PRIVACY RISK AND DE-ANONYMIZATION IN HETEROGENEOUS INFORMATION NETWORKS

BY

ASTON ZHANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2015

Urbana, Illinois

Advisers:

Professor Carl A. Gunter
Professor Jiawei Han

ABSTRACT

Anonymized user datasets are often released for research or industry applications. As an example, *t.qq.com* released its anonymized users' profile, social interaction, and recommendation log data in KDD Cup 2012 to call for recommendation algorithms. Since the entities (users and so on) and edges (links among entities) are of multiple types, the released social network is a *heterogeneous information network*. Prior work has shown how privacy can be compromised in homogeneous information networks by the use of specific types of graph patterns. We show how the extra information derived from heterogeneity can be used to relax these assumptions. To characterize and demonstrate this added threat, we formally define privacy risk in an anonymized heterogeneous information network to identify the vulnerability in the possible way such data are released, and further present a new de-anonymization attack that exploits the vulnerability. Our attack successfully de-anonymized most individuals involved in the data. We further show that the general ideas of exploiting privacy risk and de-anonymizing heterogeneous information networks can be extended to more general graphs.

To my family, for their love and support.

ACKNOWLEDGMENTS

I would like to express my gratitude to the following people for their great support and contributions in my thesis:

- **Professor Carl A. Gunter**, for his insightful advice, constant help, invaluable encouragement and sharing of his knowledge. With his invaluable guidance, I made some achievements in my research and published a few research papers.

- **Professor Jiawei Han**, for his utmost support throughout my research. His ideas and guidance kept my research in the right heading and his enthusiasm towards research always motivated me to move forward and further, especially during difficult times.

- **Dr. Xing Xie**, a senior researcher at Microsoft Research, for his comments and help during my research.

- **Mr. Hao Fu**, a Ph.D. candidate working with Dr. Xing Xie, for his contributions in extending and generalizing the key component of this research.

This thesis was supported by HHS 90TR0003-01 (SHARPS) and NSF CNS 0964392 (NSF EBAM).

Last but not least, I would like to thank all my friends for their help and encouragement.

Without any of them, this work would not be possible.

Thank you all.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The world is getting more inter-connected. Tons of social network data are generated through people's interactions, and different entities are linked across multiple relations, forming a gigantic information-rich, inter-related and multi-typed *heterogeneous information network* [1]. Is there any risk in the current efforts to avoid privacy intrusion upon the anonymized copy of a heterogeneous information network? We start with a motivating example.

1.1 Motivating Example

Various datasets containing *micro-data*, that is, information about specific individuals, have been released for different research purposes or industry applications [2]. Some datasets contain individual profiles, preferences, or transactions, which many people consider sensitive or *private*. In the recent KDD Cup 2012, *t.qq.com* (a popular microblogging site, hereinafter referred to as *t.qq*) released its 2.3 million users' profile, social interaction, and recommendation preference log data to call for more efficient recommendation algorithms [3]. In a microblogging site like *t.qq* as depicted in Figure 1.1, entities (nodes) correspond to *users*, *tweets* or *comments*, and edges correspond to different types of links (*post*, *mention*, *retweet*, *comment*, and *follow*) among them[1]. Since both nodes and links are of multiple types, such a social network is essentially a heterogeneous information network [4]. Besides identifying information such as *user ID* which has been anonymized by randomly assigned strings, some other attributes are also replaced with meaningless IDs, such as *user tags*.

In the released anonymized *target dataset*, consider an adversary that is interested in breaching privacy of some selected target users based on their preferences.

[1]The terms *edge* and *link* are used interchangeably in this work, while the term *entity* is preferred over *node* here to reflect more realistic scenarios where each node contains multiple attributes rather than a single identifier in the settings of a heterogeneous information network.

