Effective Social Graph Deanonymization Based on Graph Structure and Descriptive Information

HAO FU, University of Science and Technology of China
ASTON ZHANG, University of Illinois at Urbana-Champaign
XING XIE, Microsoft Research

The study of online social networks has attracted increasing interest. However, concerns are raised for the privacy risks of user data since they have been frequently shared among researchers, advertisers, and application developers. To solve this problem, a number of anonymization algorithms have been recently developed for protecting the privacy of social graphs. In this article, we proposed a graph node similarity measurement in consideration with both graph structure and descriptive information, and a deanonymization algorithm based on the measurement. Using the proposed algorithm, we evaluated the privacy risks of several typical anonymization algorithms on social graphs with thousands of nodes from Microsoft Academic Search, LiveJournal, and the Enron email dataset, and a social graph with millions of nodes from Tencent Weibo. Our results showed that the proposed algorithm was efficient and effective to deanonymize social graphs without any initial seed mappings. Based on the experiments, we also pointed out suggestions on how to better maintain the data utility while preserving privacy.

Categories and Subject Descriptors: H.2.7 [**Database Management**]: Database Administration—*Security, integrity, and protection*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms: Algorithms, Experimentation, Security

Additional Key Words and Phrases: Deanonymization, privacy protection, social network

ACM Reference Format:
Hao Fu, Aston Zhang, and Xing Xie. 2015. Effective social graph deanonymization based on graph structure and descriptive information. ACM Trans. Intell. Syst. Technol. 6, 4, Article 49 (July 2015), 29 pages.
DOI: http://dx.doi.org/10.1145/2700836

1. INTRODUCTION

In social networking sites or other online collaborative tools, a user is represented as her profile, which contains descriptive information such as her name, gender, and birth year. While users socialize with others via online services, different kinds of social ties are established among them. The social ties could be described by information such as type of relations (friend, family, etc.) and strength of relations (e.g., number of sent emails). User profiles and their social ties can be described as a *social graph*, which consists of both graph structure and descriptive information of user profiles and social ties.

Social networks have attracted lots of interest among researchers, advertisers, and application developers. In order to satisfy the need for analysis, operators of social networking services are increasingly sharing information that could potentially breach

Authors' addresses: H. Fu (corresponding author), School of Computer Science and Technology, University of Science and Technology of China; email: fuch@mail.ustc.edu.cn; A. Zhang, Department of Computer Science, University of Illinois at Urbana-Champaign; X. Xie, Microsoft Research, Beijing, China.

DOI: http://dx.doi.org/10.1145/2700836

users' privacy. A large portion of research on social network privacy has concentrated on *identity disclosure*, which reveals nodes' identities in the real world. To preserve privacy, the data need to be *anonymized* before publishing. A straightforward approach, namely, the naive anonymization, removes personal identifiable information such as names and leaves the graph structure as it was.

Backstrom et al. [2007] showed how an adversary can create a small number of accounts and establish highly distinguishable patterns of links among the created accounts before data publishing, and then search for those patterns and linked target accounts efficiently in the anonymized graph, namely, an *active attack*. On the contrary, a *passive attack* assumes an adversary cannot create any new nodes or edges. Instead, she collects external auxiliary information, for example, nodes of interest and partial edges among them, and then identifies target nodes with the aid of such information. Hay et al. [2008] investigated the risk of naive anonymization and found that a significant number of nodes can be uniquely identified using only the node degrees of their neighbors. In this article, we focus our discussions on identity disclosure through passive attacks.

In recent years, several studies on the potential privacy risks of social graph data have been conducted, and a number of anonymization algorithms have been proposed to protect privacy against different types of threats. A typical threat could be, for example, the leaking of node degrees, neighborhoods of breached nodes, or subgraphs of arbitrary sizes around certain nodes.

Anonymization is performed by modifying the structure and descriptive information of social graphs. Modification increases the difficulty of attacks but hurts the utility of data at the same time. In order to satisfy a higher level of privacy requirements, it is sometimes impossible to maintain sufficient utility. This would make the resulting anonymized graphs useless for analysis. In practice, while naive anonymization is still commonly adopted for good utility, it is unclear how data owners would apply the proposed anonymization algorithms that make minor modifications for better privacy preservation. In this sense, it is necessary to evaluate the deanonymization risk of the algorithms in a practical scenario.

Besides, sometimes it is difficult to obtain auxiliary knowledge from the original graph; for example, the original graph could be highly secured. As the profiles and relationships in a social networking service are usually corelated with those in the real world, knowledge from other related social graphs could be leveraged for attacks. In this case, recognizing users in an immediate manner remains unclear, since the number of overlapping users is relatively small, and their neighborhood structures may be different from each other. Narayanan and Shmatikov [2009] illustrated that about one-third of the users that were presented both in Twitter and Flickr can be mapped correctly, though the overlap is thought to be only 15%. However, their approach requires a certain number of *seed mappings* (the location of users in both graphs) to trigger the deanonymization.

In addition, while social graphs are always evolving over time [Barabâsi et al. 2002], the descriptive information is also changing. According a recent report,[1] about two-thirds of the users access social networking sites daily, and one-third of the users who ever updated their status do so nearly every day or more frequently. Therefore, deanonymizing a graph that was published a long time ago can be difficult, even if only naive anonymization is applied. We regard such evolution as a kind of "unintended" anonymization. For example, the Tencent Weibo dataset, which was published in KDD Cup 2012,[2] was taken from a snapshot of the website and naively anonymized. The dataset description did not specify when the snapshot was taken, but we learned that it

[1]http://socialhabit.com.
[2]http://www.kddcup2012.org/c/kddcup2012-track1.

might be sometime before October 12, 2011, according to the time stamps contained in the dataset. Ideally, an adversary can crawl the original social graph on Tencent Weibo website and match that with the public dataset. However, during the time before data release (February, 2012), it was unclear to what extent the graph has evolved, rendering such a matching nontrival. In addition, according to another recent report,[3] Tencent Weibo has over 540 million users by the end of 2012. That means, although the Tencent Weibo data is public, an adversary still has very limited knowledge for carrying out attacks due to the high crawling and computational costs. Actually, we have crawled 20 million user profiles during November and December in 2012, but only found a few dozen uniquely matched profiles. After manually checking, we found these profiles are not contained in the public dataset. That means, the auxiliary information could be partial and noisy, even if it is obtained from the same source as the target.

To the best of our knowledge, there are two major approaches toward the problem of deanonymizing social graphs. In the first approach, deanonymization is considered as a seeded graph matching problem. The matching starts with a small number of seed mappings. Such mappings are propagated recursively along edges to match other nodes [Narayanan and Shmatikov 2009; Yartseva and Grossglauser 2013]. This approach is usually a good choice for deanonymization when high quality seed mappings are available, and it has been successfully applied to many real-world deanonymization problems [Narayanan and Shmatikov 2009; Narayanan et al. 2011]. However, the success of attacks highly depends on the number and quality of seed mappings [Narayanan and Shmatikov 2009] (Section 6.2.5). Our main concern lies on the feasibility and quality of seed mappings.

First, in a practical attack, it could be unclear how the seed mappings are collected. A common method for seed selection is detecting cliques or nontrivial subgraphs. It has been shown that this method is not guaranteed to always work in more general scenarios [Zhang et al. 2014]. Narayanan et al. [2011] also suggested that it was unclear whether the aforementioned method is feasible, provided that how much the social graph had evolved over time was unknown. They proposed a simulated annealing algorithm for seed identification, but the success of it appeared to rely on the rough but obvious correspondence between certain kinds of nodes (e.g., high in-degree). We argue that such a correspondence does not always exist, and we have shown that it is sensitive to the type of graphs and applied anonymization algorithms (Section 6.2.5). Second, there is no guarantee on the number or quality of seed mappings. It has been shown that there is not a one-size-fits-all method for seed identification, and the outcome is sensitive to the size and the type of social graphs [Gulyas and Imre 2014]. Gulyas and Imre also showed that the chosen seed selection method can significantly influence and limit the possible outcome of the propagation. Significant differences emerged even in the same or in structurally divergent networks. Due to the preceding concerns, we believe that avoiding the reliance on seed mappings would reduce the effort of conducting an effective deanonymization attack and allow the deanonymization attack to be successful in more general scenarios.

The second approach does not require any seed mappings and is based on particular node signatures. Early works of this approach [Backstrom et al. 2007; Hay et al. 2008] require an exact matching of signatures, for example, node degrees and subgraphs. When the graph structure is modified by anonymization, those methods would fail. On the other hand, methods based on structural features [Henderson et al. 2011], descriptive information [Korayem and Crandall 2013], or node similarities [Jeh and Widom 2002; Blondel et al. 2004] are robust to structural modifications but suffer from the problem of low accuracy. In order to overcome the difficulty in collecting sufficient seed mappings of high quality, we explored the feasibility of avoiding seed mappings

[3]http://www.techinasia.com/tencentweibo-registered-users-540-million.

in our previous work [Fu et al. 2014]. It turns out that our approach is robust and effective for social graphs of different sizes and types. As a result, we believe our choice of approach is proper to study the deanonymization risk of social graphs. A detailed discussion of existing works is presented in Section 2.

In this article, we focus on our previous approach and conduct a detailed study on the problem of deanonymization with graphs that are noisy or anonymized by a certain algorithm. As descriptive information such as user profiles is usually available in a practical attack, we are also interested in incorporating such information with the graph structure to boost deanonymization.

In summary, we made the following contributions in this article:

—We proposed a graph node similarity measurement and a deanonymization algorithm in consideration with both graph structure (Section 4) and descriptive information (Section 5) without the use of seed mappings. We must emphasize that it was nontrivial to seek for such a simple but effective algorithm. Actually, we have won the deanonymization track of WSDM 2013 Data Challenge[4,5] with the proposed algorithm.
—Using the proposed algorithm, we evaluated the privacy risks of several typical anonymization algorithms (Section 6) on four real datasets, a coauthor graph from Microsoft Academic Search (8,248 nodes), a friendship graph from LiveJournal (10,000 nodes), a communication graph from the Enron email dataset (8,678 nodes), and a social graph from Tencent Weibo (2.3 million nodes).
—We found that a node is always the most similar to itself in the sense of the proposed similarity measurement, and the maximum similarity score is actually the graph node's eigenvector centrality in certain situations (Section 4.2). Experimental results show that important nodes (in terms of eigenvector centrality) are more likely to be reidentified (Section 6.2).
—We found that combining descriptive information and graph structure boosts the accuracy of deanonymization significantly. It was also revealed that descriptive information, when presented, plays a more important role than graph structure in deanonymization, even if the information is inaccurate (Section 6.3).

To the best of our knowledge, no previous work has explored the preceding problems to such extent. We believe these discoveries are of great importance, since new concerns to protect the privacy in graph data are raised based on them, and they may motivate future works in this line of research.

2. RELATED WORK

2.1. Anonymization

In a survey by Wu et al. [2010c], existing anonymization methods that modify the graph structure are categorized as k-anonymity, randomization, and clustering.

The concept of k-anonymity refers to the property that a node is indistinguishable from at least $k - 1$ other nodes in terms of certain types of patterns. Liu and Terzi [2008] suggested that an adversary can breach privacy by collecting and matching node degrees, since node degrees in a real-world graph are usually skewed. They illustrated how to achieve k-degree anonymity through a sequence of edge addition (or deletion). Tai et al. [2011] proposed the k^2-degree anonymity to resist the friendship attack, where an adversary utilizes the degrees of two connected nodes. Zhou and Pei [2008] assumed an adversary knows one specific subgraph induced by the immediate neighbors of a

[4]http://www.wsdm2013.org/index.php/authors/data-challenge.
[5]The data challenge report (4 pages) was not published.

target node. They proposed an algorithm that achieves k-neighborhood anonymity by generalizing node labels and adding edges. As a stronger type of k-anonymity, Zou et al. [2009] proposed the k-automorphism anonymity that protects privacy even if the adversary knows any subgraph around a certain node. Wu et al. [2010a] proposed a similar approach named k-symmetry. Cheng et al. [2010] proposed the k-isomorphism algorithm that guarantees that, in addition to the node k-anonymity, the adversary cannot even infer the existence of an edge between two nodes. These methods can be seen as some kind of "ultimate" solutions, since they guarantee resistance against any kind of structural attack. However, they seem to overkill since too many modifications are required. For example, the k-automorphism needs to add about 70% new edges to a coauthor graph in database and theory conferences, according to the results in Zou et al. [2009].

