node u is weight compatible with node v in graph $\mathcal{G}'$, if $G^*(u)$ and $G^*(v)$ in $\mathcal{G}$ are possible candidates of $G'^*(u)$ and $G'^*(v)$ in $\mathcal{G}'$.*

For example, in Fig. 3, $G^*(u)$ is a possible candidate of $G'^*(u)$ and $G'^*(v)$, and $G^*(v)$ is also a possible candidate of $G'^*(u)$ and $G'^*(v)$. Therefore, nodes u and v are weight compatible.

Given a pair of nodes u and v, we define that they are approximately matching as follows:

Approximate matching. *Node u and node v are approximately matching, denoted as $u \approx v$, if they are structurally similar and weight compatible.*

6 Anonymization and Randomization

6.1 Group Anonymization

Suppose that there are M groups, $g_1, \ldots, g_M$, where each group has about $[k, 2k)$ members. In Alg. 2, the candidate group set (CGS) initially consists of M groups, and the processed group set (PGS) is null. We sort groups in descending order of the maximal node degree in a group, and pick the first one as the *processing group*, denoted as $\widetilde{g}$. For each pair of nodes in $\widetilde{g}$, we call Alg. 1 to test whether they are structurally similar. If not all nodes are indistinguishable from each other, we choose the node with the maximal degree, say node u, as the *group seed*. Then, for the node $v \in \widetilde{g}$ with $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*) \geq \alpha$, we call the Graph Approach algorithm (Alg. 3) to make $G^*(u)$ and $G^*(v)$ structurally approach each other. This process will continue until any pair of nodes are structurally similar. For a group of structurally similar nodes, we use the Weight Generalization algorithm (Alg. 4) to make all nodes in a group have compatible weights. Finally, we move $\widetilde{g}$ from CGS to PGS.

Algorithm 2 Group Anonymization

```
Input: $\mathcal{G} = g_1, \ldots, g_M$
OutPut: $\widehat{\mathcal{G}} = \widehat{g}_1, \ldots, \widehat{g}_M$
1: Initialize $\widehat{g}_1, \ldots, \widehat{g}_M$ with $g_1, \ldots, g_M$, respectively
2: Sort $\widehat{g}_1, \ldots, \widehat{g}_M$ with descending order of the maximal node degree of group members
3: Set $CGS = \{\widehat{g}_1, \ldots, \widehat{g}_M\}$ and $PGS = \{\}$
4: while $CGS$ is not empty do
5:   Set the first group in $CGS$ as $\widetilde{g}$
6:   repeat
7:     for each pair of nodes $(u, v)$ in $\widetilde{g}$ do
8:       Call Alg. 1 to obtain $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*)$
9:     Choose node $u \in \widetilde{g}$ with the maximal degree as the group seed
10:    for each node $v \in \widetilde{g}$ do
11:      if $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*) \geq \alpha$ then
12:        Call Alg. 3 to approach $\widetilde{G}(v)^*$ to $\widetilde{G}(u)^*$
13:  until all nodes in $\widetilde{g}$ are structurally similar
14:  Call Alg. 4 to generalize the weights in $\widetilde{g}$
15:  Move $\widetilde{g}$ from $CGS$ to $PGS$
```

Given a pair of nodes u and v, Alg. 3 approaches $\widetilde{G}(v)^*$ to $\widetilde{G}(u)^*$ as follows. For each pair of optimal matching nodes $\omega \in V(u)$ and $\mu \in V(v)$ if $cost(\omega, \mu) \geq \theta$, we make their out-degree equal by adding edges between dummy nodes and the node with a smaller out-degree and setting the weights as ∞. Then, for each edge $e_{\omega,x} \in E(u)$, if $e_{\mu,y} \notin E(v)$, we add an edge between nodes μ and y and set $w_{\mu,y} = w_{\omega,x}$, where $\mu, y \in V(v)$ are the optimal matching nodes of $\omega, x \in V(u)$. During the approaching process, new edges will be added between the nodes in $G^*(v)$ and those not in $G^*(v)$. Specifically, the node with the minimal degree will be chosen provided that it is not a member in $\tilde{g}$ and no edge existed before. Furthermore, nodes will be preferentially chosen first from CGS and then from PGS. The anonymization process may be recursive, since some changes may impact the groups that have been processed previously. For example, if the dummy nodes are chosen from PGS, related groups must be moved to CGS. However, due to the power-law degree distribution and the small world phenomenon, this process will rapidly stop.

Alg. 4 ensures that the graphs of nodes with similar structures have compatible weights. We choose a node

Algorithm 3 Graph Approach

```
1: Obtain the optimal matching of $\widetilde{G}(u)^*$ and $\widetilde{G}(v)^*$
2: for each pair of optimal matching nodes $\omega \in V(u)$ and $\mu \in V(v)$ do
3:   if $cost(\omega, \mu) \geq \theta$ then
4:     Make the out-degrees of $\omega$ and $\mu$ equal
5:     Set the weights on newly added edges as $\infty$
6:     for each edge $e_{\omega,x} \in E(u)$ do
7:       if edge $e_{\mu,y} \notin E(v)$, where $\mu, y \in V(v)$ are the optimal matching nodes of $\omega, x \in V(u)$ then
8:         Add edge $e_{\mu,y}$ in $E(v)$, set $w_{\mu,y} = w_{\omega,x}$
```

with the maximal degree, say node u, as the group seed. Then, for each edge $e_{\omega,x}$ in $\widetilde{G}(u)^*$, we find the optimal matching nodes of nodes ω, x in other nodes' structure graphs. Let ω_i, x_i denote the optimal matching nodes of ω, x in the i-th member of $\widetilde{g}$. If e_{ω_i,x_i} exists, we set $w_{\omega_i,x_i} = [\min(\{w_{\omega_i,x_i}\}), \max(\{w_{\omega_i,x_i}\})]$. Note that an edge occurring in multiple nodes' weighted 1*-neighborhood graphs would be generalized multiple times with different range values. We choose the extended range as the result. For example, if $w_{u,v}$ is generalized to $[1, 2]$ and $[2, 3]$ in $G^*(u)$ and $G^*(v)$, respectively, we extend $w_{u,v}$ to $[1, 3]$. Furthermore, for a given weight value $w \geq 0$, we have $\min(w, \infty) \equiv \max(w, \infty) \equiv w$ and $\min(\infty, \infty) \equiv \max(\infty, \infty) \equiv \infty$. After the Group Anonymization step, we find that the weights on a fraction of edges are not processed. For edge $e_{u,v}$ with weight value ∞, we first set $w_{u,v} = SPW(u, v)$, where $SPW(u, v)$ is the sum of weights on the shortest path from node u to node v in $\mathcal{G}$, and then generalize $w_{u,v}$ with the Weight Generalization algorithm. This enables the anonymized graph to have a weight distribution close to the original graph for t-closeness.