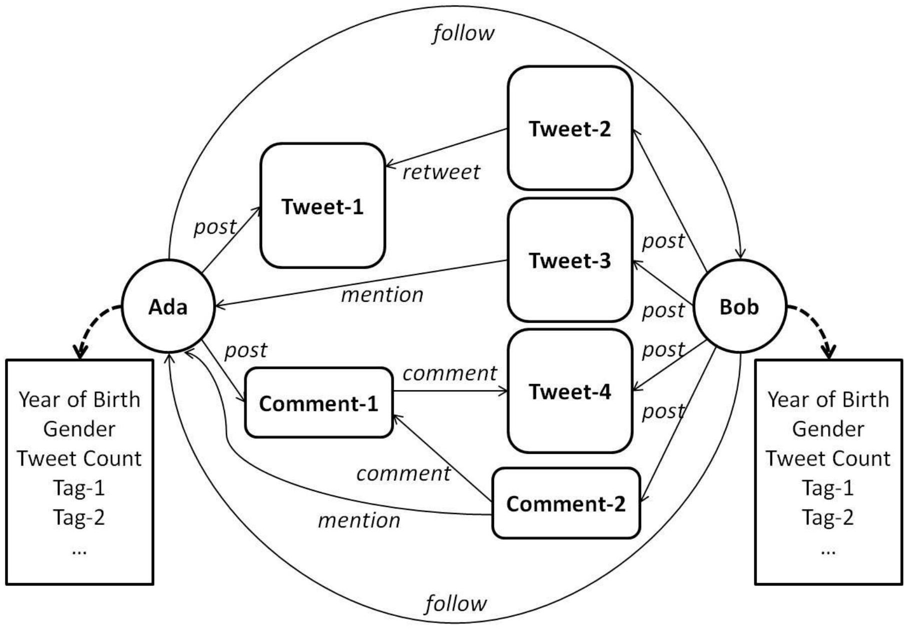


Figure 1.1: The heterogeneous information network in t.qq

The preference can be inferred from the target users' recommendation preference (acceptance/rejection) log included in the target dataset. This information is sensitive and not accessible on the *t.qq* site (the rejection log cannot be inferred from the site). Suppose the adversary obtains the non-anonymized *auxiliary dataset* from *t.qq* exactly containing the users from the same time-synchronized target dataset. To *de-anonymize* the users of interests in the target dataset, the adversary has to match the meaningless user IDs in the target dataset with the real user names in the auxiliary dataset. Given the rich information available in the heterogeneous information network as demonstrated in Figure 1.1, suppose the adversary locks his target on an anonymized user (say, *A3H*) in the target dataset who accepted the "follow Citibank" recommendation but rejected all other bank recommendations. The adversary may search in the auxiliary dataset by specifying *A3H*'s entity profile (*A3H*'s year of birth, hereinafter referred to as yob: 1980, gender: male, *etc.*) combined with *A3H*'s multiple social links (mention, retweet, comment, follow) and profile information of its neighbor entity to whom the target user connects via these links—*A3H* gave 15 comments to an anonymized female user *F8P* born in 1985 and retweeted an anonymized male user *M7R* 10 times that is born in 1970. If Ada in the non-anonymized auxiliary dataset is the only one that satisfies the matching—Ada has both the same profile information as *A3H* and Ada has the same social interactions with the other users of the same gender and age as those of *F8P* and *M7R* correspondingly; thus, the adversary successfully de-anonymizes *A3H* by establishing a *unique matching* between it in the target dataset and the real user Ada in the auxiliary dataset. Now the adversary knows Ada probably has a Citibank account or is interested in applying for it. The leak of such private information may allow scammers to spam Ada with phishing URLs camouflaged with the Citibank online-banking interface. In fact, 8% of some sampled 25 million URLs posted to microblogging sites point to phishing, malware, and scams [5].

Therefore, there is *privacy risk* in an anonymized heterogeneous information network if such unique matchings can be easily established. Users in a network of high privacy risk that can be easily de-anonymized may be vulnerable to external threats. In this work, we experimentally substantiate adversaries can exploit the privacy risk to de-anonymize over 90% users in a 1,000-user *t.qq* network of density 0.01 from a 2,320,895-user auxiliary network.

1.2 Limitations of k-Anonymity

To formalize privacy risk observed in Section 1.1, directly using the existing metric seems possible at first thought. A dataset is said to be k-anonymous if on the minimal set of attributes in the table that can be joined with external information to de-anonymize individual records, each record is indistinguishable from at least $k-1$ other records within the same dataset [6]. The larger the value of k, the better the privacy is preserved.

Consider target dataset T_{1000} that satisfies 1000-anonymity and another target dataset T_2 that satisfies 2-anonymity, together with their original non-anonymized counterparts. Imagine a new tuple t^* is created and inserted into both T_{1000} and T_2. After anonymization processes still no any other tuple in either dataset has the same value of t^*, and the new datasets are T^*_{1000} and T^*_2 respectively. Both T^*_{1000} and T^*_2 are now 1-anonymity simply because of the injection of t^*—both T^*_{1000} and T^*_2 are same vulnerable in terms of the same k-anonymity. Suppose a selective adversary is not interested in de-anonymizing t^*, then the remaining T^*_{1000} of 1000-anonymity seems much less vulnerable than the remaining T^*_2 of 2-anonymity, which may be misled by the same 1-anonymity.

Due to limitations of k-anonymity in differentiating individuals in the same target dataset, it is not suitable to formalize privacy risk in a more general scenario where adversaries may not be equally interested in de-anonymizing all users. In this work we define privacy risk in a more general sense, and prove it can be very high in the anonymized heterogeneous information network.

1.3 New Settings, New Threats

Social media are getting popular with more and more functionalities. As shown in Section 1.1, *t.qq* allows its over 500 million users to connect with one another in different ways such as follow, mention, retweet, and comment. The growing multi-typed heterogeneous information networks out of the growing social media functionalities may render the existing homogeneous information network anonymization algorithms no more effective.

Existing de-anonymization attacks on social networks made several assumptions, such as both target and auxiliary graphs are large-scale so random graphs or non-trivial cliques can be re-identified from both graphs [7, 8]. It should be highlighted that, in the new settings of a heterogeneous information network, if new attacks are feasible while relaxing these assumptions, such attacks must be addressed in the proposal of all relevant anonymization algorithms.