Randomization protects privacy in a probabilistic manner and does not guarantee any level of anonymity. Bonchi et al. [2011] described the sparsification and the perturbation approaches. The sparsification approach randomly deletes edges, and the perturbation approach randomly deletes and adds the same number of edges. The switching approach randomly switches a specified number of pairs of existing edges, thus the degree of each node is preserved. Ying and Wu [2008] suggested that a graph's spectrum is related to several important graph metrics, and proposed a randomization algorithm that can preserve a graph's spectrum. Beyond randomizing the graph structure, Wu et al. [2010b] considered a low-rank approximation approach to reconstruct a graph from the anonymized one to obtain more accurate feature values. Several works focused on the problem of link privacy in the randomization approach. Ying and Wu [2009] considered an algorithm that is expected to better preserve prescribed features by following given constraints in the process of switching edges. The link privacy risks of the algorithm were also analyzed. An edge randomization algorithm based on random walks was proposed by Mittal et al. [2013] to avoid *link disclosure*.

The last category works in a very different manner compared with k-anonymity and randomization. Clustering methods group nodes and edges into partitions, and usually generalize attributes. Since only the size and density of the partitions are published, an adversary is unable to distinguish any individual in a partition [Zheleva and Getoor 2008; Cormode et al. 2008; Campan and Truta 2009; Tassa and Cohen 2013]. In this article, we are interested in the first two types of approaches.

In recent years, the notion of differential privacy [Dwork 2006] has attracted much attention. Differential privacy provides a certain privacy guarantee of the data release mechanism in a statistical database. For example, the presence or absence of any individual item in a dataset would not affect the output of a differentially private algorithm significantly. A number of recent studies [Hay et al. 2010; Sala et al. 2011; Karwa et al. 2011] focus on the differential privacy in graphs. The problem of attacking those methods is essentially different from what we are considering in this article, since they are mainly designed to enable specific kinds of analyses by providing statistical information or synthetic graphs.

2.2. Deanonymization

Backstrom et al. [2007] proposed a family of deanonymization methods, including the active attack and the passive attack, which make it possible for an adversary to learn whether edges exist or not between two specified nodes. However, the methods only work for naive anonymization and is infeasible once the graph is modified before publishing.

Narayanan and Shmatikov [2009] proposed a deanonymization algorithm that reidentifies nodes using only structural information of a different social graph. The algorithm starts with a few seed mappings and propagates the mappings through edges. As

the previous mappings might be wrong, it then propagates back to correct those mappings. The algorithm was shown by experiments to be effective in deanonymization. However, the authors suggested that a successful attack relies heavily on the quality of seed mappings. Narayanan et al. [2011] later suggested that it was unclear whether the clique search technique is feasible, provided that how much the social graph had evolved over time was unknown. They then proposed an algorithm for seed identification based on simulated annealing. The success of the proposed algorithm appeared to rely on the rough but obvious correspondence between certain kinds of nodes (e.g., high in-degree).

Even when collecting seed mappings is feasible in an attack, there is no guarantee on the number or the quality of the mappings. For example, the simulated annealing algorithm proposed in Narayanan et al. [2011] succeeded in providing high quality seed mappings in a naively anonymized Flickr graph, but our experimental results (Section 6.2.5) showed that its accuracy was unsatisfactory when the graph was modified by a particular anonymization algorithm. Choosing the proper method for seed identification could also be problematic. Gulyas and Imre [2014] evaluated multiple seed identification methods on various types of social graphs. Their results showed that the choice of seed identification method depended on the size and type of the graph, and it could significantly influence and limit the possible outcome of the propagation. In this article, we intend to study the privacy risks of several typical anonymization algorithms, so developing a robust deanonymization algorithm without seed mappings is more appropriate.

A number of works on privacy risks related to *attribute disclosure* were also conducted. Mislove et al. [2010] showed that certain types of attributes are highly related to the graph's community structure, and a user's descriptive information can thus be inferred. Gundecha et al. [2011] investigated the privacy risks about vulnerable friends by introducing a set of privacy measurements based attributes. Although we focus on the problem of identity disclosure instead of attribute disclosure, privacy issues about attributes were also studied. In addition, several works attempting to model the privacy risk in social graphs were also conducted [Pedarsani and Grossglauser 2011; Zhang et al. 2014].

2.3. Node Similarity

Several recent works investigate the privacy risks in social graphs by exploiting structure similarities. Henderson et al. [2011] proposed a family of structural features by aggregating features from neighboring nodes recursively. Deanonymization of nodes is achieved by comparing the Euclidean distance of feature vectors. However, their method appears only able to reduce a certain fraction of guesses required to find the real identity. The link prediction problem focuses on the evolution of social graphs and is to predict the existence of a link in the future. Several proximity measures were shown to be effective for the link prediction problem [Liben-Nowell and Kleinberg 2003]. Ying and Wu [2011] illustrated that the proximity measures could also be exploited to predict the existence of sensitive links.

There are a number of works on graph node similarity for purposes other than deanonymization in literature. Blondel et al. [2004] proposed a graph node similarity measurement that is calculated iteratively by summing up the similarity scores between the neighbors of two nodes. The "SimRank" proposed by Jeh and Widom [2002] compares nodes in the same graph, so it is infeasible for deanonymization (in which two graphs are compared). Melink et al. [2002] proposed a general framework of graph node similarity named "Similarity Flooding," which takes graph structure, node, and edge attributes into consideration, and can be viewed as a generalization of the preceding two measurements. In our initial experiments, we evaluated the preceding

measurements, and found empirically that they are not suitable for deanonymization (Section 4.2), so we decided to find a new node similarity measurement that is effective in deanonymization.

3. PRELIMINARIES

Social Graphs. Two models for social networks are described in Wu et al. [2010c]. A *simple graph* is an undirected graph $G = (V, E)$ without any descriptive information. Nodes in V correspond to users, and an edge $(i, j) \in E$ indicates there is a social tie between users i and j.

A *rich graph* is the combination of a directed or undirected graph, and two attribute sets X and Y. User i's descriptive information (or *node attribute*) is denoted as $X(i)$, and the description of a social tie (i, j) (or *edge attribute*) is represented as $Y(i, j)$. For example, if a user's profile contains fields such as name, gender, and birth year, the corresponding $X(i)$ may look like {Alice, female, 1990}. If user i marks user j as a "friend" and has sent the friend five messages, the edge attributes $Y(i, j)$ can be represented as {friend, 5}.

Data Publishing. We refer to the published social graph as the *target graph*. In order to preserve privacy, social graphs are anonymized before publishing. The anonymization usually involves modification of a graph's structure and attributes, and can be summarized in two steps:

(1) Modify the original social graph with a certain anonymization algorithm, for example, algorithms that achieve some kind of k-anonymity, or randomization.
(2) Assign new random identifiers to nodes.

Threat Model. It is commonly assumed that an adversary knows some kind of prior knowledge about the target nodes. The adversary can then use the knowledge to locate the target nodes in the anonymized social graph. We assume that an adversary can always collect a subgraph around target nodes. The collected graph, in which the real identities of nodes are known, is referred to as the *auxiliary graph*. This assumption is practical because online social networking sites are usually partially or fully accessible. Note that this is the only prior knowledge: The adversary does not know the real identity, or seed mapping, of any node in the target graph.

Problem Formulation. Given an auxiliary graph $G_1 = (V_1, E_1)$ and a target graph $G_2 = (V_2, E_2)$, the goal of deanonymization is to find *identity disclosures* in the form of 1-1 mappings as many and accurate as possible. An identity disclosure (i, j) indicates that the two nodes $i \in V_1$ and $j \in V_2$ actually correspond to the same user.

4. DEANONYMIZING SIMPLE GRAPHS

Most anonymization algorithms for simple graphs achieve their anonymization goals by adding and/or deleting edges. For algorithms where either edge addition or deletion is involved, the anonymized or the original graph is subisomorphic to the other. In theory, the graph can be perfectly deanonymized by solving the subgraph isomorphism problem. However, as the subgraph isomorphism problem is known to be NP-complete [Cook 1971], this approach is infeasible for large-scale graphs. Once both edge addition and deletion are involved, the problem becomes complicated and the subisomorphism no longer holds. The basic assumption in this article is that the published graph is "similar" to the original graph, otherwise the data utility will be low. Motivated by this assumption, we propose a measurement for node similarity with respect to the graph structure. We then introduce a heuristic to produce node mappings as the final output for deanonymization. Finally, we discuss the relation between node similarity

and privacy risks. Different from Narayanan and Shmatikov [2009], no initial seed mapping is required in our approach.

4.1. Node Similarity

Suppose we are trying to compare graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Although swapping the two graphs will not change the output, we still prefer to denote the auxiliary graph as G_1 and the target graph as G_2. For nodes $i \in V_1$ and $j \in V_2$, we introduce the *node similarity score* $\mathcal{S}(i, j)$ as a structural measurement of how similar the two nodes are. In order to make the measurement effective for deanonymization, the following properties are required:

(1) If a graph is compared to itself, every node should be the most similar to itself. This property is naturally required by most similarity measurements but lacked by the measurements mentioned in Section 2.
(2) Two nodes are as similar as their neighbors are. This intuitive property is commonly assumed by all the node similarity measures we investigated earlier.

Based on the preceding requirements, we propose our definition of the node similarity. Denoting i's neighbor nodes as $N_1(i)$ and j's neighbor nodes as $N_2(j)$, we try to match similar nodes in $N_1(i)$ and $N_2(j)$ in an effort to maximize the overall similarity score. The similarity score $\mathcal{S}(i, j)$ is then assigned with such an overall score.

The idea of matching neighbors follows the following intuition. We first assume the node similarity scores between $N_1(i)$ and $N_2(j)$, that is, $\{\mathcal{S}(i', j')|(i', j') \in N_1(i) \times N_2(j)\}$, are precalculated. If i and j indeed correspond to the same identity, their neighbors $N_1(i)$ and $N_2(j)$ should "fit" to each other in the sense of a pairwise correspondence. In other words, we should be able to find a unique corresponding node j' in $N_2(j)$ for every node i' in $N_1(i)$ or vice versa, so that i' and j' correspond to the same identity. We quantify the "fitness" of $N_1(i)$ and $N_2(j)$ as the maximum sum of similarity scores between matched nodes, and take it as the similarity score $\mathcal{S}(i, j)$. A high value of $\mathcal{S}(i, j)$ indicates a possible correspondence between $N_1(i)$ and $N_2(j)$, while a low value suggests that the neighbors may be not similar.

According to the preceding definition, $\mathcal{S}(i, j)$ depends on the similarity scores between $N_1(i)$ and $N_2(j)$, but the latter also depends on the former. We handle the recursive definition in a recursive way: We update the similarity scores iteratively until a fixed point is reached (see Algorithm 1 for pseudocode). Because the similarity score is calculated by matching neighbors, we refer to the deanonymization algorithm as `NeighborMatch`.