Fig. 4 shows the working process of the Graph Approach and Weight Generalization. The optimal matching node pairs for graphs $G^*(u)$ and $G^*(v)$ include $\langle v_1, v_6\rangle$, $\langle v_2, v_7\rangle$, $\langle v_3, v_8\rangle$, $\langle v_4, v_9\rangle$, $\langle v_5, v_{10}\rangle$. For example, in Fig. 4-(b), the out-degrees of v_2 and v_7 are 1 and 0, respectively. Therefore, we add one dummy node to v_7, and set the weight as ∞ to make two nodes have equal out-degrees. For edge $e_{4,5} \in G^*(u)$, we find that edge $e_{9,10} \notin G^*(v)$. Therefore, we bridge v_9 and v_{10} with the same weight value of $e_{4,5}$, say 2, to make a pair of optimal matching nodes have identical links. In Fig. 4-(c), the weights on both $e_{2,3}$ and $e_{7,8}$ are 1, which will be generalized to 1. The weights on $e_{1,2} \in G^*(u)$ and $e_{6,7} \in G^*(v)$ are 4 and 2, respectively, which will be generalized to $[2, 4]$.

6.2 Randomization

Consider a graph $\widehat{\mathcal{G}}$ and a randomization probability p. We randomize the graph to generate an anonymized graph $\mathcal{G}'$ with the edge-based graph

Algorithm 4 Weight Generalization

```
1: Choose node $u \in \widetilde{g}$ with the maximal degree as the group seed
2: for each edge $(\omega, x) \in G(u)$ do
3:   Find optimal matching nodes $\omega_i$ and $x_i$ of $\omega$ and $x$ in the 1-neighborhood graph of the $i$-th member of $\widetilde{g}$
4:   if edge $e_{\omega_i,x_i}$ exists then
5:     Generalize $w_{\omega_i,x_i}$ with $[\min(\{w_{\omega_i,x_i}\}), \max(\{w_{\omega_i,x_i}\})]$
```

perturbation strategies [14]. We first randomly switch a pair of existing edges $e_{\omega,x}$ and $e_{\mu,y}$ to $e_{\omega,\mu}$ and $e_{x,y}$ under the condition that $e_{\omega,\mu}$ and $e_{x,y}$ do not exist in $\widehat{\mathcal{G}}$. Then, we set $w_{\omega,\mu} = SPW(\omega, \mu)$ and $w_{x,y} = SPW(x, y)$ to make the weight distribution close to the original one, where $SPW(x, y)$ is the sum of weights on the shortest path from x to y in $\mathcal{G}$. This process will repeat $\lfloor p * ||\mathcal{E}||\rfloor$ times. After the Randomization step, the randomized graph is expected to be different from the original one. The key problem lies in determining p to randomize the graph. As discussed in [15], the utility of the social network will be largely reduced when the perturbation rate reaches about 10%. To obtain a reasonable parameter, we conduct experiments in Section 8.1.

7 Analysis

7.1 Anonymization Strength

Theorem 1. *From the anonymized graph $\mathcal{G}'$, an attacker with the knowledge of any target's weighted 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 weighted 1*-neighborhood graph $G^*(t)$ in the original graph (attacker's knowledge). There are two possible consequences after searching $\mathcal{G}'$:

- **Case 1.** The attacker found at least an exact match of the target.
- **Case 2.** The attacker cannot find an exact match of the 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 $P(G^*(u) = G^*(t) \wedge G'^*(u) = G'^*(t)) = (1 - p)^{2\cdot||E(t)||}$.

To identify whether or not node u is the target node t, $G^*(u)$ is the only subgraph that is of concern, no matter whether $G^*(u)$ overlaps with other nodes' 1*-neighborhood graphs or not.

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 $P(G^*(v) \neq G^*(t) \wedge G'^*(v) = G'^*(t)) \geq (1-p)^{2\cdot||E(t)||}$ (otherwise, v and t will violate the indistinguishability requirement in g). Let $\tau = (1-p)^{2\cdot||E(t)||}$. Therefore, under Case 1 with the first strategy, the probability that an exact match node u is the correctly identified target t is:

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

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 heuristically indistinguishable group anonymization scheme ≥ k-anonymity social network defined in [2].*

Proof: The proof of Corollary 1 is obvious. First, the k-anonymity social network defined in [2] assumes that the attacker only knows a target's 1-neighborhood, which contradicts the reality since the attacker usually collects more information about one-hop neighbors than only the connection information between them. With the weighted 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 information loss due to the exact matching), the k-anonymity social network only guarantees that the attacker cannot identify the target with a confidence higher than $1/k$. According to Theorem 1, our scheme will produce equal or greater anonymization strength. Since a group of generalized weights will not provide more useful background knowledge to the attacker, we consider only the unweighed graphs in the above proof. □

7.2 Information Loss

Our solution anonymizes a graph by Graph Approach, Weight Generalization, and Randomization. The entire process requires the addition and deletion of edges and the generalization of weights, and thus will lead to some information loss.

Consider a weighted social network graph $\mathcal{G}$, where $MAXW$ denotes the maximal weight. Let $null$ denote the weights associated with nonexistent edges. We generalize all operations to *weight transformation* as follows: (1) *edge addition* transforms the weight from $null$ to a particular value; (2) *edge deletion* transforms a particular weight to $null$; (3) *weight generalization* transforms a particular weight to an interval. Note that edge addition/deletion not only changes the node degree, but also the related edge weight in $\mathcal{G}$; this incurs more information loss compared with weight generalization on existing edges.

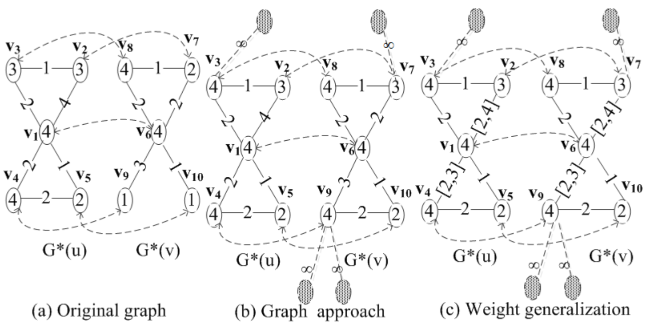


Fig. 4. Working process of Graph Approach and Weight Generation. The dashed, double-sided arrow connects a pair of optimal matching nodes, and the number on the node denotes node degree.