1.4 Our Contributions

In this work we make three unique contributions. First, we propose a definition of privacy risk tuned to the concerns of heterogenous information networks. In particular, this definition considers a more general situation where adversaries may not be equally interested in compromising all users' privacy. We show that the privacy risk can be high in an anonymized heterogeneous information network, and can be exploited in practice.

Second, we present a de-anonymization algorithm against heterogeneous information networks which exploits the identified privacy risk without requiring creating new accounts or relying on easily-detectable graph structures in a large-scale network. While central in illuminating the privacy issue for a heterogeneous information network, we also expect our algorithm to be applied to de-anonymizing a homogeneous information network (with slight performance degradation).

Our third contribution is a practical evaluation of the KDD Cup 2012 *t.qq* anonymized dataset, which contains 2.3 million users and over 60 million multiple types of social links among them. To demonstrate the effectiveness of the de-anonymization algorithm, we apply the state-of-the-art graph anonymization algorithms to the *t.qq* dataset, which were claimed effective by their designers for defending graph structural attacks. The experiments show that our algorithm is able to beat the investigated graph anonymization algorithms in the settings of a heterogeneous information network even without knowledge of the specific anonymization technique in use. It undermines the notion of "security by obscurity" for privacy preservation: ignorance of the anonymization does not prevent an adversary from de-anonymizing successfully.

Last but not least, we further theoretically and empirically show that the ideas of exploiting privacy risk and de-anonymizing heterogeneous information networks can be extended to general graphs.

CHAPTER 2

RELATED WORK

Simply replacing sensitive information with random strings cannot guarantee privacy and how to release data for different research purposes or industry applications without leaking any privacy information has been an interesting problem.

2.1 Relational Data Anonymization

Privacy preservation on relational data has been studied extensively. A major category of privacy attacks on relational data is to de-anonymize individuals by joining a released table containing sensitive information with some external tables modeling the auxiliary dataset of attackers. Table joining attacks include de-anonymization of a Massachusetts hospital discharge database by joining it with a public voter database [9] and privacy intrusions on the AOL search data [10]. To mitigate this type of attacks, k-anonymity was proposed [6]. Further enhanced techniques include l-diversity [11] and t-closeness [12].

Narayanan and Shmatikov proposed de-anonymization attacks against high-dimensional micro-data and showed success in Netflix Prize dataset [2]. They pointed out micro-data are characterized by high dimensionality and sparsity. A recent study by Narayanan *et al.* further demonstrated the feasibility of internet-scale author identification via linguistic stylometry [13]. However, all the aforementioned studies assume that an adversary utilizes *attribute information* of micro-data and can deal with relational data only.

2.2 Graph Structural Attacks

In a large-scale social network, it is hard to observe non-trivial random subgraphs or cliques [14]. Hence they easily stand out if they exist. Backstrom *et al.* discussed active attacks where adversaries create users and establish connections randomly among them and attach such random subgraphs ("sybil nodes") into the target nodes in the auxiliary graph data [7]. Since such random subgraphs can be easily detected from the anonymized counterpart of the original data, the target nodes connected to the sybil nodes are then de-anonymized by consulting the original auxiliary graph. Narayanan and Shmatikov pointed out the main drawback of this active attack is that, creating accounts, links among themselves and links to target nodes, is not feasible on a large-scale [8]. They designed an attack propagating the de-anonymization process via neighbor structure from the initial precisely-matched "seed nodes". Hence success of this attack heavily depends on if such seed nodes can be detected precisely; thus, seed nodes must stand out easily both in the target and auxiliary dataset. So non-trivial cliques are chosen [8]. Since there is no guarantee that the released anonymized network is always large, this attack is not always successful because non-trivial cliques cannot always be detectable.

2.3 Graph Data Anonymization

For graph-based social network data, the degree of nodes in a graph can reveal the identities of individuals. Liu and Terzi studied a specific graph-anonymization problem and called a graph k-degree anonymous if for every node v, there exist at least $k - 1$ other nodes in the graph with the same degree [15]. This definition of anonymity prevents de-anonymization of individuals by adversaries with a background knowledge of the degree of certain nodes.

Zhou and Pei identified a structural *neighborhood attack* and tackled it by proposing k-neighborhood anonymization [16]. They assumed an adversary may know the neighbors of the target nodes and their inter-connections. The privacy preservation goal is to protect neighborhood attacks which use neighbor structure matching to de-anonymize nodes. For a social network, suppose an adversary knows the neighbor structure for a node. If such neighbor structure has at least k isomorphic copies in the anonymized social network, then the node can be de-anonymized in the target dataset with confidence at most $1/k$ [17]. Due to its heavy isomorphism testing computation, a limitation of this attack is only distance-1 neighbors can be evaluated effectively.