The `NeighborMatch` algorithm is iterative and updates $\mathcal{S}(i, j)$ in each iteration. First, the initial values are taken as $\mathcal{S}^{(0)}(i, j) = 1$. In the k^{th} iteration, the algorithm starts matching i's neighbor nodes $N_1(i)$ and j's neighbor nodes $N_2(j)$ by constructing a complete bipartite graph $B_{i,j}^{(k+1)} = (N_1(i), N_2(j), N_1(i) \times N_2(j))$, where each edge (i', j') is weighted as $\mathcal{S}^{(k)}(i', j')$. The `MaxMatch` procedure is then invoked to find the maximum weighted match of $B_{i,j}^{(k+1)}$. We denote the set of matched edges as $\Theta_{i,j}^{(k+1)}$. An edge (i', j') in $\Theta_{i,j}^{(k+1)}$ indicates that $i' \in N_1(i)$ is matched to $j' \in N_2(j)$. Finally, $\mathcal{S}^{(k+1)}(i, j)$ is calculated as the sum of matched neighbors' similarity scores:

$$\mathcal{S}^{(k+1)}(i, j) = \sum_{(i', j') \in \Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i', j') \tag{1}$$

The optimal match $\Theta_{i,j}^{(k+1)}$ is recalculated in every iteration based on the node similarity scores $\mathcal{S}^{(k)}$. The preceding procedure is repeated until the normalized scores converge. The normalization is done by dividing $\mathcal{S}^{(k)}$ by the maximum $\mathcal{S}^{(k)}(i, j)$. The convergence

ALGORITHM 1: The `NeighborMatch` algorithm

Input : Auxiliary graph G_1, target graph G_2, number of expected mappings M
Output: Identity disclosures in the form of 1-1 mappings

foreach $(i, j) \in V_1 \times V_2$ **do**
  $\mathcal{S}^{(0)}(i, j) \leftarrow 1$;
$k \leftarrow 1$;
repeat
  foreach $(i, j) \in V_1 \times V_2$ **do**
    $B_{i,j}^{(k+1)} \leftarrow (N_1(i), N_2(j), N_1(i) \times N_2(j))$ where $w(i', j') = \mathcal{S}^{(k)}(i', j')$;
    $\Theta_{i,j}^{(k+1)} \leftarrow \texttt{MaxMatch}(B_{i,j}^{(k+1)})$;
    $\mathcal{S}^{(k+1)}(i, j) \leftarrow \sum_{(i', j') \in \Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i', j')$;
  $k \leftarrow k + 1$;
until normalized $\mathcal{S}^{(k)}$ converges;
$B \leftarrow (V_1, V_2, V_1 \times V_2)$ where $w(i, j) = \mathcal{S}^{(k)}(i, j)$;
$\Theta \leftarrow \texttt{MaxMatch}(B)$;
return (top M mappings in Θ with largest scores);

of Algorithm 1 is analyzed in Appendix A. As the diameter of a social graph is usually small, we found it unnecessary to repeat the iteration for many times, so we set the number of iterations to five in experiments.

We now introduce our approach to obtain identity disclosures. We regard the identity disclosures in the form of 1-1 mappings. Basically, we try to match V_1 and V_2 by maximizing the overall similarity score. Specifically, a complete bipartite graph $B = (V_1, V_2, V_1 \times V_2)$ is constructed by weighting edge $(i, j) \in V_1 \times V_2$ with similarity score $\mathcal{S}(i, j)$. The match Θ that reveals the real identities of users is taken as the maximum weighted matching of B. The algorithm outputs the top M node mappings in Θ in the sense of node similarity score, since node pairs with a higher similarity score are more likely to be correct.

In addition, methods to produce identity disclosure are not limited to the proposed approach. For example, a ranking of candidates for each node can be produced by sorting the candidates' similarity scores. The adversary can later check top candidates manually by comparing the profiles with domain knowledge.

To the best of our knowledge, the maximum weighted bipartite matching problem can be solved by the Hungarian algorithm in $O(n^3)$ time [Kuhn 1955]. The overall running time of a single iteration is then $O(|V_1||V_2|d^3)$, where d is the upper bound of node degrees. The running time can be reduced to $O(|V_1||V_2|d^2 \log d)$ by utilizing a greedy approximation algorithm instead: the edges of the bipartite graph are added to the matching one by one in descending order of weight. We found empirically that this approximation works better than expected in our algorithm.

4.2. Privacy Risk

We conduct a study on the relation between privacy risks and the proposed node similarity. For the sake of simplicity, we assume that G_2 has the same labeling as G_1 for nodes that appear in both graphs, that is, node $i \in V_1$ corresponds to $i \in V_2$. We start with the case where G_1 is a subgraph of G_2. We define the similarity score between node i and its real image as the *self-similarity score* $\mathcal{T}(i) = \mathcal{S}(i, i)$. We study the property of $\mathcal{T}(i)$ and find

THEOREM 4.1. *A node is always among the most similar candidates to itself, that is,* $\mathcal{T}^{(k)}(i) \geq \mathcal{S}^{(k)}(i, j)$ *for any nodes* $i \in V_1$, $j \in V_2$.

PROOF. Theorem 4.1 is proved by induction. For the base case of induction, we need to prove $\mathcal{T}^{(k)}(i) \geq \mathcal{S}^{(k)}(i, j)$ for $k = 0$. As $\mathcal{S}^{(0)}(i, j)$ equals 1, the claim holds in this case. For $k \geq 0$, we assume the claim is true for k. Consider the edge weight of bipartite graph $B_{i,j}^{(k+1)}$ in the $(k+1)^{\text{th}}$ iteration. The assumption indicates edge (i', i') has the maximum weight among $\{(i', j')|j' \in N_2(j)\}$, so one of the optimal matches for $\mathcal{T}^{(k)}(i)$ is $\Theta_{i,i}^{(k+1)} = \{(i', i')|i' \in N_1(i)\}$. As all optimal matches have the same overall scores, the updated self-similarity score is

$$\begin{aligned} \mathcal{T}^{(k+1)}(i) &= \sum_{i' \in N_1(i)} \mathcal{S}^{(k)}(i', i') \\ &= \sum_{i' \in N_1(i)} \mathcal{T}^{(k)}(i'). \end{aligned} \tag{2}$$

Based on the fact that only a subset of $N_1(i)$ is matched with regard to $N_2(j)$, we can then derive

$$\begin{aligned} \mathcal{S}^{(k+1)}(i, j) &= \sum_{(i', j') \in \Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i', j') \\ &\leq \sum_{(i', j') \in \Theta_{i,j}^{(k+1)}} \mathcal{T}^{(k)}(i') \\ &\leq \sum_{i' \in N_1(i)} \mathcal{T}^{(k)}(i') \\ &= \mathcal{T}^{(k+1)}(i). \quad \square \end{aligned}$$

Theorem 4.1 shows that $\mathcal{S}(i, j)$ is bounded by $\mathcal{T}(i)$. The following theorem further explains what $\mathcal{T}$ really is.

THEOREM 4.2. *The self-similarity score $\mathcal{T}$ is the principal eigenvector of G_1's adjacency matrix.*

PROOF. (Sketch) The updating rule of self-similarity scores in Equation (2) can be rearranged in the matrix form $\mathcal{T}^{(k+1)} = A_1 \mathcal{T}^{(k)}$, where A_1 is the adjacency matrix of G_1. It is identical to the power iteration algorithm that solves the principal eigenvector of a matrix, except for the scaling factor for normalization, so normalized $\mathcal{T}^{(k)}$ converges to A_1's principal eigenvector. $\square$

We now proceed to the general case where G_1 and G_2 have arbitrary overlap. The auxiliary graph G_1 is regarded as the combination of a subgraph $G_1' = (V_1 \cap V_2, E_1 \cap E_2)$ of G_2 and additional noise $N = (V_1 - V_2, E_1 - E_2)$. The noise N is caused by (1) the modifications that are made by anonymization algorithms and (2) extra nodes and edges that are involved during the process of collecting auxiliary graph. With the existence of the noise N, Theorems 4.1 and 4.2 no longer hold, and it is possible that $\mathcal{T}^{(k)}(i) < \mathcal{S}^{(k)}(i, j)$ for some $i \neq j$. In addition, we find empirically that this usually happens when i's and j's corresponding values in the principal eigenvector are close to each other. In this case, we are unable to identify the real image of node i only by examining the similarity scores, so we perform a bipartite matching between V_1 and V_2 to maximize the overall similarity score, since each node is supposed to have at most one image. However, this method could still not distinguish them if there are reasonably many such nodes. The preceding discussions motivate a possible anonymization approach against our attack that modifies the graph structure so that the value of every node is close to (e.g., less than a threshold) a sufficient number of other values in the principal eigenvector.

Recall the properties required by deanonymization in Section 4.1. The first property is proven to be satisfied directly by Theorem 4.1, and apparently the second property is also satisfied. For the node similarity measurements mentioned in Section 2, the first property is not guaranteed to be satisfied, since one can easily construct graphs where some nodes are less similar to themselves than the other nodes. As a consequence, they failed to provide reasonably accurate results for deanonymization. For example, the node similarity proposed by Blondel et al. [2004] can only identify less than 2% nodes in a coauthor graph from Microsoft Academic Search, once the graph is modified by any of the anonymization algorithms listed in Section 6.1.2. Only when the graph structure is not modified at all (naive anonymization), the measurement could identify about 45% nodes. We also obtained a similar result for "Similarity Flooding" [Melnik et al. 2002].

5. DEANONYMIZING RICH GRAPHS

In order to incorporate graph structure and attributes for deanonymization, we extend the algorithm described in Section 4 to process directed or undirected rich graphs. We start by introducing a framework of attribute similarity measurement, and then modify the updating rule of node similarity score to utilize attribute information.

5.1. Attribute Similarity

The *node-attribute similarity* $\mathcal{S}_X(i, j)$ represents the similarity between node-attribute sets $X(i)$ and $X(j)$. Analogously, the *edge-attribute similarity* $\mathcal{S}_Y(i_1, j_1, i_2, j_2)$ measures how similar edge-attribute sets $Y(i_1, j_1)$ and $Y(i_2, j_2)$ are. We assume that both measurements range from 0 to 1 inclusively and two attribute sets are more similar if the corresponding similarity score is larger, for example, value 0 indicates completely different and 1 suggests possible equivalence.

In a directed graph, there could be two edges of opposite directions between two specific nodes. In order to combine the information of the two edges, we introduce the concept of *relation*, which is an ordered nodes pair. Relation (i_1, j_1) being similar to (i_2, j_2) does not necessarily mean that it is similar to (j_2, i_2). We then propose the *relation similarity* $\mathcal{S}_R(i_1, j_1, i_2, j_2)$, which measures the similarity of relations (i_1, j_1) and (i_2, j_2) in conjunction with edges, for example, as the average of edge-attribute similarities $\mathcal{S}_Y(i_1, j_1, i_2, j_2)$ and $\mathcal{S}_Y(j_1, i_1, j_2, i_2)$. Again, determining the exact definition of relation similarity is nontrivial, so we only assume that it ranges from 0 to 1 and relies on edge-attribute similarity.

Generally, determining the exact definition of similarity measurement can be considered as a binary classification problem. Given a pair of nodes (or edges), the classifier is required to infer whether they correspond to the same identity. In this sense, a supervised learning approach can be adopted to incorporate all types of attributes for inference [Korayem and Crandall 2013]. When the ground truth for training is infeasible to collect, the adversary may need some heuristics based on the comprehension and domain knowledge of the attributes. Herein, we discuss general ideas of the heuristics based on our initial experiments. A concrete example is presented in Section 6.3.1.

For a binary or categorical attribute, it is usually fine to define the similarity as 1 for equal values, or 0 for inequality. For example, suppose the adversary has learned that a target user lives in Beijing. She may define the attribute similarity as 1 if the other user also lives in Beijing, or 0 if the latter lives in other cities. In cases where the location is less specific, for example, the other user is only known to live in China, an intermediate value could be assigned. For a numeric attribute, the similarity can be determined based on the distribution of its value. If the adversary knows that a user was born in 1980s, she may regard only the users who were also born in 1980s as candidates. In cases where little is known about the distribution, we find that

measuring the relative difference ($r(x, y) = \min\{x, y\}/\max\{x, y\}$) is usually a good choice for nonnegative values.

For a given pair of sets that contain multiple attributes, we tried several choices to combine the similarity of individual pairs of attributes. It turns out taking the average yields the best performance. For the relation similarity, taking the average of edge-attribute similarities also seems to be a good choice.

Despite the preceding intuition, we must emphasize that a good attribute similarity measurement depends on the domain knowledge of the attributes. More sophisticated methods [Henderson et al. 2011] are surely preferred for deanonymization.

Basically, we try to utilize the attribute information of graphs by introducing the preceding similarity measurements. $\mathcal{S}_X$ and $\mathcal{S}_Y$ are supposed to measure the similarity of nodes and edges with respect to attributes. $\mathcal{S}_R$ is proposed for combining the information of opposite edges between a node pair in a directed graph. The way these similarity scores are calculated depends on the adversary's comprehension and domain knowledge of the graph.