To capture this concept, the information loss for transforming weight w_j to w'_j can be calculated as:

$$\phi(w_j, w'_j) = \begin{cases} \frac{|\max w'_j - \min w'_j|}{MAXW}, & w'_j \neq null \wedge w'_j \neq null \\ \frac{\rho \cdot DIFF}{MAXW}, & otherwise \end{cases}$$

where $[\max w'_j, \min w'_j]$ denotes the weight interval assigned to related edge in $\mathcal{G}'$, and $DIFF = \max\{|\max w'_j - \min w'_j|\}_{w'_j \in \mathcal{G}'}$ is the worst-case cost for $|\max w'_j - \min w'_j|$. When the factor ρ is larger than 1, the above definition can guarantee that the information loss of generalizing existing edges is less than that of edge addition/deletion.

To generate an anonymized graph $\mathcal{G}'$ from $\mathcal{G}$, our scheme first classifies the nodes into M groups, $g_1, \ldots, g_M$. For each group g_i, we transfer a set of original weighted 1*-neighborhood graphs $\{G^*(u)|u \in g_i\}$ to a set of indistinguishable weighted 1*-neighborhood graphs $\{G'^*(u)|u \in g'_i\}$. Let L_i denote the number of edges in $\{G^*(u)|u \in g_i\}$, and let m_i and n_i denote the number of added edges and removed edges for transforming g_i to g'_i, respectively. The information loss $C_{g_i \to g'_i}$ for transferring g_i to g'_i can be calculated with Eq. 7:

$$C_{g_i \to g'_i} = a\frac{\sum_{j=1}^{m_i} \phi(w_j, w'_j)}{L_i} + b\frac{\sum_{j=1}^{n_i} \phi(w_j, w'_j)}{L_i} + c\frac{\sum_{j=1}^{l_i} \phi(w_j, w'_j)}{L_i} \quad (7)$$

where $l_i = L_i + m_i - n_i$ denotes the number of edges in $\{G'^*(u)|u \in g'_i\}$, and a, b, and c are the weights associated with each component. Therefore, given M groups, the total information loss for anonymizing $\mathcal{G}$ to $\mathcal{G}'$ is calculated with Eq. 8:

$$C_{\mathcal{G} \to \mathcal{G}'} = \sum\nolimits_{i=1}^{M} C_{g_i \to g'_i} \quad (8)$$

7.3 Performance Analysis

Time complexity. Let $||\mathcal{V}||$ and $||\mathcal{E}||$ denote the number of nodes and edges in graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, respectively. Group Formation takes $O(M_1 \cdot ||\mathcal{V}||)$ to classify nodes into M_1 groups. Group Reshaping first utilizes the Quicksort algorithm to sort groups, which takes $O(M_1 \cdot \log M_1)$. It then takes $O(M_1)$ to combine groups, and $O(M_2 \cdot R \cdot ||g||^2/k)$ to split groups into subgroups with sizes between $[k, 2k)$, where M_1 and M_2 are the numbers of groups before and after group combination, respectively, R is the number of iterations, and $||g||$ is the maximal group size before splitting. The most expensive operation in Randomization is the calculation of *SPW* between a pair of nodes. The Dijkstra algorithm is used to solve the single-source shortest path problem, which takes $O(||\mathcal{V}|| \cdot \log ||\mathcal{V}|| + ||\mathcal{E}||)$ while it is implemented with the priority queue.

The most time consuming algorithm is Group Anonymization. Given M groups of a size between $[k, 2k)$, it first utilizes the Quicksort algorithm, which takes $O(M \cdot \log M)$ to sort groups. Then, it takes $O(M \cdot k^2 \cdot T_1)$ to perform a structure similarity measurement where T_1 is the average overhead for Alg. 1. Given $\widetilde{G}(u)^*$ and $\widetilde{G}(v)^*$, Alg. 1 takes about $O(\hat{n} \cdot \hat{m})$ for the random walk and $O(\hat{n} \cdot \log^6 \hat{n}/\varepsilon^3)$ for the Monte-Carlo algorithm, where $\hat{n}$ and $\hat{m}$ are the maximal numbers of nodes and edges in $\widetilde{G}(u)^*$ and $\widetilde{G}(v)^*$, and $\varepsilon > 0$ is a real number. Next, Group Anonymization takes $O(M \cdot k \cdot T_2)$ for Graph Approach, and $O(M \cdot T_3)$ for Weight Generalization, where T_2 and T_3 are the average overheads for Alg. 3 and Alg. 4, respectively. Given the optimal matching of $\widetilde{G}(v)^*$ and $\widetilde{G}(u)^*$, Alg. 3 is very efficient for adding edges between nodes that exist in $\widetilde{G}(v)^*$. For adding an edge between the node that exists in $\widetilde{G}(v)^*$ and that is not in $\widetilde{G}(v)^*$, it takes $O(||\mathcal{V}||)$ to find the node with the minimal degree in the graph. To ensure a group of nodes have compatible weights, Alg. 4 takes $O(\hat{e} \cdot k)$ for weight generalization, where $\hat{e}$ is the maximal number of edges in the group.

Memory complexity. Graph information is recorded in a $3 \times ||\mathcal{E}||$ matrix *EdgeMatrix*. Group Formulation takes *EdgeMatrix* as the input and outputs the following matrixes: *Attribute* recording ID, metrics, and group ID for each node, and *Neighbor* recording neighbor IDs and associated weights for each node. The memory complexity is $O(||\mathcal{E}|| + ||\mathcal{V}|| \cdot \hat{D})$ where $\hat{D}$ is the maximal degree for nodes in $\mathcal{G}$. Group Reshaping takes *Attribute* and *Neighbor* as inputs and outputs the following matrixes: *GroupInfo* recording the size and maximal degree for M group and *GroupMember* recording the member IDs for M group. The complexity is $O(M + M \cdot k)$. Moreover, CMS and FMS will be constructed and released in the runtime, the size of which relates to the group size.