Zou *et at.* assumed an attacking model where an adversary can know any subgraph that contains the targeted individual and proposed k-automorphic anonymity that the graph must has $k-1$ non-trivial automorphism and no node is mapped to itself under the $k-1$ non-trivial automorphism [18]. Wu *et al.* proposed a similar k-symmetry [19].

Cheng *et al.* identified that k-automorphism approach is insufficient for protecting link privacy and proposed the k-security anonymity [20]. In their approach, an anonymized graph satisfies k-security if for any two target individuals and any subgraphs containing either individual, the adversary cannot determine either whether a node that is linked to either target individual (NodeInfo Security) or whether both target individuals are linked by a path of a certain length (LinkInfo Security), with probability higher than $1/k$.

Although these recent graph data anonymization algorithms can be applied to social network data against graph structural attacks in Section 2.2, their applicability has not been demonstrated in the more challenging settings of a heterogeneous information network. Our evaluation in Section 6 shows that these graph data anonymization algorithms are not effective to preserve privacy of an anonymized heterogeneous information network.

CHAPTER 3

HETEROGENEOUS INFORMATION NETWORKS

In this section, we formalize the general anonymized heterogeneous information network settings that are frequently discussed in the remaining of the thesis and illustrate them with the motivating example discussed in Section 1.1.

Definition 1. *The* ***information network*** *is a directed graph $G = (V, E)$ with an entity type mapping function $\tau : V \to \mathcal{E}$ and a link type mapping function $\phi : E \to \mathcal{L}$, where each entity $v \in V$ belongs to one particular entity type $\tau(v) \in \mathcal{E}$, and each edge $e \in E$ belongs to a particular link type $\phi(e) \in \mathcal{L}$. If two edges belong to the same link type, they must share the same starting and ending entity types.*

Definition 2. *The* ***heterogeneous information network*** *is an information network where $|\mathcal{E}| > 1$ or $|\mathcal{L}| > 1$.*

A sample heterogeneous information network for the *t.qq* dataset is depicted in Figure 1.1. Given a complicated heterogeneous information network, it is necessary to provide its meta level (*i.e.*, schema-level) description for better understanding the network, and *network schema* is to describe the meta structure of a network.

Definition 3. *The* ***network schema****, denoted as $T_G = (\mathcal{E}, \mathcal{L})$, is a meta template for a heterogeneous information network $G = (V, E)$ with the entity type mapping $\tau : V \to \mathcal{E}$ and the link mapping $\phi : E \to \mathcal{L}$, which is a directed graph defined over entity types $\mathcal{E}$, with edges as links from $\mathcal{L}$.*

Figure 3.1 shows the network schema for the heterogeneous information network in Figure 1.1. In practice data publishers may not release information about all the entities and links in the original network schema while links among the same entity type (also the target entity type of adversaries' interests) are generally available either directly or indirectly via summarization over different entity types [3]. In view of this, although we believe providing richer information about

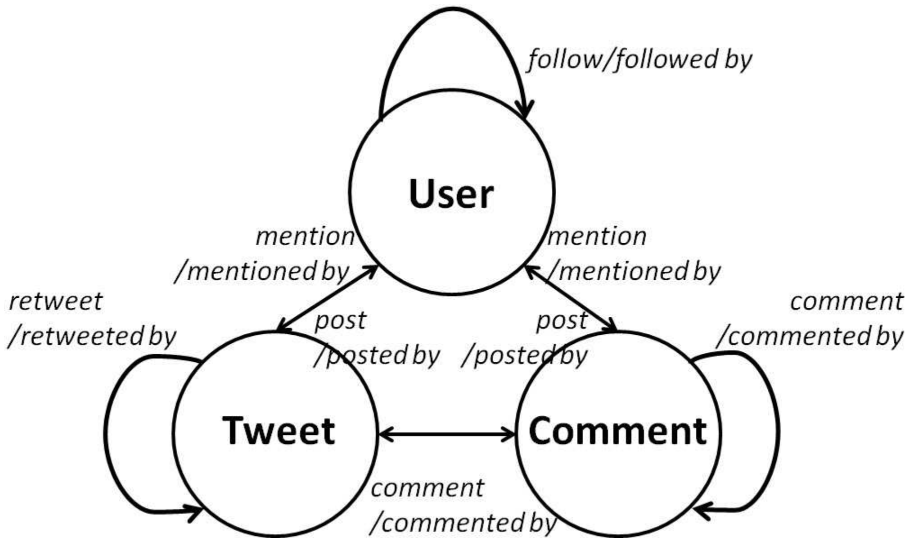


Figure 3.1: The corresponding network schema for the heterogeneous information network in Figure 1.1

multiple types of entities could further facilitate de-anonymization, in this work, we consider a more challenging and practical scenario where data publishers only provide limited information about how the same type of entity (*i.e.*, *target entity type* $\mathcal{E}^*$) can be linked via different types of links or over different types of entities. Thus, a simplified network schema is needed such that it reflects only the relationships over the target entity type.