5.2. Generalized Node Similarity

Given a pair of rich graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, we still use the notation $\mathcal{S}(i, j)$ to represent the generalized similarity score of node $i \in V_1$ and $j \in V_2$, while additional attribute information is considered. The idea of neighbor matching is generalized to adopt attribute information. Two nodes are considered similar if (a) their attributes are similar, (b) their neighbors are similar, and (c) their relations with neighbors are similar.

In a directed graph, two nodes are considered to be adjacent if there is an edge between them, regardless of the edge's direction. The neighbor set notations $N_1(i)$ and $N_2(j)$ are generalized by regarding adjacent nodes as neighbors. In the k^{th} iteration, a bipartite graph $B_{i,j}^{(k+1)} = (N_1(i), N_2(j), N_1(i) \times N_2(j))$ is constructed by weighting edge $(i', j') \in N_1(i) \times N_2(j)$ as $\mathcal{S}^{(k)}(i', j')\mathcal{S}_R(i, i', j, j')$. Denoting the maximum matching as $\Theta_{i,j}^{(k+1)}$, the generalized node similarity is calculated iteratively as

$$\mathcal{S}^{(k+1)}(i, j) = \alpha \cdot \sum_{(i', j') \in \Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i', j')\mathcal{S}_R(i, i', j, j') + \mathcal{S}_X(i, j). \tag{3}$$

The constant factor α trades off the importance of node attribute against graph structure and edge attribute. The initial values are taken as $\mathcal{S}^{(0)}(i, j) = \mathcal{S}_X(i, j)$. The identity disclosures are obtained in exactly the same manner as described in Section 4.1.

Suppose G_1 is obtained by removing the nodes and edges from G_2 without attribute perturbation. It can be proven analogously, as in Section 4.2, that Theorem 4.1 still holds in this case (see Appendix B), so the generalized node similarity score still guarantees that a node is always one of the most similar candidates to itself in this case.

The original versions of the proposed algorithms require one to consider $|V_1||V_2|$ pairs of nodes, and they are surely not scalable. To solve this problem, we propose a pruning strategy to reduce the number of node pairs in consideration.

We refer to a node pair (i, j) as a *candidate pair*, where $i \in V_1$ and $j \in V_2$, and the similarity score of the candidate pair is $\mathcal{S}(i, j)$. We introduce a heuristic that limits the number of candidate pairs to enhance the scalability of the proposed algorithm with a little cost of accuracy. We find empirically that only a few candidate pairs have significantly large similarity scores and tend to be correct mappings, while candidate pairs with small similarity scores are very likely to be incorrect. This observation implies that we may simply maintain the top K candidate pairs with largest similarity scores in a set $\mathcal{C}$ during the algorithm process, and update only their similarity scores. For

simple graphs, the initial set $\mathcal{C}$ can be obtained by searching for similar nodes in the sense of node features such as node degrees [Hay et al. 2008], clustering coefficients [Watts and Strogatz 1998], centralities [Bonacich 1987], and recursive structural features proposed in Henderson et al. [2011]. Previous works of Hay et al. [2008] and Henderson et al. [2011] have shown that structural features can be used to prune a large fraction of candidate pairs at a reasonably good accuracy. With less remaining candidate pairs, our algorithm can scale for larger social graphs. For rich graphs, additional attribute information can be combined with structural features to obtain a smaller $\mathcal{C}$. While more sophisticated methods, for example, Korayem and Crandall [2013], can yield better performance, our experiments (Section 6.3) showed that even simply filtering by node attributes is sufficiently effective.

Assume the times required by comparing node-attribute sets and edge-attribute sets are $O(t_X)$ and $O(t_Y)$, respectively. In a single iteration of the original algorithm, $|V_1||V_2|$ pairs of node-attribute sets are compared. For a particular pair of nodes, at most d^2 pairs of edge-attribute sets are compared to construct the bipartite graph. Therefore, the original algorithm requires $O(|V_1||V_2|(t_X + d^2 t_Y + d^2 \log d))$ time for a single iteration. After applying the preceding heuristic, the running time is reduced to $O(K(t_X + d_c^2 t_Y + d_c^2 \log d_c))$. The space requirement for similarity scores is reduced to $O(K)$, where d_c denotes the upper bound of the node degree that is counted only with respect to the nodes in $\mathcal{C}$.

6. EXPERIMENTS

In this section, we start by introducing experiment settings, for example, the datasets, anonymization algorithms to be attacked, graph extraction algorithms, and evaluation criteria. We then evaluate the performance of the proposed algorithm on both simple graphs and rich graphs in various scenarios. Finally, we present empirical results about the relation between privacy risk and the eigenvector centrality.

6.1. Experiment Settings

6.1.1. Dataset. We used four public datasets for evaluation: a coauthor graph from Microsoft Academic Search (`msas`), a friendship graph from LiveJournal (`lj`), the Enron email dataset (`enron`), and user profile and relationship data from Tencent Weibo (`tweibo`).

The `msas` graph was published in WSDM 2013 Data Challenge. It was extracted from a snapshot (May 18, 2012) of the Microsoft Academic Search database. The graph is undirected, consists of 8,248 nodes and 18,732 edges, and does not contain any attribute. Every node corresponds to an author in the database. Two authors are linked by a single edge only if they have collaborated on at least one paper.

The `lj` dataset was analyzed in Yang and Leskovec [2012] and published at http://snap.stanford.edu/data/. The nodes correspond to users and the edges represent friendship between users. The original dataset contains about 4 million nodes. We randomly extracted a subgraph containing 10,000 nodes and 72,831 edges for our evaluation.

The `enron` dataset is derived from emails sent from and to managers in Enron Corporation, and it is available at http://www.cs.cmu.edu/~enron/. We regarded unique email addresses as nodes. An edge was added between two nodes if they have exchanged at least one pair of emails. The resulting graph contains 8,678 nodes and 29,400 edges.

The `tweibo` dataset was published in KDD Cup 2012 and contains a social graph and recommendation records. The dataset was naively anonymized, that is, all key words and user identifiers were replaced by random unique numbers, but attributes like gender and birth year were preserved as they were. We model the `tweibo` dataset as a directed graph. The nodes correspond to user profiles, and the directed edges

describe the relation between two users. An edge (i, j) is described by a set of attributes, including whether user i is following user j, how many times j is mentioned in i's tweets, how many times i has retweeted j's tweets, and the number of comments that i has sent to j. In total, 2.3 million nodes and 55.4 million directed edges were extracted from the dataset.

6.1.2. Anonymization. In our experiments, we chose the following typical algorithms to generate anonymized target graphs.

Naive Anonymization. The naive approach simply shuffles the identifiers of nodes, and leaves the structure as it was. Since this is the simplest approach, we would expect the best deanonymization result for this approach.

k-Degree Anonymity(k). For every node i in the anonymized graph, the k-degree anonymity requires that there are at least $k-1$ other nodes of the same degrees with i. We implemented two versions of the algorithms proposed by Liu and Terzi [2008]: the one that only adds edges, and the one that adds and deletes edges simultaneously.

Sparsification(p). The sparsification approach removes $p|E|$ edges randomly, where the parameter p controls the level of anonymization.

Perturbation(p). The perturbation approach first removes edges in exactly the same way as the sparsification does, and then adds false edges until the number of edges of the anonymized graph is the same as the original graph. This approach can be viewed as a kind of simulation of social graph evolution, or "unintended" anonymization.

Switching(p). The switching approach randomly selects two edges (i_1, j_1) and (i_2, j_2), provided that (i_1, j_2) and (i_2, j_1) are not in the graph. The selected edges are then "switched," that is, (i_1, j_1) and (i_2, j_2) are deleted, and (i_1, j_2) and (i_2, j_1) are added to the graph. Such procedure is repeated $p|E|/2$ times, resulting in $p|E|$ edge additions and $p|E|$ edge deletions.

Spectrum Preserving(p). Ying and Wu [2008] proposed two spectrum preserving randomization algorithms that can be viewed as variants of random perturbation and switching, and they were shown to keep better data utility. We implemented the two algorithms and used parameter p to control the number of modifications as earlier.

The change of attributes in rich graphs (`tweibo`) was simulated by randomization. For each node or edge in the target graph, we perturbed a field in its attribute set independently with a probability t as follows:

Following. As it was binary, the value was simply flipped.

Gender. The gender was chosen arbitrarily between male and female if the original value was unknown, or it was changed to unknown if it was originally known.

Birth Year. The birth year was assigned with an arbitrary value from 1900 to 2000.

Other Numeric Attributes. The new value of numeric attributes such as the number of tweets, comments, etc., was chosen arbitrarily in range $[x - xt, x + xt]$, provided that the original value was x.

6.1.3. Graph Extraction. The test data were generated from the original graph (`msas`, `lj`, `enron`, or `tweibo`) by extracting subgraphs. For a certain experiment setting, a pair of graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ with specified overlap were extracted from the original graph. We took G_1 as the auxiliary graph and G_2 as the target graph. Copies of G_2 were anonymized with different combinations of algorithms and parameters.

We defined the node overlap as $\beta_V = |V_1 \cap V_2|/|V_1 \cup V_2|$. Given β_V, V_1's desired size n_1, and the original graph $G = (V, E)$, we first extracted a subgraph $G_O = (V_O, E_O)$ of size $\beta_V|V|$ from G, using the Breadth-First Search (BFS), which is a strategy in online

social network crawling. The other $|V| - |V_O|$ nodes were randomly partitioned into two groups of size $n_1 - |V_O|$ and $|V| - n_1$, respectively. The node sets V_1 and V_2 were then obtained by combining the corresponding groups with V_O. Finally, G_1 and G_2 were obtained by projecting G on V_1 and V_2.

6.1.4. Evaluation Criteria. We measured how successful an attack is by its *precision* and *recall*. The precision was defined as the proportion of correctly matched nodes among all matched nodes, and the recall was defined as the proportion of correctly matched nodes among all overlapping nodes of G_1 and G_2. To save space, in most of our experiments, we only reported the precision p under various M (the number of outputted mappings; see Algorithm 1), and the recall can be then calculated as $r = pM/|V_1 \cap V_2|$. All experiments were performed 10 times with independently extracted graphs.

6.2. Simple Graphs

The `msas`, `lj`, and `enron` graphs were used to evaluate the performance of our algorithm on simple graphs. We started by evaluating the performance when the overlap of the auxiliary and the target graphs varied. We then compared our algorithm with the algorithms proposed in Narayanan et al. [2011] and Narayanan and Shmatikov [2009]. Empirical results about the relation between privacy risks and eigenvector centrality were also reported.

6.2.1. Overlap. We assumed that the auxiliary graph $G_1 = (V_1, E_1)$ might have arbitrary overlap with the target graph $G_2 = (V_2, E_2)$. We generated test data by extracting graph pairs with different overlap (25%, 50%, and 100%). Although the sizes of the auxiliary graph and the target graph may vary in a practical attack, we took the case where the two graphs were equally sized for illustration, that is, each of the graphs from `msas` contains 5,155, 6,186, or 8,248 nodes; 6,250, 7,500, or 10,000 nodes for the `lj` graph; or 5,424, 6,509, and 8,678 nodes for the `enron` graph, depending on the overlap. For each graph pair, copies of G_2 were sanitized with different anonymization algorithms. The results (Figures 1, 2, and 3) showed that the precision and the recall (see Section 6.1.4) for identity disclosure of our algorithm were reasonably high, even if the overlap was relatively small.

Provided that the adversary knows only the degree of nodes, the k-degree anonymity guarantees a node to be reidentified with a probability of at most $1/k$. However, with extra structure knowledge, the probability of success was increased dramatically. As there is usually no guarantee of what kind of information that an adversary has in practice, the result showed the potential privacy risks in such situations.

It was expected that the first few mappings that the algorithm generated were very likely to be correct, but this appeared not true for the k-degree anonymization algorithm, which only adds edges. Our algorithm appeared to favor large degree nodes first, for example, about half of the first 100 mappings corresponded to the first 100 top degree nodes. As node degree in a real-world graph is usually skewed [Albert and Barabási 2002], the degree difference between top nodes tended to be large, so significantly more edge additions were required to achieve k-degree anonymity. Taking the `msas` graph and $k = 10$ for example, the average degree of the top 100 nodes was increased by 34%, while the overall degree increment was only 3%. The neighborhoods of the top degree nodes were greatly changed, so they were difficult to be reidentified.