Alg. 2 first uses dynamic arrays to store CGS and PGS, the sizes of which are $O(M)$. To calculate $cost(\widetilde{G}(u)^*, \widetilde{G}(v)^*)$, it constructs a $\hat{n} \times \hat{n}$ matrix for the random walk and optimal matching of the bipartite graph, where $\hat{n}$ is the maximal number of nodes in $\widetilde{G}(u)^*$ and $\widetilde{G}(u)^*$. Furthermore, for a group of k nodes, a $k^2 \times \hat{D}$ matrix *MatchList* is constructed to record the optimal matching nodes where $\hat{D}$ is the maximal node degree in the group. The generated matrixes will be released at the end of operation, and the memory consumed will not increase with time. In Randomization, the Dijkstra algorithm is implemented with the adjacency list, the size of which is $O(||\mathcal{V}||)$.

8 Evaluation

In this section, we will analyze the performance of our HIGA scheme and evaluate it on synthetic and real data sets. Experiments are conducted on a local machine, running the Microsoft Windows 7 Ultimate operating system, with an Intel Core 2 Duo E8400 3.0 GHz CPU and 8 GB RAM. The programs are implemented in C++, compiled using Dev C++ 5.4.0.

8.1 Parameter Setting

Let $\widetilde{L}$ and $\overline{L}$ denote the label associated with non-dummy nodes and dummy nodes, respectively. Tendency functions are defined as follows: $f_l(L_u, L_v) = f_l(L_v, L_u) = 0.5$, where $L_u, L_v \in \{\widetilde{L}, \overline{L}\}$; $f_j(L_u, L_v) = 0.95$, where L_u, L_v have the same labels and $f_j(L_u, L_v) = 0.05$, where L_u, L_v have different labels. Furthermore, the damping factors $DF = [0.7, 0.8, 0.9]$, and the parameters a, b, and c in Eq. 7 are set to 0.5, 0.4, and 0.1, respectively. We consider that removing edges will lead to loss of most of the information.

To obtain a reasonable threshold value α for a given structure graph $\widetilde{G}(u)^*$, we first generate a similar graph $\widetilde{G}(u)'$ by randomly modifying p_c percentage of edges in $\widetilde{G}(u)^*$. Then, we calculate the cost for optimal matching $\widetilde{G}(u)^*$ and $\widetilde{G}(u)'$. The above process will be conducted for multiple rounds, and the average value is used as the threshold value α. In our experiments, p_c is set to 10% and the value of α associated with each node will be dynamically extracted based on different experimental data sets. To obtain a reasonable randomization probability p, we perturb a graph with different p values and calculate the percentage P of structure graphs being changed in the randomized graph. This process will be done multiple times, and the average value will be used to measure the impact of p. Table 2 shows the relationship between p and P for real data sets. As shown in Table 3, the average node degrees (AVE) for Facebook, CA-CondMat, Enron, and Douban are 44, 8, 10, and 4.2, respectively. Thus, for a given P, the higher the AVE, the lower p.

The weights assigned on edges are in $[1, 80]$ following the power-law distribution. To evaluate the utility of the anonymized graph, we test the following metrics: average clustering coefficient (AC-

TABLE 2
Parameter p vs. percentage P on real data sets

	Nodes	Edges	p	P
Facebook[2]	4,039	88,234	0.0003	48.9%
CA-CondMat[3]	23,133	93,497	0.01	45.4%
Enron[4]	36,692	183,831	0.002	51%
Douban[5]	154,907	327,094	0.02	49.9%

C), degree distribution, weight distribution, minimal/maximal/average degree (MIN/MAX/AVE), average shortest path weight (APW), and average shortest path length (APL). Specifically, the ACC of graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ can be calculated as $(1/||\mathcal{V}||)\sum_{v\in\mathcal{V}} C_v$, where C_v is the local clustering coefficient for node $v \in \mathcal{V}$ (defined in Section 4.1). In the anonymized network $\mathcal{G}'$, the weight on the edge may be generalized as a range. Therefore, we uniformly sampled from the generalized weight range $w_{u,v}$ in $\mathcal{G}'$ to denote the weight on edge $e_{u,v}$ for answering APW and APL queries. In the experiments, we randomly choose $\mathcal{P} = 20,000$ pairs of nodes in the networks based on the sampling algorithms proposed in [16]. For each pair of nodes u and v, we calculate the shortest paths between them in $\mathcal{G}$ and $\mathcal{G}'$, denoted as paths $path_{uv}$ and $path'_{uv}$, respectively. The shortest path weight of $path_{uv}$ and $path'_{uv}$, denoted as $SPW(u,v)$ and $SPW'(u,v)$, are the sum of weights on the edges in $path_{uv}$ and $path'_{uv}$, respectively. Given $\mathcal{P}$ pairs of nodes, we have $APL = (1/\mathcal{P})\sum_{i=1}^{\mathcal{P}} ||path_{u_iv_i}||$ and $APW = (1/\mathcal{P})\sum_{i=1}^{\mathcal{P}} SPW(u,v)$ for $\mathcal{G}$, and the anonymized graph can be calculated in the same way.

8.2 Synthetic Data Set

We use the B-A [9] and R-MAT models [17] to generate synthetic data sets. Both models maintain two major properties of real social networks: the graphs show small-world characteristics, and node degrees follow power-law distribution [18]. In our experiments, the generated networks contain $n = 5,000 \sim 25,000$ nodes, where the AVE ranges from 4 to 10. To enable the percentage of the modified structure graph to approach 50%, the randomization probability p is set to 0.01 for B-A networks with AVE=4, 0.002 for B-A networks with AVE=10, 0.015 for R-MAT networks with AVE=4, and 0.004 for R-MAT networks with AVE=10.

The subfigures (a)$\sim$(d) in Figs. 5 and 7 show the number of modified edges and running time in our scheme with respect to a different group size, k, and different graph settings. We know that as k or n increases, the number of modified edges as well as the running time increases. Furthermore, our method performs better in R-MAT networks than B-A networks. For example, when $k = 5$, our scheme takes 513s and changes 2,195 edges for the R-MAT network and takes 3,805s and changes 7,102 edges for the B-A network under the setting of $n = 25,000$ and $AVE = 4$. Figs. 6-(a)$\sim$(d) show the related information loss calculated with Eq. 8. We know that the information loss increases as k increases or as n decreases. In both networks, we observed that our HIGA scheme has a better performance as the AVE increases.