Definition 4. *The* ***target meta paths (target network schema links)*** $\mathcal{P}(\mathcal{E}^*)$*, are paths defined on the graph of network schema* $T_G = (\mathcal{E}, \mathcal{L})$*, denoted by* $\mathcal{E}^* \xrightarrow{\mathcal{L}_1} \mathcal{E}_1 \xrightarrow{\mathcal{L}_2} \ldots \xrightarrow{\mathcal{L}_n} \mathcal{E}^*$.

Definition 5. *The* ***target network schema*** $T_G^* = (\mathcal{E}^*, \mathcal{L}^*)$ *is projected from* $T_G = (\mathcal{E}, \mathcal{L})$ *where* $\mathcal{L}^*$ *are* reproduced *or* short-circuited *from target meta paths* $\mathcal{P}(\mathcal{E}^*)$ *and target entity type* $\mathcal{E}^*$.

To illustrate, we take the released target *t.qq* dataset as an example. This anonymized dataset contains the following files and attributes (anonymized attributes are marked with underlines):

- *recommendation preference data*: user ID($\mathcal{A}$), recommended item ID($\mathcal{R}$), result (whether $\mathcal{A}$ likes $\mathcal{R}$)
- *user profile data*: user ID, yob, gender, tweet count (no. of tweets), tag IDs

- *user mention data*: <u>user ID</u>($\mathcal{A}$), <u>user ID</u>($\mathcal{B}$), the number of times $\mathcal{A}$ mentioned $\mathcal{B}$ either in $\mathcal{A}$'s tweets or comments (mention strength)
- *user retweet data*: <u>user ID</u>($\mathcal{A}$), <u>user ID</u>($\mathcal{B}$), the number of times $\mathcal{A}$ retweeted $\mathcal{B}$'s tweets (retweet strength)
- *user comment data*: <u>user ID</u>($\mathcal{A}$), <u>user ID</u>($\mathcal{B}$), the number of times $\mathcal{A}$ commented $\mathcal{B}$ either in $\mathcal{B}$'s tweets or comments (comment strength)
- *user follow data*: <u>user ID</u>(follower), <u>user ID</u>(followee)

In the above dataset, besides user entities' profile information, users' multiple social interactions are also available. Thus, the adversary can decide to project the original network schema in Figure 3.1 to only reflect relationships among his target user entity. Navigating the original network schema based on the above user mention, retweet, comment, and follow data, these target meta paths connecting users across different types of entities are possible:

- *user mention path*: $User \xrightarrow{post} Tweet \xrightarrow{mention} User$ or $User \xrightarrow{post} Comment \xrightarrow{mention} User$ (short-circuited feature: mention strength)
- *user retweet path*: $User \xrightarrow{post} Tweet \xrightarrow{retweet} Tweet \xrightarrow{posted\ by} User$ (short-circuited feature: retweet strength)
- *user comment path*: $User \xrightarrow{post} Comment \xrightarrow{comment} Tweet \xrightarrow{posted\ by} User$ or $User \xrightarrow{post} Comment \xrightarrow{comment} Comment \xrightarrow{posted\ by} User$ (short-circuited feature: comment strength)
- *user follow path*: $User \xrightarrow{follow} User$

The target meta paths allow the adversary to produce a new network schema by projecting the original network schema to a simplified one to only reflect particular few relationships over the target entity type. Specifically, the user mention, retweet and comment paths can be *short-circuited* to produce new links over users respectively while the user following path can be *reproduced* in the projection. It is also emphasized that, the target meta paths are able to greatly enrich the features (attributes) of the target entity by utilizing different *distances* of *neighbors* from the target entity along the specified meta paths. Specifically, target meta paths that are short-circuited across different types of entities and different types of links, may preserve the link heterogeneity information of the network by generating new *short-circuited feature (attribute)* and further enrich the features of the target entity. For instance, the short-circuited feature *mention strength* can be newly generated from the user mention path.

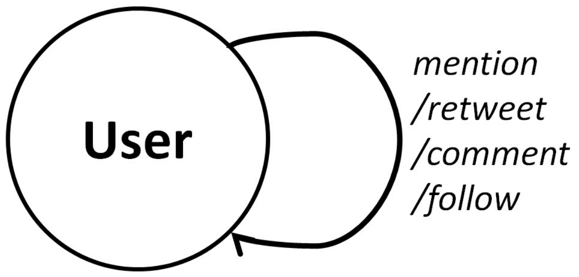


Figure 3.2: The target network schema for Figure 3.1

The target network schema for Figure 3.1 is shown in Figure 3.2. Since target meta paths may span across multiple types of entities, entity heterogeneity information is still preserved, although not fully, in target network schema only containing the target entity type.

Therefore, the de-anonymization problem in the settings of a heterogeneous information network can be formulated as follows. Detailed illustrations are provided in Section 5.