For a given randomization parameter p, the sparsification was the most vulnerable to be attacked. It was not surprising, since it made the least modifications to the graph by only deleting edges. Although the switching adds and deletes the same numbers of edges as the perturbation, it provided less protection. We believed it is due to the fact that it preserved node degrees. For the spectrum preserving approach, switching also seemed to provide less protection than perturbation does. With the same number of modifications, the spectrum preserving approach was thought to better preserve

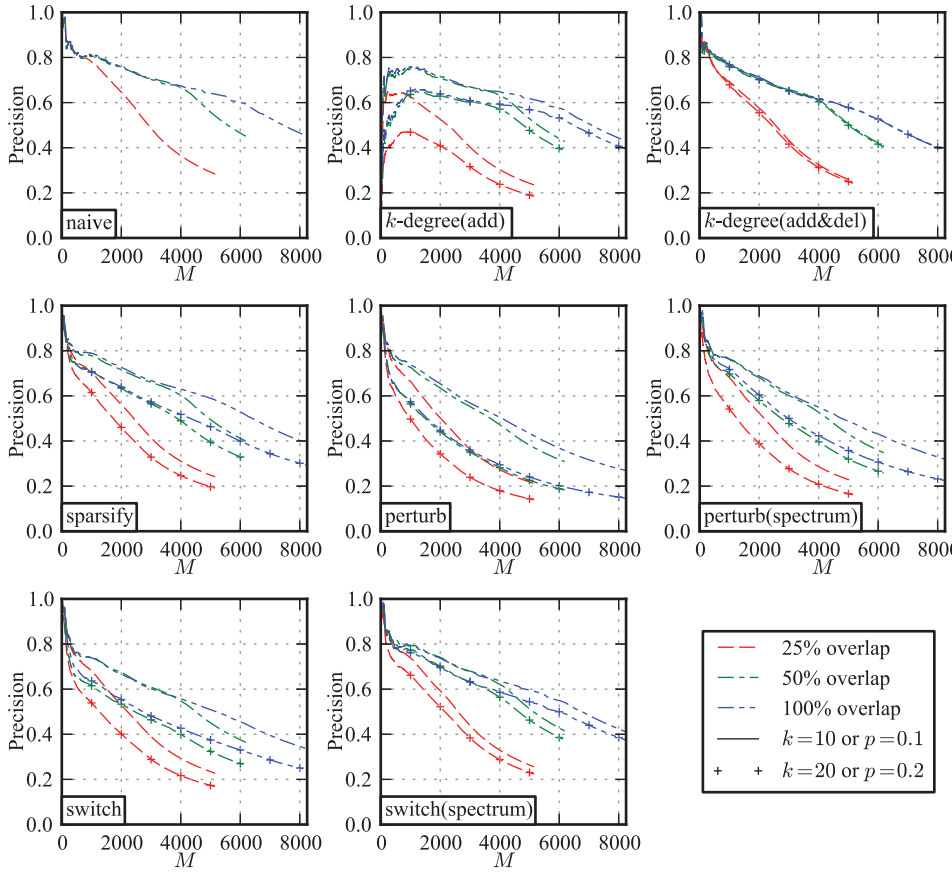

Fig. 1. Precision of attacks with different M (the number of mappings) and node overlap on msas graph.

the graph's utility, but it was also shown to provide less protection than the uniform randomization approach. For all anonymization algorithms here, increasing the level of anonymization indeed brought more difficulty to attacks. The randomization methods were supposed to protect privacy in a probabilistic manner, but the result showed that the nodes could still be reidentified with high precision even if 20% of the edges were modified.

6.2.2. Subgraph Attack. As the subgraph attack was widely studied in previous works, we have also evaluated the performance of our algorithm on subgraph attacks. We extracted subgraphs containing fractions of 25%, 50%, and 100% nodes of the original graphs (2,062, 4,124, and 8,248 nodes for the msas graph; 2,169, 4,339, and 8,678 nodes for the enron graph; or 2,500, 5,000, and 10,000 nodes for the lj graph), and anonymized copies of the original graph with the same anonymization algorithms and parameters to obtain the target graphs. The results showed similar accuracy and behaviors, so they were omitted here.

6.2.3. Eigenvector Centrality. Bonacich [1987] proposed a family of centrality measurements that measure the importance or influence of a node in a graph. One of them is the *eigenvector centrality*, which is defined as the principal eigenvector of a graph's adjacency matrix. We grouped the nodes in the auxiliary graph by their normalized

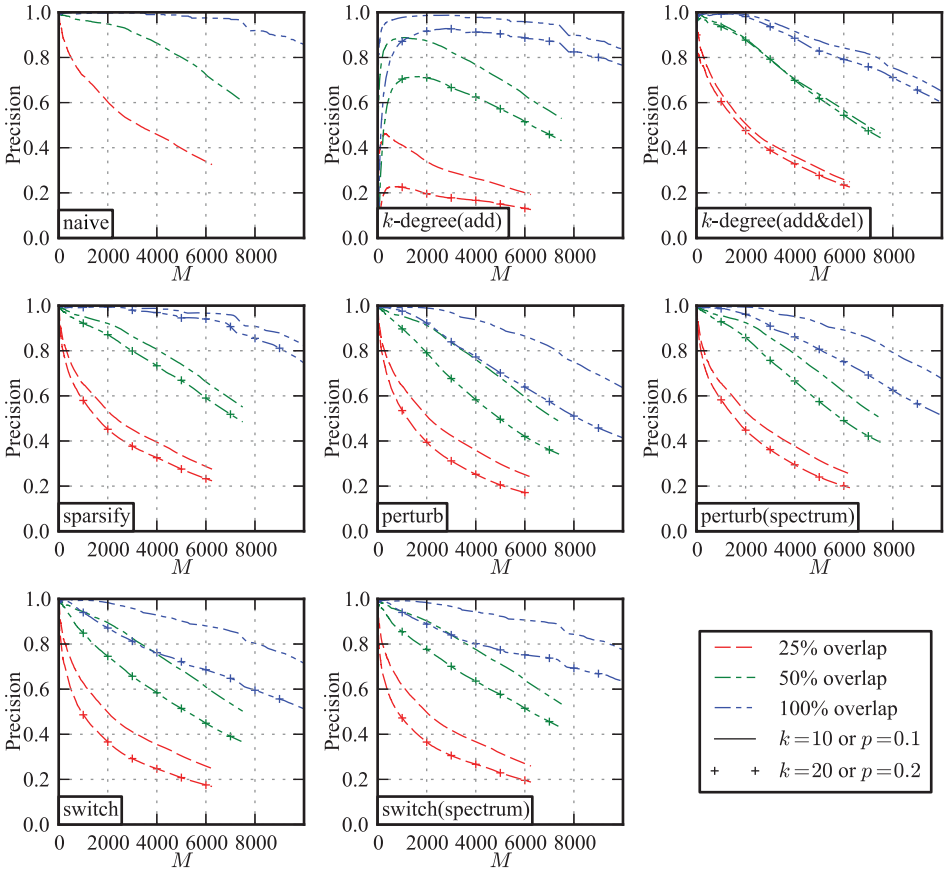

Fig. 2. Precision of attacks with different M (the number of mappings) and node overlap on `1j` graph.

eigenvector centrality and calculated the average deanonymization precision for every group. The ranges of bins were selected in an exponential manner to balance the number of nodes in different bins. The result (Figure 4) showed that important nodes (with high eigenvector centrality) were more likely to be reidentified.

We studied the distribution of eigenvector centrality of the `msas`, `1j`, and `enron` graphs and found that, while a few nodes had unusually high centrality, the others' centralities were rather low. We also found that those nodes with high centrality were very distinguishable from each other, since the centralities were diverse. As discussed in Section 4.2, nodes with similar eigenvector centrality could be confused with each others. Therefore, it was expected that nodes with larger eigenvector centrality were easier to be reidentified.

6.2.4. Efficiency. We implemented our algorithm in C++ with multithreading. The experiments were performed on a server equipped with an Intel Xeon X5660 CPU (six cores, 2.8GHz), and 2GB memory was actually used. For the sake of simplicity, we kept all $|V_1||V_2|$ candidate pairs, so the time complexity of a single iteration is $O(|V_1||V_2|d^2 \log d)$ and the total running time is propositional to the graph size. As mentioned in Section 4.1, we limited the number of iterations to five. Take the experiment on the `1j` graph presented in Section 6.2.1 for example, where the auxiliary and

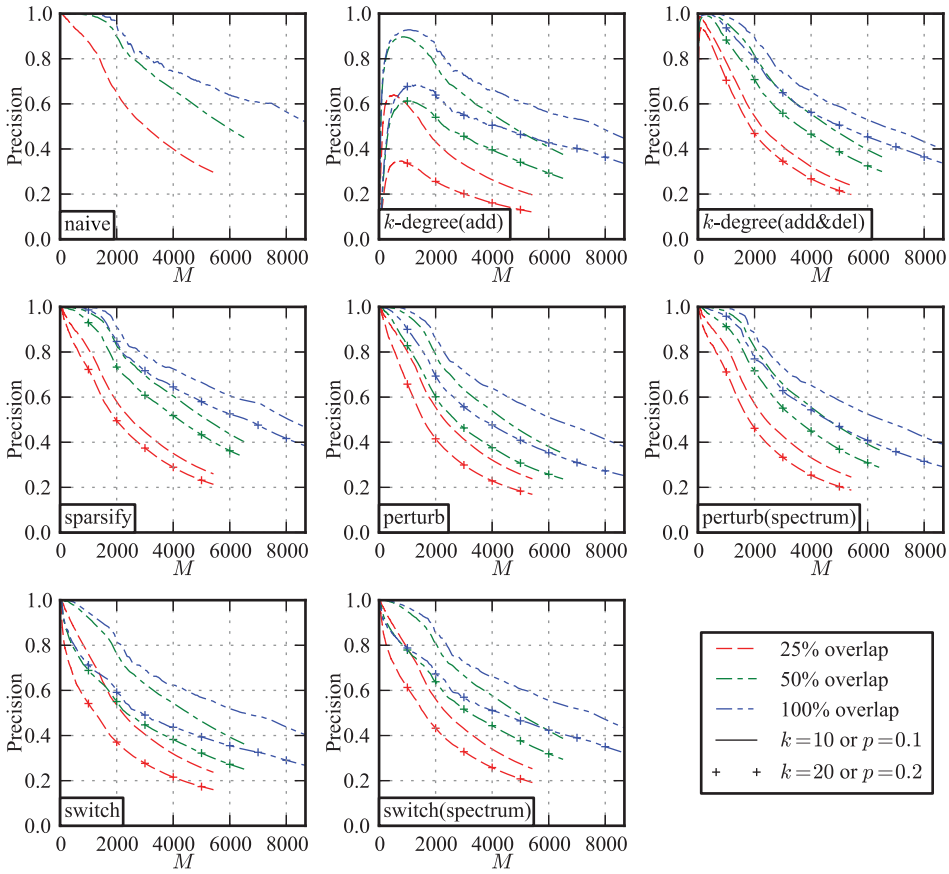

Fig. 3. Precision of attacks with different M (the number of mappings) and node overlap on `enron` graph.

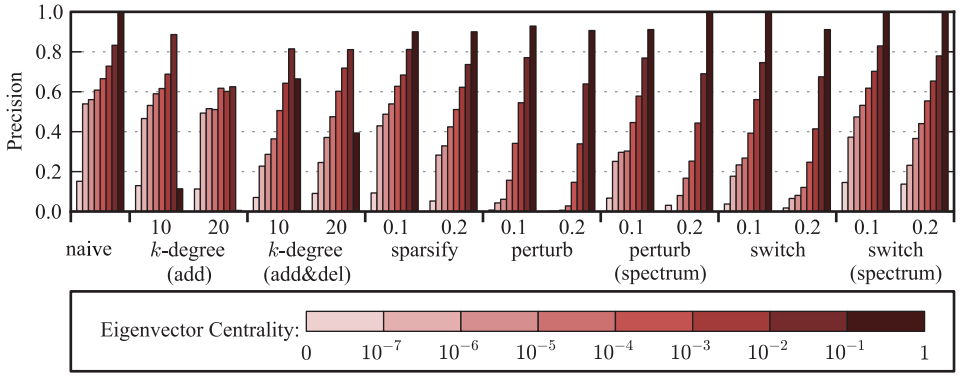

Fig. 4. Precision of identity disclosure grouped by the auxiliary graph's normalized eigenvector centrality (100% overlap, msas). Results for other graphs were similar.