In the experiments, we found that two major factors impact the ACC: the AVE and the number of nodes n. Actually, the ACC increases as the AVE increases or as n decreases. Furthermore, the performance of our scheme is mainly affected by the AVE and by the number of nodes other than the ACC. For example, given the fixed $k = 25$ and $AVE = 10$, the percentage of modified edges decreases from to 14% to 10.7% in B-A networks and from to 6.7% to 3.2% in R-MAT networks while n increases from 5,000 to 25,000 (resp. the ACC decreases from 0.0123 to 0.003 in B-A networks and from 0.0068 to 0.0019 in R-MAT networks). However, Fig. 5-(e) shows a different trend. Figs. 5$\sim$7-(e) show the impact of the ACC on the performance of our HIGA scheme under the fixed setting of $n = 1,000$ and $k = 5$.

8.3 Real Data Set

Graph characteristics of the real data sets are shown in Table 2. We group nodes based on certain metrics. In Facebook, nodes are classified in 3,137 groups, where group sizes range from 1 to 43. In CA-Condmat, nodes are classified in 3,439 groups, where group sizes range from 1 to 2,836. In Enron, nodes are classified in 4,333 groups, where group sizes range from 1 to 3,706. In Douban, nodes are classified in 3,585 groups, where group sizes range from 1 to 89,765.

From Fig. 8, we know that as k increases, our HIGA scheme performs worse. For example, as k increases from 5 to 25, the information loss in Douban increases from 0.026 to 0.078, and the running time for Enron increases from $1.6h$ to $4.6h$. Facebook and Enron incur a higher growth rate in terms of information loss compared with other real data sets. This is mainly because of the difference between the maximal degree and the minimal degree in the network. As shown in Table 3, the minimal/maximal degrees in Facebook, CA-Condmat, Enron, and Douban are 1/1,045, 1/279, 1/1,383, and 1/287, respectively. Given a network with a bigger degree difference, our method needs to add more edges to ensure a group of nodes have similar 1*-neighborhood graphs. In the worst case, the percentage of modified edges is 23% in Facebook, 7.2% in CA-Condmat, 13.2% in Enron, and 17% in Douban. For the Facebook data set with a small number of nodes and a large degree difference, the percentage of modified edges grows rapidly (from 4% to 23%, as k increases from 5 to 25).

In addition, our running time grows linearly as the number of nodes/edges increases. For example, while

2. http://snap.stanford.edu/data/egonets-Facebook.html
3. http://snap.stanford.edu/data/ca-CondMat.html
4. http://snap.stanford.edu/data/email-Enron.html
5. http://socialcomputing.asu.edu/datasets/Douban

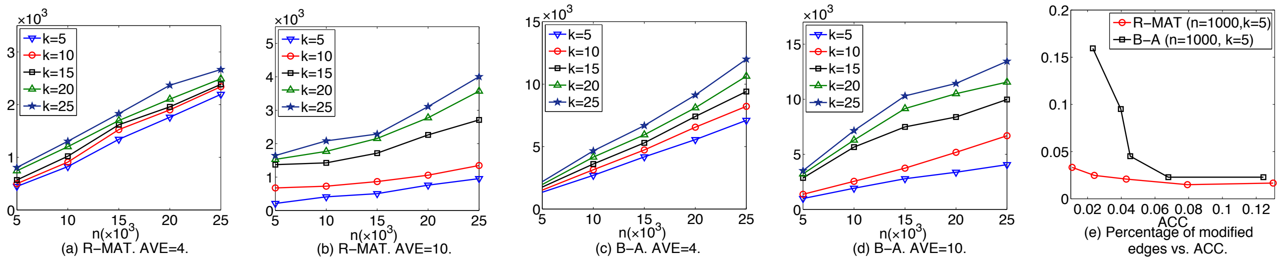


Fig. 5. Number of modified edges.

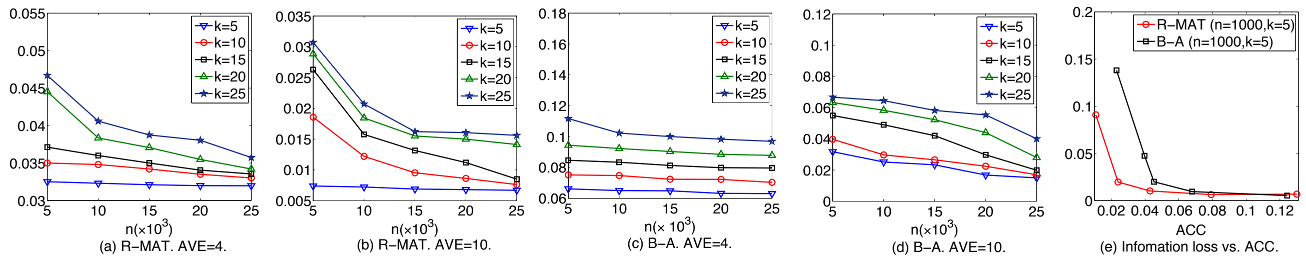


Fig. 6. Information loss.

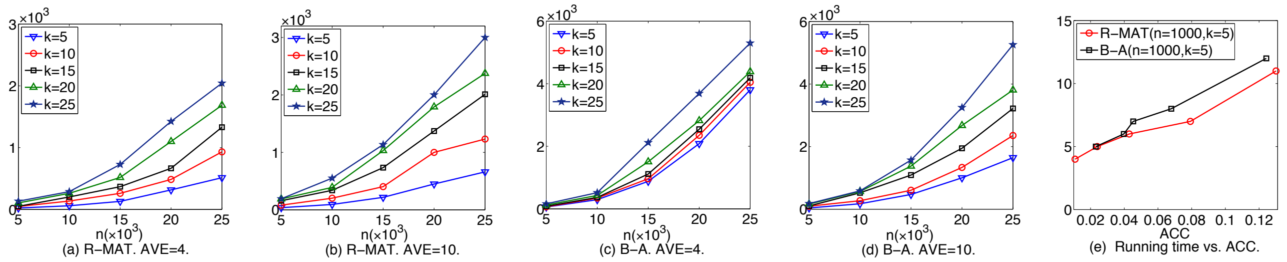


Fig. 7. Running time (s).