Definition 6. *The* ***de-anonymization problem in heterogeneous information network*** *is utilizing the background knowledge of the public graph $G = (V, E)$, the private graph $G' = (V', E')$, and the target network schema T_G^* to de-anonymize a target entity $v' \in V'$ by establishing matches between v' and a candidate set $C \subseteq V$ where the anonymized v''s counterpart $v \in C$. If $|C| = 1$ and the only element $v \in C$ is the correct counterpart of v', the de-anonymization is successful.*

CHAPTER 4

PRIVACY RISK ANALYSIS

Intuitively, privacy risk in a heterogeneous information network is the ease of formulating unique attribute-metapath-combined values as formalized in Section 3. Formal analysis is derived from the definition of privacy risk in general anonymized datasets.

4.1 Attribute-Metapath-Combined Values of Target Entities

Data publishers anonymize data through generalization, suppression, adding, deleting, switching edges or nodes [21][17]. Naturally, such modifications cause information loss and for a certain privacy preservation goal they should be minimized to ensure the anonymized data still satisfy the need for how they are expected to be used, *i.e.*, the need for *utility*. Generally, a certain level of utility has to be preserved for the anonymized *t.qq* dataset in order to design effective and reliable recommendation algorithms; thus, an adversary is expected to be able to compromise some sacrificed privacy due to the natural tradeoff between utility and privacy preservation [17]. In the *t.qq* dataset case, the utility is preserved in the sense that, some attribute values of user entities and most of the social interactions among different user entities are preserved (non-anonymized) as in the available target dataset descriptions in Section 3 (*e.g.*, non-anonymized attributes are not underlined).

Based on the target network schema in Figure 3.2, Figure 4.1 describes an example of how user entities are directly inter-connected via part of different types of links in the *t.qq* dataset. Here m, r, c, f stands for *mention*, *retweet*, *comment*, *follow* links in the target network schema shown in Figure 3.2.

As mentioned in Section 3, target meta paths that are short-circuited across different types of entities and different types of links preserve the link heterogeneity information of the information network and further enrich the features of the target entity. It should be noted that, following the user mention path identified in Section 3, $5m$ in Figure 4.1 from *A1X* to *U2V* indicates a new numerical feature (attribute) short-circuited from the user mention path—the mention strength from *A1X* to *U2V* in the target dataset of value 5 either through the tweet entity or comment entity. Thus, multiple meta-paths inject richer heterogeneity information for target entities in the settings of a heterogeneity information network.

If target user entities in the target dataset can form unique *attribute-metapath-combined values* across the entire network, these users can be de-anonymized from the auxiliary dataset by establishing unique matches and the dataset is not secure. To analyze the privacy risk of a heterogeneous information network, which can be intuitively considered similar to the ease of formulating unique attribute-metapath-combined values, one way is to expand the attribute dimensions of micro-data by navigating from user entities to their neighbors, neighbors' neighbors, and so on, via their multiple types of target meta paths.

With the assumption made in Section 1.1 that the target and auxiliary datasets are time-synchronized counterparts, take *A1X* in Figure 4.1 as an example. Without utilizing meta paths and only utilizing profile attribute information, the features of *A1X* are:

- Max. Distance-0: *yob, gender, ...*

After utilizing his immediate distance-1 neighbors along target meta paths, the features of *A1X* are expanded to (here "5-time-mentionee" means a mentionee mentioned 5 times by the target entity, *i.e.*, mention strength = 5):

- Max. Distance-1: *yob, gender, ..., 5-time-mentionee (U2V)'s yob, 5-time-mentionee's gender, ..., 15-time-mentionee (P3M)'s yob, 15-time-mentionee's gender, ..., 10-time-retweetee (E4G)'s yob, 10-time-retweetee's gender, ...*

Further utilizing his distance-2 neighbors (neighbors of distance-2 along target meta paths from *A1X*), the features of *A1X* are further expanded to:

- Max. Distance-2: *yob, gender, ..., 5-time-mentionee's yob, 5-time-mentionee's gender, ..., 15-time-mentionee's yob, 15-time-mentionee's gender, ..., 10-time-retweetee's yob, 10-time-retweetee's gender, ..., 10-time-retweetee's followee (B8R)'s yob, 10-time-retweetee's followee's gender, 10-time- retweetee's 1-time-mentionee (Y9Z)'s yob, 10-time-retweetee's 1-time-mentionee's gender, ...*

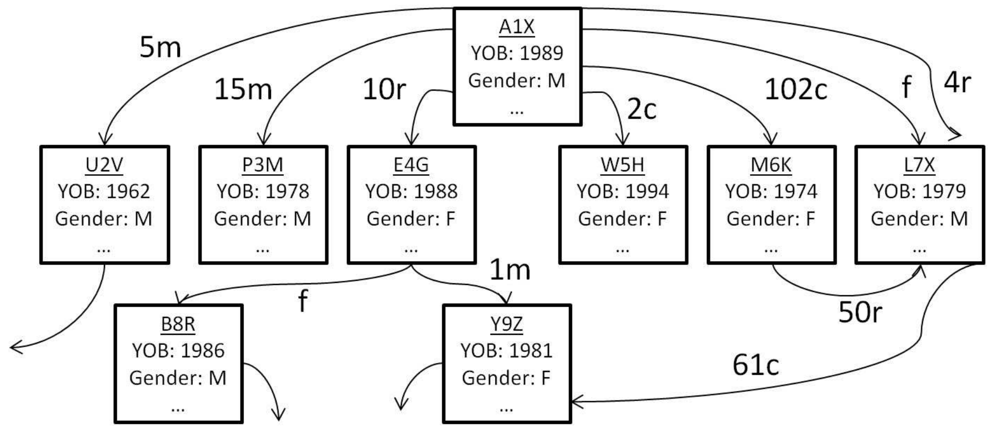


Figure 4.1: The neighbors of the target entity *A1X* are generated along target meta paths