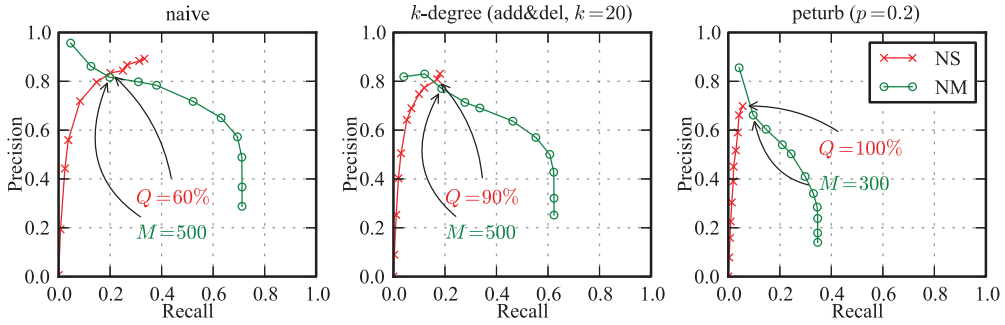

Fig. 5. Performance of NeighborMatch (NM) and NS attacks (25% overlap, msas). The curves for NM and NS attacks are generated by varying the number of mappings ($M = 100, 300, 500, 800, 1{,}000, 1{,}500, 2{,}000, 2{,}500, 3{,}000, 4{,}000, 5{,}000$) and the quality of seed mappings ($Q = 0\%, 10\%, \ldots, 100\%$), respectively. Note that the curves for NS attacks are not standard precision/recall curves due to the different parameters for different data points. Results for other anonymization algorithms and datasets are similar and thus omitted.

the target graphs contain 6,250, 7,500, or 10,000 nodes. The total running time for a single attack ranged from 10 to 20 minutes.

6.2.5. Comparison. Narayanan and Shmatikov [2009] proposed a deanonymization algorithm (denoted as `NS`) that requires a certain number of *seed mappings* to invoke large-scale reidentification. Collecting seed mappings can be considered as a small-scale deanonymization, which is not always feasible in practice. Taking the `tweibo` dataset as an example, we were only able to match a few dozens of user profiles, and the mappings were proven to be incorrect after manual checking. That means, even if an adversary managed to collect a few seed mappings, there was no guarantee on the quality of mappings, that is, the mappings could be incorrect.

In our experiments, we assumed an ideal situation where seed mappings were given in advance. We generated 50 mappings by randomly sampling the overlapping nodes, provided that their degrees were at least 10. We were interested in the impact of seed mapping quality, so we randomly picked 0%,10%, . . . , 100% of the given mappings and made them incorrect (by replacing one of the two nodes with an arbitrary node). We evaluated the precision and recall of the `NS` algorithm in such settings, and the process was repeated 10 times independently. We set the parameter `theta` in the `NS` algorithm to 0.1 after trying various choices.

To compare with `NS`, we reported the performance of our algorithm with different values of M (the number of outputted mappings). The result (Figure 5) showed that for `NS`, both precision and recall increased as the quality of seed mappings increased. However, its recall was low, though all seed mappings were correct (we have tried to use more than 50 seed mappings but the result was similar). The result also showed that only when high quality seed mappings were provided, the `NS` algorithm could outperform ours. Note that this comparison is actually unfair for us, since additional information is provided to the `NS` algorithm. We believe the observed differences are mainly due to two aspects:

—Our algorithm represents the node mappings with similarity scores, while the `NS` only maintains a set of discrete 1-1 mappings. Therefore, our algorithm is able to capture more information about the graph structure.
—The `NS` algorithm accepts a mapping only when the two nodes are both the most similar to each other. This constraint is so strict that the algorithm only accepts mappings that are very certain to be correct. Therefore, it prevents the propagation

through uncertain nodes. On the contrary, our algorithm provides more flexibility as one may tune the parameter M to trade off between precision and recall, depending on the application.

Narayanan et al. [2011] later proposed a graph matching algorithm based on simulated annealing (denoted as SA). The algorithm was used to generate seed mappings by matching the top n degree nodes ($n \leq 100$), based on the observation that the top degree nodes of both graphs roughly correspond to each other. We measured the performance by accuracy, which was defined as the fraction of correct mappings among all actual mappings between the top n degree nodes of both graphs. In our experiments, we ran the SA algorithm with $n = 20$ on graph pairs with 100% overlap, and the result was reported as the median of 30 trials. For most of the anonymization algorithms mentioned in Section 6.1.2, the average accuracy of our algorithm for the top 20 nodes was at least 99%, 95%, and 94% for the msas, lj, and enron graphs, respectively. The highest accuracy of the SA algorithm, which turned out to be obtained against the naive anonymization, was 32%, 84%, and 60%. For the k-degree anonymization algorithm that only adds edges, both of the algorithms performed poorly on the top 20 nodes ($\leq$8%), and this was expected as discussed in Section 6.2.1. We also tried other values for n and the node overlap, and obtained similar results. We further took the output of the SA algorithm ($n = 50$) as seed mappings for the NS algorithm. The resulting precision varied from 2% to 76%, depending on datasets and anonymization algorithms, but the recall was always lower than 10%.

As shown in Section 6.2.3, important nodes with higher eigenvector centrality were more likely to be reidentified. We investigated the possibility of using eigenvector centrality to obtain better seed mappings. We ran the SA algorithm to identify the top $n = 20$ nodes with the same experiment settings as earlier. It turned out that picking the top eigenvector centrality nodes was indeed a better strategy, and the highest accuracy (obtained against the naive anonymization) for the msas, lj, and enron graphs were 91%, 74%, and 100%, respectively. However, the accuracy varied depending on the anonymization algorithm that was applied. For example, the accuracy for the three graphs decreased to 82%, 58%, and 60%, respectively, when the target graphs were sparsified with $p = 0.1$. On the other hand, our algorithm was robust against different kinds of anonymizaton algorithms, and its accuracy was always close to 100%.

6.3. Rich Graphs

We further explored the performance of our algorithm on rich graphs. As deanonymizing the actual tweibo graph was infeasible, we simulated the process of auxiliary graph collection and deanonymization instead.

6.3.1. Algorithm Settings. As mentioned in previous sections, the attribute similarity measurements are determined by the adversary's domain knowledge of the graph. Herein, we introduce the exact definitions for node-attribute similarity, edge-attribute similarity, and relation similarity used in our experiments.

Node-Attribute Similarity. We used three attributes—gender, birth year, and number of tweets—to measure node-attribute similarity. The similarity of gender was measured in a trivial way: 1 for equal values, 1/2 if either value is unknown, or 0 otherwise. The birth years of two users were compared in this way: 1 for equal values, 1/2 if the absolute difference is 1, or 0 otherwise. The numbers of tweets were compared as $r(x, y) = \min\{x, y\}/\max\{x, y\}$. The node-attribute similarity score S_X was then taken as the average of the three scores.

Edge-Attribute Similarity. The attribute of "following" relation was measured as 1 for equal values, or 0 if not equal. The other three numeric attributes were

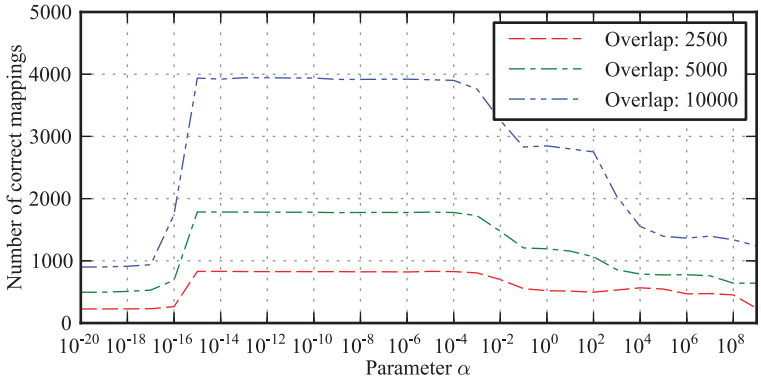

Fig. 6. Precision of attacks for naive anonymization with various α (tweibo).

all measured by $r(x, y)$. The edge-attribute similarity score $\mathcal{S}_Y$ was taken as the average of the four scores.

Relation Similarity. To compare relations (i_1, j_1) and (i_2, j_2), we first calculated the edge-attribute similarity scores of two edge pairs in opposite directions, which were denoted as $\mathcal{S}_{Y1} = \mathcal{S}_Y(i_1, j_1, i_2, j_2)$ and $\mathcal{S}_{Y2} = \mathcal{S}_Y(j_1, i_1, j_2, i_2)$. If both edge pairs were presented, $\mathcal{S}_R$ was taken as the average of $\mathcal{S}_{Y1}$ and $\mathcal{S}_{Y2}$. Otherwise, the pair with a missing edge was omitted, or zero if too many edges were missing.

6.3.2. Parameters. We kept the top $K = 10^7$ candidate pairs (see Section 5.2) to reduce running time and space requirement. The parameter α (see Equation (3)) trades off the importance of node attributes against graph structure and edge attributes. In addition, the value of α is expected to be small to scale down the sum of neighboring similarity scores to a comparable magnitude of node-attribute similarity for large-scale graphs. In this sense, the determination of the value of α also involves the size of graphs. While it is nontrivial to assess the impact of the preceding aspects, an empirical approach could be employed to determine the proper value for α.

We carried out an experiment over different choices of α to find the best α in terms of overall precision. We randomly extracted graph pairs with different overlaps (2,500, 5,000, and 10,000 nodes), where the auxiliary graph was limited to 10,000 nodes and the target graph contained the other nodes (about 2.3 million). The result (Figure 6) showed that values in a range between 10^{-15} and 10^{-4} yielded approximately the highest precision. While our algorithm was not sensitive for different choices of α in this range, we took $\alpha = 10^{-4}$ as a modest choice in consideration of robustness. With this setting, the sum of neighboring similarity scores was scaled to a similar magnitude of node-attribute similarity in Equation (3).

The result also indicated that utilizing only the information of graph structure and edge attributes ($\alpha \to \infty$) was not satisfactory for deanonymization. Similarly, node attributes alone ($\alpha \to 0$) were shown insufficient to identify the nodes too. However, when the two types of information were incorporated, the precision of attacks was boosted significantly.

6.3.3. Evaluation. For rich graphs, we were interested in evaluating the impact of altering graph structure and attributes on the deanonymization performance. We modified the graph structure by applying anonymization algorithms described in Section 6.1.2. The k-degree anonymity approach was infeasible for rich graphs due to the attributes, so we omitted it here. For the perturbation approach, it was hard to decide the attributes of new edges, so we took the approach proposed in Narayanan and Shmatikov

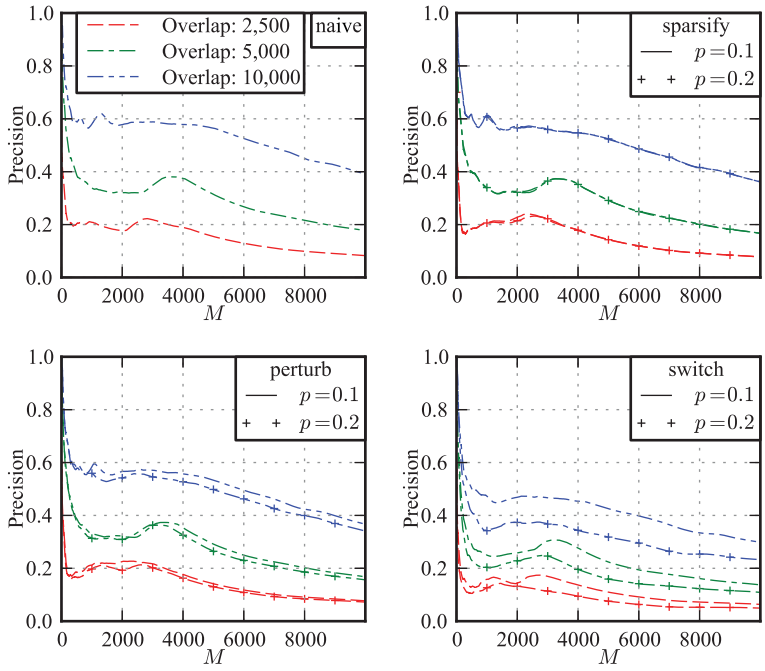

Fig. 7. Precision of attacks with different node overlap (tweibo, $K = 10^7$).