$k = 25$, our HIGA scheme will terminate in $491s$ for Facebook, but it requires $95h$ for Douban. Therefore, it is essential to improve the efficiency of the implementation for large-scale data sets with millions of nodes and edges. Our basic idea is to utilize the graph partition technique proposed in [19] to enable parallel graph anonymization [20]. Specifically, a large-scale graph will be divided into many partitions, each consisting of a set of nodes and the edges in those nodes' 1-neighborhood graphs. Assignment of a node to a partition depends solely on the node ID by computing a partitioning function $hash(ID) \mod N$, where N is the number of partitions. Then, we run our HIGA scheme on each partition in parallel and combine all the anonymized partitions by removing the duplicated added edges.

Given partition i, only the nodes satisfying $hash(ID) \mod N = i$ will be processed and thus, the running time will be largely reduced. If we apply N servers running in parallel, an anonymized Douban network can be generated within one hour when $k = 25$ and $N = 50$. From Fig. 8, we know that the running time decreases as N increases. However, the information loss increases as N increases. Therefore, two key problems needed to be considered: 1) how to determine an appropriate partition number N; 2) how to split the large graphs. As part of our future work, we will investigate how to make our scheme work well in parallel graph anonymization.

To show the effectiveness of our HIGA scheme, we compare it with existing k-neighbor anonymity and weight anonymization approaches. Specifically, we first compare our results with those of [2] in terms of the number of modified edges under the KDD cup 2003 co-authorship data set[6]. To fairly perform comparisons, we extract 120,640 edges from the data set, so that our experiment setting is the same as theirs. The work in [2] aims to generate a k-neighbor anonymous network where each node's 1-neighborhood graph is isomorphic to at least $k-1$ other nodes'. Unlike their scheme requiring the isomorphism of graphs, our scheme applies the concept of approximate matching to achieve probabilistic indistinguishability in the anonymized network and thus, the number of modified edges is relatively low. From Fig. 9-(a), we know that our scheme outperforms [2] while keeping the same level of privacy.

Then, we compare the APW with the work in [21] under the weighted network, NetSci[7], which consists of 1,589 nodes and 2,742 edges. The work in [21] aims to anonymize weight volumes and weight distribution without involving the anonymization of sub-

6. http://www.cs.cornell.edu/projects/kddcup/datasets.html

7. http://www-personal.umich.edu/∼mejn/netdata/

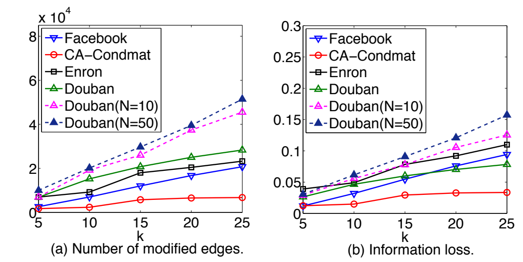


Fig. 8. Performance of HIGA in real data sets.

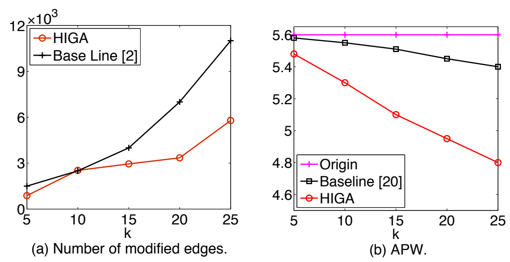


Fig. 9. Comparison with existing work.

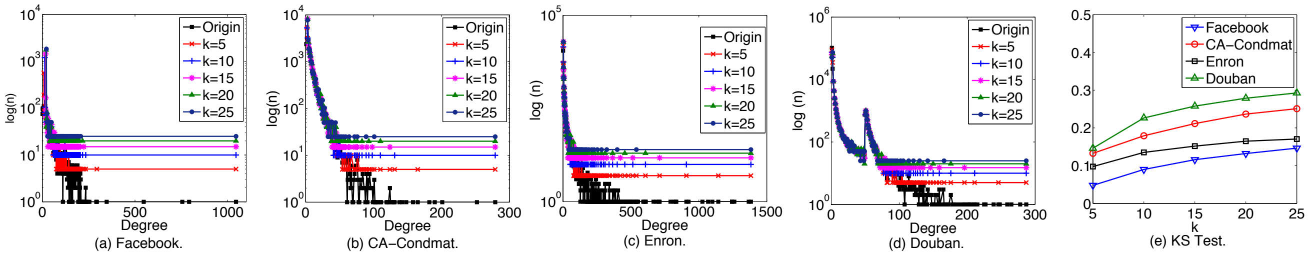


Fig. 10. Degree distribution.

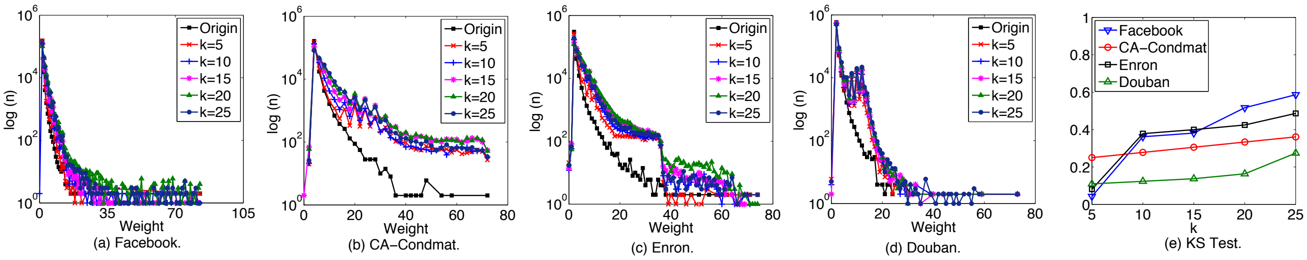


Fig. 11. Weight distribution.

graphs. That is, [21] does not require the structure similarity of a group of nodes and thus, the anonymized networks neither achieve k-neighbor anonymity nor have the property of probabilistic indistinguishability. Therefore, [21] has a high graph utility at the cost of privacy. Fig. 9-(b) shows that our HIGA scheme can preserve graph utility nearly as well as theirs.

8.4 Utility

Subfigures $(a) \sim (d)$ in Figs. 10 and 11 show that both the degree and the weight distributions follow the power-law distribution in the anonymized graphs. To quantify the comparison results, we utilize the Kolmogorov-Smirnov (KS) test to show the magnitude of difference between the original and anonymized distributions. From Fig. 10-(e) and Fig. 11-(e), we know that the difference between two distributions increases as the group size, k, increases. For example, given the significance level 0.01 for the weight distribution in Facebook, it fails to reject the null hypothesis with a probability of 95% when $k = 5$, but it rejects the null hypothesis as k increases to 10.