Consistent with the idea by Narayanan and Shmatikov that large dimensions of micro-data give rise to risks of privacy [2], the expansion of dimensions by propagating via multiple types of target meta paths seems to increase the possibility for a user entity to form a unique attribute-metapath-combined value under all the expanded features across the entire dataset, which can be considered as privacy risk. In the remaining of this section, we formally prove this intuition from the observations.

4.2 Privacy Risk in General Anonymized Datasets

Privacy Risk indicates risk that privacy of a given dataset can be compromised—the higher privacy risk, the lower security and *vice versa*. Hence it might be tempting to directly adopt the notion of widely-used k-anonymity and simply reverse its value to obtain the measure of privacy risk. Here we state that, k-anonymity is not able to differentiate users from one another in terms of their different levels of security or privacy risk.

As discussed in Section 1.2, k-anonymity may be misleading in more general situations where adversaries may not be equally interested in compromising all users' privacy. To address its limitations, when quantifying risk of any user in any dataset, we consider factors that influence privacy risk both socially and mathematically.

In real life, it is highly possible that an adversary is not equally interested in compromising everyone's privacy in a dataset. As illustrated in Section 1.1, an adversary may be more motivated to de-anonymize an anonymized user who probably has a Citibank account. We denote the loss function of tuple t_i by $l(t_i)$, with values between 0 and 1. $l(t_i)$ can be considered as the potential loss of a user whose privacy is compromised given that this user does care about his loss of privacy. Therefore, in a social network, $l(t_i)$ is a certain user's privacy need because such need is positively correlated with the cost of privacy breach; hence, it is the *social factor* of a user's privacy risk.

Similar to the concept of k-anonymity, we make the same assumption that the target dataset is an anonymized copy of the same auxiliary dataset. In any given dataset T, if there are $k(t_i) - 1$ other tuples of the same value of tuple t_i, the probability that each of these $k(t_i)$ tuples, say t_i, can be de-anonymized by random guessing with probability no higher than $\frac{1}{k(t_i)}$. Therefore, the higher value of $\frac{1}{k(t_i)}$, the higher possibility that the privacy of user t_i can be compromised—hence the higher privacy risk of the user t_i. $\frac{1}{k(t_i)}$ is the *mathematical factor*. Mathematical factor can be considered positively correlated with the attack incentive as well: given the same social factor, the adversary is more motivated to de-anonymize the user with a higher mathematical factor because the potential attack precision is higher.

Combining both social and mathematical factors, we define the privacy risk of a tuple in a dataset as follows.

Definition 7. *We define the* ***privacy risk*** $\mathfrak{R}(t_i)$ *of tuple* t_i *in dataset* T *as follows:*

$$\mathfrak{R}(t_i) = \frac{l(t_i)}{k(t_i)},$$

where $k(t_i)$ *is the number of tuples in* T *with the same value of tuple* t_i*, and* $l(t_i)$ *is the loss function of tuple* t_i*.*

Averaging the risk $\mathfrak{R}(t_i)$ for each tuple t_i in dataset T, the risk $\mathfrak{R}(T)$ for dataset T is defined as follows.

Definition 8. *The* ***privacy risk*** $\mathfrak{R}(T)$ *of dataset* T *is*

$$\mathfrak{R}(T) = \frac{\sum_{i=1}^{N} \mathfrak{R}(t_i)}{N},$$

where size N *is the number of tuples* t_i *in* T*.*

It is noted that the privacy risk value $\mathfrak{R}(T) \in [0,1]$. Denoting by $\mathbb{C}(T)$ the *cardinality* of T—the number of distinct values, or distinct combined values under different attributes, describing each tuple t_i in T, we give the following lemma.