[2009]. Briefly, for a desired perturbation proportion p, we removed a proportion of $q = p/(2 - p)$ edges in both the auxiliary and target graphs independently, therefore the expected proportion of edge overlap was $(1 - q)^2/(1 - q^2) = 1 - p$.

Recall that the node overlap is between only 0.1% and 0.4%; the result (Figure 7) showed reasonable precision and recall in deanonymizing rich graphs. Compared with the naive approach, modifying the graph structure did not bring much difficulty to deanonymization. Different randomization strategies did not differ much in terms of precision. Further results (Figure 8) showed that attributes played important roles in deanonymization and attribute perturbation reduced the precision to a considerable extent. As the attributes of nodes and edges provided meaningful information to distinguish them, it was expected that attacking would be easier with such additional information even if the auxiliary graph was rather smaller than the target graph.

6.3.4. Efficiency. According to Section 5.2, the running time highly depends on K (the number of maintained top candidate pairs). We used naive anonymization as a case study to demonstrate the performance of our algorithm with different values of K. The result (Figure 9) showed that our algorithm was efficient, provided that a reasonable amount of the outputted mappings were correct. It can also be seen that the number of correct mappings did not increase when a certain number of candidate pairs were reached.

7. CONCLUSIONS

In this article, we proposed a deanonymization algorithm for social graphs based on a node similarity measurement in consideration of graph structure and attributes. We

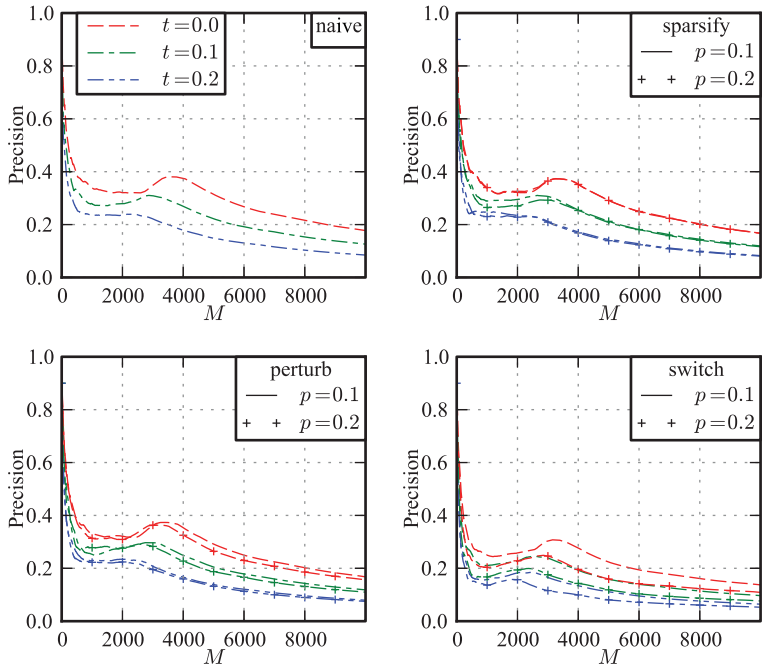

Fig. 8. Precision of attacks with different attribute perturbation (tweibo, $K = 10^7$, 5,000 overlap). Results for other overlaps show similar behavior and are thus omitted.

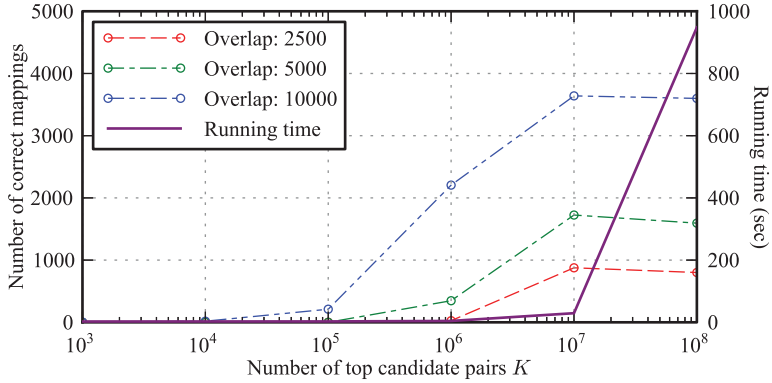

Fig. 9. Precision and running time of attacks for naive anonymization (tweibo, $K = 10^3, 10^4, \ldots, 10^8$).

evaluated the proposed algorithm against several typical anonymization algorithms on four real datasets. Our results suggested that

—the proposed algorithm was efficient to deanonymize social graphs on a large scale;
—using the proposed attack and structural knowledge, for example, subgraphs around target nodes, an adversary can easily break the anonymization algorithms that were evaluated in this article;
—additional descriptive information made attacks more effective, even if it was inaccurate and the graph overlap is small; and
—important nodes (in terms of eigenvector centrality) were at high privacy risks in the proposed attack.

While it is usually agreed that a data publisher should never totally rely on any anonymization algorithm, our result illustrated how easily an adversary could break several typical anonymization algorithms. For simple graphs, the most straightforward approach to resist our attack is to modify the graph structure so that the eigenvector centrality is not that diverse. However, we doubt the resulting data utility, because the eigenvectors of a graph's adjacency matrix are related to several important graph metrics [Ying and Wu 2008].

As the scale of users is of millions in social networking sites, one may think of publishing a small but representative sample [Leskovec and Faloutsos 2006] of the full data source. As long as the auxiliary graph collected by the adversary has little overlap with the published graph, an effective attack would become infeasible. However, the sampling could be tricky. For example, the dataset of IJCNN 2011 Social Network Challenge, which is obtained from Flickr, was revealed to be biased towards high-degree nodes. During the challenge, Narayanan et al. [2011] employed a crawling strategy that is also biased toward high-degree nodes.[6] Although the challenge dataset (1 million nodes) and the crawled dataset (2 million nodes) account for only small fractions of Flickr (51 million users in 2011[7]), the edge overlap turns out to be quite high (95.6%) and that is one of the keys of the success of deanonymization.

A rich graph contains more details about users and their interactions, so it is surely of more interest for analysis. Our results indicated several challenges in protecting privacy in rich graphs. First, graph structure and attributes may be of different importance for application. On the other hand, our result illustrated the different impact of modifying graph structure and attributes on privacy preservation. Therefore, the balance between the two aspects should be considered when designing anonymization algorithms for rich graphs. Second, the generalization approach is usually adopted to anonymize attributes [Cormode et al. 2008; Campan and Truta 2009]. Instead of providing the exact values of attributes thta make the deanonymization easier, the generalization approach replaces attributes to values less specific, for example, generalizing birth year 1991 to 1990s. The trade-off is that the more general the values are, the better protection and the less utility are provided. However, comparison of attributes in an immediate manner is still feasible. As suggested by our results, the generalization approach appeared to be vulnerable to our attack when the graph structure was presented. Finally, given that rich graphs are much more vulnerable than simple graphs, we believe that it is necessary to rethink the problem of releasing such data; for example, a different schema other than modifying graph structure and attributes may be necessary.

APPENDIX

A. CONVERGENCE OF THE NEIGHBORMATCH ALGORITHM

Analogously to Section 4.1, we start with the case where G_1 is a subgraph of G_2. We denote the adjacency matrix of G_1 as A_1 and its principal eigenvalue as λ_1. The corresponding eigenvector is denoted as $\mathbf{v}_1$ and it is normalized so that $\max\{v|v \in \mathbf{v}_1\} = 1$.

We first consider a minor modification of Algorithm 1. Instead of $\mathcal{S}^{(0)}(i, j) = 1$, we take $\mathcal{S}^{(0)}(i, j) = \mathbf{v}(i)$ as the initial values (Line 2 in Algorithm 1). The following discussion is based on the modified version of the algorithm. The modification provides a simple representation of the normalized similarity scores, as stated in the following.

[6]http://www.kaggle.com/c/socialNetwork/forums/t/257/how-we-did-it-ind-cca.
[7]http://royal.pingdom.com/2012/01/17/internet-2011-in-numbers.

THEOREM A.1. *The normalized node similarity score is obtained as* $\mathcal{R}^{(k)}(i,j) = \lambda_1^{-k}\mathcal{S}^{(k)}(i,j)$.

PROOF. We first need to prove that Theorem 4.1 still applies to the modified algorithm. The proof in Section 4.2 actually still works here, because the base case of induction $\mathcal{S}^{(0)}(i,i) \geq \mathcal{S}^{(0)}(i,j)$ still holds and nothing else is changed.

According to Equation (2), we have $\mathcal{T}^{(k+1)} = A_1\mathcal{T}^{(k)}$ for $k \geq 0$. On the other hand, $\mathcal{T}^{(0)} = \mathbf{v}_1$ and $A_1\mathbf{v}_1 = \lambda_1\mathbf{v}_1$ by definition, so we have that

$$\begin{aligned}\mathcal{T}^{(k)} &= A_1^k\mathbf{v}_1 \\ &= \lambda_1^k\mathbf{v}_1. \end{aligned} \tag{4}$$

The inverse of the scaling factor for normalization is

$$\begin{aligned}\max_{i\in V_1, j\in V_2}\{\mathcal{S}^{(k)}(i,j)\} &= \max_{i\in V_1}\{\mathcal{T}^{(k)}(i)\} \\ &= \lambda_1^k \cdot \max_{i\in V_1}\{\mathbf{v}_1(i)\} \\ &= \lambda_1^k.\end{aligned}$$

This shows that the normalization is simply done by dividing $\mathcal{S}^{(k)}$ by λ_1^k, so the normalized similarity score is $\mathcal{R}^{(k)}(i,j) = \lambda_1^{-k}\mathcal{S}^{(k)}(i,j)$. □

Now we are ready to prove the convergence of the normalized similarity scores.

THEOREM A.2. *For any* $i \in V_1$ *and* $j \in V_2$*, the normalized similarity score* $\mathcal{R}^{(k)}(i,j)$ *converges.*

PROOF. We shall prove the sequence $\{\mathcal{R}^{(0)}(i,j), \mathcal{R}^{(1)}(i,j), \mathcal{R}^{(2)}(i,j), \ldots\}$ is decreasing by induction. By definition, we have $\mathcal{R}^{(0)}(i,j) = \mathcal{S}^{(0)}(i,j) = \mathbf{v}_1(i)$. Theorem 4.1 and Theorem A.1 show that

$$\begin{aligned}\mathcal{R}^{(1)}(i,j) &= \lambda_1^{-1}\mathcal{S}^{(1)}(i,j) \\ &\leq \lambda_1^{-1}\mathcal{T}^{(1)}(i) \\ &= \lambda_1^{-1} \cdot (\lambda_1\mathbf{v}_1(i)) \\ &= \mathcal{R}^{(0)}(i,j).\end{aligned} \tag{5}$$

Hence, the claim $\mathcal{R}^{(0)}(i,j) \geq \mathcal{R}^{(1)}(i,j)$ holds for the base case.