Table 3 shows the comparison results for answering aggregate queries. Similarly, the utility of the network decreases as k increases. However, the query results in all the real networks except Facebook are still very useful even when $k = 25$. From Table 3, we know that ACC will decrease as k increases. For the networks with a higher ACC, the drop rate of ACC is higher during anonymization. Furthermore, the APW and the APL decrease as k increases while anonymizing networks with a small number of nodes (e.g., Facebook); these values increase as k increases for large-scale networks (e.g., Enron and Douban).

9 Related Work

Our work is on publishing weighted social networks while preserving individuals' identity privacy. To defend the re-identification attacks, the work in [22] advocated the k-anonymity model, where every node should be indistinguishable with at least $k-1$ other nodes in terms of both the attributes and the associated structural information, such as neighborhood and node degree. With such a property, the attacker cannot re-identify any individual from the published networks with a confidence higher than $1/k$. To achieve k-anonymity, existing approaches can be classified as clustering-based approaches and graph modification-based approaches. Clustering-based approaches cluster nodes and edges into groups and anonymize subgraphs into super nodes, thus leading to great changes in the structure of the original graph.

To preserve the scale and the local structures of the original graph, the graph modification-based approaches try to locally modify the graph structure

TABLE 3
Utility of the anonymized social network

Facebook	**MIN**	**MAX**	**AVE**	**ACC**	**APW**	**APL**
Original	*1*	*1045*	*44*	*0.605*	*7.6*	*4.7*
k=5	8	1045	45	0.537	6.42	4.06
k=10	13	1045	47	0.508	6.39	3.98
k=15	17	1045	49	0.478	6.15	3.85
k=20	20	1045	52	0.468	6.03	3.81
k=25	22	1045	54	0.466	5.8	3.73
CA-Condmat	**MIN**	**MAX**	**AVE**	**ACC**	**APW**	**APL**
Original	*1*	*279*	*8*	*0.633*	*11.1*	*6.4*
k=5	2	279	8.2	0.577	11.38	6.38
k=10	2	279	8.3	0.534	11.86	6.42
k=15	3	279	8.5	0.412	12.51	6.49
k=20	3	279	8.6	0.505	12.68	6.58
k=25	3	279	8.7	0.499	13.28	6.64
Enron	**MIN**	**MAX**	**AVE**	**ACC**	**APW**	**APL**
Original	*1*	*1383*	*10*	*0.497*	*8.18*	*4.9*
k=5	2	1383	10.4	0.496	8.93	4.92
k=10	2	1383	10.6	0.4813	9.29	4.99
k=15	3	1383	11	0.476	9.63	5
k=20	3	1383	11.2	0.445	9.75	5.01
k=25	3	1383	11.3	0.423	10.05	5.05
Douban	**MIN**	**MAX**	**AVE**	**ACC**	**APW**	**APL**
Original	*1*	*287*	*4.2*	*0.016*	*9.66*	*5.7*
k=5	1	287	4.3	0.0156	11	5.86
k=10	1	287	4.4	0.0153	11.01	5.87
k=15	1	287	4.5	0.0152	11.03	5.87
k=20	1	287	4.5	0.0151	11.06	5.88
k=25	1	287	4.6	0.015	11.15	5.88

to achieve the privacy preservation requirement. For example, the work in [1] proposed the guarantee of k-anonymity on node degrees, so that for every node v, there are at least $k-1$ other nodes that have the same node degrees as v. The work in [2] provided a heuristic solution against the 1-neighborhood attack. The work in [3] anonymized the data graph by graph partition, block alignment, and edge copy so that the resulting graph is k-automorphic. The work in [15] quantified the privacy risks associated with different kinds of attacks on social networks, and proposed a randomization method to perturb the graph. The work in [23] identified two realistic targets of attacks, NodeInfo and LinkInfo, and proposed a solution to form k pairwise isomorphic subgraphs.

Existing work mainly focuses on unweighted social networks. Research in the area of anonymizing weighted graphs is still in its infancy. As a pioneering work, the work in [24] built a linear programming model on weighted social networks and reassigned edge weights to obtain an anonymized graph with the linear property. The work in [21] formalized a general model for weighted graph anonymization to achieve k-histogram anonymity. The work in [12] proposed a generation-based anonymization approach to generate a k-possible anonymity weighted social network. The work in [25] addressed the problem of outsourcing weighted social networks to the cloud while preserving shortest distances. The work in [26] proposed a clustering-based k-anonymization method to prevent identity disclosure in weighted social networks. The work in [13] presented a two-phase approach to generate a structure and text-aware k-anonymity social network. However, most of the existing solutions target k-anonymity. Our previous work [27] defined the probabilistic indistinguishability property for generating an anonymized social network with high utility. The main drawback is that the proposed scheme applies to only unweighted social networks. A weighted graph is a generalization of the unweighted graph. Therefore, our HIGA scheme is more practical.

10 CONCLUSION

In this paper, we investigate the problem of publishing weighted social network data in a privacy-preserving way. We identify a weighted 1*-neighborhood attack and define a key property, probabilistic indistinguishability, to resist this attack. Then, we propose a HIGA scheme to generate a probabilistically indistinguishable social network. The empirical study indicates that the anonymized social networks can be used to answer aggregate queries with high accuracy. In our future work, we will try to introduce other privacy mechanisms to our scheme, e.g., by combining with l-diversity and t-closeness, we will enable the nodes in a group to be associated with at least l different sensitive attributes, and the distribution of the sensitive attributes in a group will be close to the distribution of the attributes in the overall data.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants CNS 1449860, CNS 1461932, CNS 1460971, CNS 1439672, DUE 1431330, CNS 1301774, and ECCS 1231461; NSFC grants 61632009, 61272151, 61472451, 61402161; and the Hunan Provincial Natural Science Foundation of China (Grant No. 2015JJ3046).

REFERENCES

[1] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proc. of ACM SIGMOD*, 2008.

[2] B. Zhou and J. Pei, "The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks," *Knowledge and Information Systems*, 2011.

[3] L. Zou, L. Chen, and M. Özsu, "K-automorphism: A general framework for privacy preserving network publication," in *Proc. of the VLDB*, 2009.

[4] F. Bonchi, A. Gionis, and T. Tassa, "Identity obfuscation in graphs through the information theoretic lens," *Information Sciences*, 2014.

[5] Y. Boshmaf, I. Muslukhov, and et al., "Design and analysis of a social botnet," *Computer Networks*, 2013.

[6] M. Gori, M. Maggini, and et al., "Exact and approximate graph matching using random walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[7] L. Page, S. Brin, and et al., "The PageRank citation ranking: Bringing order to the web," Stanford InfoLab, Tech. Rep., 1999.

[8] E. Hazan, S. Safra, and et al., "On the complexity of approximating k-dimensional matching," in *Proc. of APPROX*, 2003.

[9] A. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, 1999.

[10] T. Kanungo, D. Mount, and et al., "An efficient k-means clustering algorithm: Analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

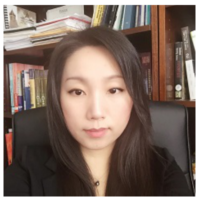
[11] K. R. Varadarajan and P. K. Agarwal, "Approximation algorithms for bipartite and non-bipartite matching in the plane." in *Proc. of SODA*, 1999.

[12] X. Liu and X. Yang, "A generalization based approach for anonymizing weighted social network graphs," in *Web-Age Information Management*, 2011.
[13] Y. Hao, H. Cao, and et al., "K-anonymity for social networks containing rich structural and textual information," *Social Network Analysis and Mining*, 2014.
[14] X. Ying and X. Wu, "Randomizing social networks: a spectrum preserving approach." in *Proc. of SIAM SDM*, 2008.
[15] M. Hay, G. Miklau, and et al., "Anonymizing social networks," Tech. Rep., 2007.
[16] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proc. of ACM SigKDD*, 2006.
[17] D. Chakrabarti, Y. Zhan, and C. Faloutsos, "R-MAT: A recursive model for graph mining." in *Proc. of SIAM SDM*, 2004.
[18] A. Sala, L. Cao, C. Wilson, R. Zablit, H. Zheng, and B. Y. Zhao, "Measurement-calibrated graph models for social network experiments," in *Proc. of WWW*, 2010.
[19] G. Malewicz, M. H. Austern, and et al., "Pregel: a system for large-scale graph processing," in *Proc. of ACM SIGMOD*, 2010.
[20] A. Lenharth, D. Nguyen, and K. Pingali, "Parallel graph analytics," *Communications of the ACM*, 2016.
[21] Y. Li and H. Shen, "Anonymizing graphs against weight-based attacks," in *Proc. of IEEE ICDMW*, 2010.
[22] A. Campan and T. Truta, "A clustering approach for data and structural anonymity in social networks," in *Proc. of PinKDD*, 2008.
[23] J. Cheng, A. Fu, and J. Liu, "K-isomorphism: privacy preserving network publication against structural attacks," in *Proc. of ACM COMAD*, 2010.
[24] S. Das, O. Egecioglu, and A. El Abbadi, "Anonymizing weighted social network graphs," in *Proc. of IEEE ICDE*, 2010.
[25] J. Gao, J. Yu, and et al., "Neighborhood-privacy protected shortest distance computing in cloud," in *Proc. of ACM COMAD*, 2011.
[26] M. E. Skarkala, M. Maragoudakis, and et al., "Privacy preservation by k-anonymization of weighted social networks," in *Proc. of IEEE ASONAM*, 2012.
[27] G. Wang, Q. Liu, and et al., "Outsourcing privacy-preserving social networks to a cloud," in *Proc. of IEEE INFOCOM*, 2013.

Qin Liu received her B.S. in Computer Science in 2004 from Hunan Normal University, China, received her M.S. in Computer Science in 2007, and received her Ph.D. in Computer Science in 2012 from Central South University, China. She has been a Visiting Student at Temple University, USA. Her research interests include security and privacy issues in cloud computing. She is an Assistant Professor in the College of Computer Science and Electronic Engineering at Hunan University, China.

Guojun Wang received his B.S. in Geophysics, in 1992, M.S. in Computer Science, in 1996, and Ph.D. in Computer Science, in 2002, from Central South University, China. He is the Pearl River Scholar Professor of Guangdong Province at School of Computer Science and Educational Software, Guangzhou University, China. He has been an Adjunct Professor at Temple University, USA; a Visiting Scholar at Florida Atlantic University, USA; a Visiting Researcher at the University of Aizu, Japan; and a Research Fellow at the Hong Kong Polytechnic University. His research interests include network and information security, Internet of things, and cloud computing. He is a distinguished member of CCF, and a member of IEEE, ACM, and IEICE.

Feng Li earned his B.E. (Computer Science and Technology, 2002) and M.S. (Computer Science, 2005) from Southeast University (Nanjing, China). He received his Ph.D. in Computer Science from Florida Atlantic University in Aug. 2009. His Ph.D. advisor is Prof. Jie Wu. He is an Associate Professor of the Department of Computer, Information, and Technology at Indiana University-Purdue University Indianapolis (IUPUI). Dr. Li teaches Wireless Network and Wireless courses in the department. Dr. Li has been actively engaged in research on computer networks, security and trust issues. He has published more than 20 papers in top conferences including INFOCOM and ICDCS. He welcomes any research/project collaboration on the above research topics.



Shuhui Yang is an assistant professor in the Department of Math, Computer Science, and Statistics at Purdue University, Calumet campus since 2009. She worked in the Department of Computer Science at Rensselear Polytechnic Institute as a postdoc research associate after she graduated in 2007. Her host advisor is Dr. Wei Zhao. She received her Ph.D. degree from the Department of Computer Science and Engineering, Florida Atlantic University, in August 2007. She obtained her M.S. degree from Department of Computer Science and Technology, Nanjing University, Nanjing, China in 2003 and B.S. degree from Jiangsu University, Zhengjiang, China in 2000, both in Computer Science. Her Ph.D. dissertation advisors were Dr. Jie Wu (currently with Temple University) and Dr. Mihaela Cardei. Her research interests include Wireless Networks and Mobile Computing, Parallel and Distributed Systems, and Wireless Security and Privacy.

Jie Wu is the Associate Vice Provost for International Affairs at Temple University. He also serves as Director of Center for Networked Computing and Laura H. Carnell professor in the Department of Computer and Information Sciences. Prior to joining Temple University, he was a program director at the National Science Foundation and was a distinguished professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Service Computing and the Journal of Parallel and Distributed Computing. Dr. Wu was general cochair/chair for IEEE MASS 2006, IEEE IPDPS 2008, IEEE ICDCS 2013, and ACM MobiHoc 2014, as well as program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.