For $k \geq 1$, we assume $\mathcal{R}^{(k-1)}(i,j) \geq \mathcal{R}^{(k)}(i,j)$. The updating rule of $\mathcal{S}(i,j)$ (Equation (1)) is rearranged as

$$\begin{aligned}\mathcal{R}^{(k+1)}(i,j) &= \lambda_1^{-(k+1)} \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i',j') \\ &= \lambda_1^{-1} \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{R}^{(k)}(i',j').\end{aligned} \tag{6}$$

When the weights of the bipartite graph $B_{i,j}^{(k+1)}$ are assigned based on $\mathcal{R}^{(k)}$, the match $\Theta_{i,j}^{(k+1)}$ is still optimal, since $\mathcal{R}^{(k)}$ differs from $\mathcal{S}^{(k)}$ by a factor of λ_1^{-k} that is a constant

within an iteration. By assumption, we have that

$$\begin{aligned}\mathcal{R}^{(k+1)}(i,j) &= \lambda_1^{-1} \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{R}^{(k)}(i',j') \\ &\le \lambda_1^{-1} \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{R}^{(k-1)}(i',j') \\ &\le \lambda_1^{-1} \sum_{(i',j')\in\Theta_{i,j}^{(k)}} \mathcal{R}^{(k-1)}(i',j') \\ &= \mathcal{R}^{(k)}(i,j). \end{aligned} \tag{7}$$

Given that the sequence $\{\mathcal{R}^{(0)}(i,j), \mathcal{R}^{(1)}(i,j), \mathcal{R}^{(2)}(i,j), \ldots\}$ is decreasing and bounded below by 0, the monotone convergence theorem shows that $\mathcal{R}^{(k)}(i,j)$ converges as k increases. □

For the general case where G_1 and G_2 have arbitrary overlap, it is unclear yet if the original algorithm (Algorithm 1) will always converge, but our initial experiments show that the difference of similarity scores goes below 10^{-3} after less than 20 iterations for the datasets used in Section 6. In addition, we find it unnecessary to iterate until convergence, and a small number of iterations (five iterations in our experiments) are enough for effective deanonymization.

B. PROOF OF THEOREM 4.1 FOR GENERALIZED NODE SIMILARITY

PROOF. By assumption, we have $\mathcal{T}^{(0)}(i) = \mathcal{S}_X(i,i) = 1$ and $\mathcal{S}^{(0)}(i,j) = \mathcal{S}_X(i,j) \le 1$, so the claim $\mathcal{T}^{(0)}(i) \ge \mathcal{S}^{(0)}(i,j)$ holds for the base case of induction. For $k \ge 0$, we assume the claim is true for k. By definition of the bipartite graph $B_{i,j}^{(k+1)}$, the edge (i',i') has the maximum weight:

$$\begin{aligned} w(i',i') &= \mathcal{T}^{(k)}(i)\mathcal{S}_R(i,i',i,i') \\ &\ge \mathcal{S}^{(k)}(i,j)\mathcal{S}_R(i,i',j,j') \\ &= w(i',j'). \end{aligned}$$

Hence, $\Theta_{i,i}^{(k+1)} = \{(i',i') | i' \in N_1(i)\}$ is one of the optimal matches for $\mathcal{T}^{(k)}(i)$. As all optimal matches have the same overall scores, the updated self-similarity score is

$$\begin{aligned}\mathcal{T}^{(k+1)}(i) &= \alpha \cdot \sum_{i'\in N_1(i)} \mathcal{S}^{(k)}(i',i') + \mathcal{S}_X(i,i) \\ &= \alpha \cdot \sum_{i'\in N_1(i)} \mathcal{T}^{(k)}(i') + 1. \end{aligned} \tag{8}$$

We then have that

$$\begin{aligned}\mathcal{S}^{(k+1)}(i,j) &= \alpha \cdot \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{S}^{(k)}(i',j') + \mathcal{S}_X(i,j) \\ &\le \alpha \cdot \sum_{(i',j')\in\Theta_{i,j}^{(k+1)}} \mathcal{T}^{(k)}(i') + 1 \\ &\le \alpha \cdot \sum_{i'\in N_1(i)} \mathcal{T}^{(k)}(i') + 1 \\ &= \mathcal{T}^{(k+1)}(i). \quad \square \end{aligned}$$

ACKNOWLEDGMENTS

We would like to gratefully acknowledge the organizers of WSDM 2013 Data Challenge as well as Microsoft Academic Search for providing the `msas` dataset. We also acknowledge the organizers of KDD Cup 2012 as well as Tencent Inc. for making the `tweibo` dataset available. Finally, special thanks to Lei Zou from Peking University and Nicholas Yuan from Microsoft Research for their advice, and Yang Chen from Wuhan University for her special support.

REFERENCES

Réka Albert and Albert-László Barabási. 2002. Statistical mechanics of complex networks. *Reviews of Modern Physics* 74, 1 (Jan. 2002), 47–97.

Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. 2007. Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*. 181–190.

Albert-László Barabási, Hawoong Jeong, Zoltán Néda, Erzsébet Ravasz, András Schubert, and Tamás Vicsek. 2002. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications* 311, 3 (2002), 590–614.

Vincent D. Blondel, Anahí Gajardo, Maureen Heymans, Pierre Senellart, and Paul Van Dooren. 2004. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM Review* 46, 4 (April 2004), 647–666.

Phillip Bonacich. 1987. Power and centrality: A family of measures. *American Journa of Sociology* (1987), 1170–1182.

Francesco Bonchi, Aristides Gionis, and Tamir Tassa. 2011. Identity obfuscation in graphs through the information theoretic lens. In *Proceedings of the 27th International Conference on Data Engineering (ICDE'11)*. 924–935.

Alina Campan and Traian Marius Truta. 2009. Data and structural k-anonymity in social networks. In *Privacy, Security, and Trust in KDD*. 33–54.

James Cheng, Ada Wai-chee Fu, and Jia Liu. 2010. K-isomorphism: Privacy preserving network publication against structural attacks. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*. 459–470.

Stephen A Cook. 1971. The complexity of theorem-proving procedures. In *Proceedings of the 3rd ACM Symposium on Theory of Computing*. 151–158.

Graham Cormode, Divesh Srivastava, Ting Yu, and Qing Zhang. 2008. Anonymizing bipartite graph data using safe groupings. *Proceedings of the VLDB Endowment* 1, 1 (Aug. 2008), 833–844.

Cynthia Dwork. 2006. Differential privacy. *Automata, Languages and Programming* 4052 (2006), 1–12.

Hao Fu, Aston Zhang, and Xing Xie. 2014. De-anonymizing social graphs via node similarity. In *Proceedings of the 23rd International Conference on World Wide Web Companion (WWW'14)*.

Gabor Gyorgy Gulyas and Sandor Imre. 2014. Measuring importance of seeding for structural deanonymization attacks in social networks. In *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops'14)*. 610–615.

Pritam Gundecha, Geoffrey Barbier, and Huan Liu. 2011. Exploiting vulnerability to secure user privacy on a social networking site. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (KDD'11)*. 511–519.

Michael Hay, Gerome Miklau, David Jensen, Don Towsley, and Philipp Weis. 2008. Resisting structural reidentification in anonymized social networks. *Proceedings of the VLDB Endowment* 1, 1 (2008), 102–114.

Michael Hay, Vibhor Rastogi, Gerome Miklau, and Dan Suciu. 2010. Boosting the accuracy of differentially private histograms through consistency. *Proceedings of the VLDB Endowment* 3, 1 (2010).

Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. 2011. It's who you know: Graph mining using recursive structural features. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (KDD'11)*. 663–671.

Glen Jeh and Jennifer Widom. 2002. SimRank: A measure of structural-context similarity. In *Proceedings of the 8th ACM International Conference on Knowledge Discovery and Data Mining (KDD'11)*. 538–543.

Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. 2011. Private analysis of graph structure. *Proceedings of the VLDB Endowment* 4, 11 (2011), 1146–1157.

Mohammed Korayem and David J Crandall. 2013. De-anonymizing users across heterogeneous social computing platforms. In *the 7th International AAAI Conference on Weblogs and Social Media (ICWSM'13)*.

Harold W. Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1–2 (1955), 83–97.

Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (KDD'06)*. 631–636.

David Liben-Nowell and Jon Kleinberg. 2003. The link prediction problem for social networks. In *Proceedings of the 12th International Conference on Information and Knowledge Management (CIKM'03)*. 556–559.

Kun Liu and Evimaria Terzi. 2008. Towards identity anonymization on graphs. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*. 93–106.

Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. 2002. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th International Conference on Data Engineering (ICDE'02)*. 117–128.

Alan Mislove, Bimal Viswanath, Krishna P. Gummadi, and Peter Druschel. 2010. You are who you know: Inferring user profiles in online social networks. In *Proceedings of the 3rd International Conference on Web Search and Data Mining (WSDM'10)*. 251–260.

Prateek Mittal, Charalampos Papamanthou, and Dawn Song. 2013. Preserving link privacy in social network based systems. In *Network and Distributed System Security Symposium (NDSS'13)*.

Arvind Narayanan, Elaine Shi, and Benjamin I. P. Rubinstein. 2011. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN)*. 1825–1834.

Arvind Narayanan and Vitaly Shmatikov. 2009. De-anonymizing social networks. In *Proceedings of the 2009 IEEE Symposium on Security and Privacy*. 173–187.

Pedram Pedarsani and Matthias Grossglauser. 2011. On the privacy of anonymized networks. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (KDD'11)*. 1235–1243.

Alessandra Sala, Xiaohan Zhao, Christo Wilson, Haitao Zheng, and Ben Y. Zhao. 2011. Sharing graphs using differentially private graph models. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference (IMC'11)*. 81–98.

Chih-Hua Tai, Philip S. Yu, De-Nian Yang, and Ming-Syan Chen. 2011. Privacy-preserving social network publication against friendship attacks. In *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (KDD'11)*. 1262–1270.

Tamir Tassa and Dror J. Cohen. 2013. Anonymization of centralized and distributed social networks by sequential clustering. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 2 (Feb. 2013), 311–324.

Duncan J. Watts and Steven H. Strogatz. 1998. Collective dynamics of'small-world' networks. *Nature* 393, 6684 (June 1998), 440–442.

Leting Wu, Xiaowei Ying, and Xintao Wu. 2010b. Reconstruction from randomized graph via low rank approximation. In *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM'10)*. 60–71.

Wentao Wu, Yanghua Xiao, Wei Wang, Zhenying He, and Zhihui Wang. 2010a. *K*-symmetry model for identity anonymization in social networks. In *Proceedings of the 13th International Conference on Extending Database Technology (EDBT'10)*. 111–122.

Xintao Wu, Xiaowei Ying, Kun Liu, and Lei Chen. 2010c. A survey of privacy-preservation of graphs and social networks. *Managing and Mining Graph Data* (2010), 421–453.

Jaewon Yang and Jure Leskovec. 2012. Defining and evaluating network communities based on ground-truth. In *Proceedings of the 2012 IEEE International Conference on Data Mining (ICDM'10)*. 745–754.

Lyudmila Yartseva and Matthias Grossglauser. 2013. On the performance of percolation graph matching. In *Proceedings of the 1st ACM Conference on Online Social Networks (COSN'13)*. 119–130.

Xiaowei Ying and Xintao Wu. 2008. Randomizing social networks: A spectrum preserving approach. In *Proceedings of the 2008 SIAM International Conference on Data Mining (SDM)*. 739–750.

Xiaowei Ying and Xintao Wu. 2009. Graph generation with prescribed feature constraints. In *Proceedings of the 2009 SIAM International Conference on Data Mining (SDM'09)*, Vol. 9. 966–977.

Xiaowei Ying and Xintao Wu. 2011. On link privacy in randomizing social networks. *Knowledge and Information Systems* 28, 3 (2011), 645–663.

Aston Zhang, Xing Xie, Kevin Chen-Chuan, Carl A. Gunter, Jiawei Han, and XiaoFeng Wang. 2014. Privacy risk in anonymized heterogeneous information networks. In *Proceedings of the 17th International Conference on Extending Database Technology (EDBT'14)*.

Elena Zheleva and Lise Getoor. 2008. Preserving the privacy of sensitive relationships in graph data. In *Privacy, Security, and Trust in KDD*. 153–171.

Bin Zhou and Jian Pei. 2008. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the 24th International Conference on Data Engineering (ICDE'08)*. 506–515.

Lei Zou, Lei Chen, and M. Tamer Özsu. 2009. *K*-automorphism: A general framework for privacy preserving network publication. *Proceedings of the VLDB Endowment* 2, 1 (2009), 946–957.

Received October 2014; accepted December 2014