Lemma 1. *Given dataset T with the cardinality $\mathbb{C}(T)$, for each tuple t_i in T, assuming the loss function is independent of $\frac{1}{k(t_i)}$ with mean value μ, the expected privacy risk*

$$\mathbb{E}(\mathfrak{R}(T)) = \frac{\mu\mathbb{C}(T)}{N}.$$

Proof. By Definition 7 and 8,

$$\mathfrak{R}(T) = \frac{\sum_{i=1}^{N}\frac{l(t_i)}{k(t_i)}}{N}.$$

$$\begin{aligned}\mathbb{E}(\mathfrak{R}(T)) &= \frac{\sum_{i=1}^{N}\mathbb{E}(\frac{1}{k(t_i)})\mathbb{E}(l(t_i))}{N}\\ &= \frac{\sum_{i=1}^{N}\mu\mathbb{E}(\frac{1}{k(t_i)})}{N}\\ &= \frac{\mu\mathbb{E}(\sum_{i=1}^{N}\frac{1}{k(t_i)})}{N}\\ &= \frac{\mu\mathbb{E}(\mathbb{C}(T))}{N}\\ &= \frac{\mu\mathbb{C}(T)}{N}.\end{aligned}$$

□

Lemma 1 provides an estimation of dataset privacy risk in a relatively general sense. For instance, if the loss function for each tuple is a random number between 0 and 1 and independent of $\frac{1}{k(t_i)}$, the expected privacy risk of the dataset is $\frac{\mathbb{C}(T)}{2N}$. Although it may be interesting to quantify the social factor in other ways, to guarantee the highest possible privacy need from all users has been considered, in the remaining analysis we focus on the mathematical factor and set the value of every loss function $l(t_i)$ to 1. Adversaries may still have varying attack incentives in terms of different mathematical factors as discussed earlier in this section.

Theorem 1. *The privacy risk $\mathfrak{R}(T)$ of dataset T is*

$$\mathfrak{R}(T) = \frac{\mathbb{C}(T)}{N}, \quad (\mathfrak{R}(T) \in [\frac{1}{N}, 1]),$$

where in T, N is the number of tuples, and cardinality $\mathbb{C}(T)$ is the number of distinct (combined) attribute values describing tuples.

Proof. The proof can be completed by applying Lemma 1 and mathematical derivation with $l(t_i) = 1$. $\mathfrak{R}(T)$ is lowest when all the tuples are of the same value; in contrast, if every t_i has a unique value in T, $\mathfrak{R}(T) = 1$. $\square$

Back to the example of T_{1000} and T_2 in Section 1.2, suppose they are both of the same size 1000—T_{1000} has 1000 tuples of the same value while T_2 has 500 same-value tuple pairs and values from different pairs are distinct. By Definition 8, $\mathfrak{R}(T_{1000}) = 0.001$ and $\mathfrak{R}(T_2) = 0.5$ and the result is consistent with k-anonymity in terms of relative privacy risk. After inserting the unique tuple t^*, $\mathfrak{R}(T^*_{1000}) = \frac{2}{1001}$ and $\mathfrak{R}(T^*_2) = \frac{501}{1001}$, reasonably indicating T^*_{1000} is in general still much less vulnerable than T^*_2. It addresses the identified limitations of k-anonymity when adversaries may not select some users to de-anonymize in the target dataset.

4.3 Privacy Risk in Anonymized Heterogeneous Info Networks

Section 4.1 informally shows entity attribute dimensions grow fast when neighbors are utilized. It is highlighted that, rather than the exact value of privacy risk, it is the growth of privacy risk with respect to max. distances of utilized neighbors n that we focus on. Hence, given any anonymized dataset, the number of tuples N is fixed as a constant. So Theorem 1 implies that privacy risk $\mathfrak{R}(T)$ is of the same order of growth as that of the cardinality $\mathbb{C}(T)$.

Theorem 2. *For power-law distribution of the user out-degree, the lower and upper bounds for the expected heterogeneous information network cardinality grows faster than double exponentially with respect to the max. distance of utilized neighbors.*

Proof. Given a network schema $T^*_G = (\mathcal{E}^*, \mathcal{L}^*)$ projected from its original schema $T_G = (\mathcal{E}, \mathcal{L})$ and the network entity size N is ideally large enough and all possible distinct values describing $\mathcal{E}^*$ appear in T^*_G. Let $\mathcal{A}(\mathcal{E}^*)_j$ and $\mathcal{A}(L^*_i)_j$ denote the j-th attribute of the entity type $\mathcal{E}^*$ and the link type L^*_i. We assume independence among entity attributes and link types with attributes along target meta paths. To focus on the analysis of key factors that may affect the bounds of network cardinality, we also assume an entity has at most in-degree 1, the link among each pair of entities is of all types and the out-degree k of each entity follows the power-law distribution $P_K(k) = ck^{-\alpha}$, which are commonly adopted in social network analysis with $\alpha \in [2, 3]$ [14][16].

To analyze the number of distinct attribute-metapath-combined values describing $\mathcal{E}^*$, or the cardinality $\mathbb{C}(T_G^*)$, of the network schema T_G^*, we begin with the network cardinality $\mathbb{C}(T_G^*)$ without utilizing any neighbors (distance-0); it is equal to the *entity cardinality* $\mathbb{C}(\mathcal{E}^*)$, which is the actual observed number of distinct combined attribute values describing entities:

$$\mathbb{C}(T_G^*)_0 = \mathbb{C}(\mathcal{E}^*).$$

Theoretically, $\mathbb{C}(\mathcal{E}^*)$ can be as high as the product of each entity attribute's cardinality:

$$\mathbb{C}(\mathcal{E}^*) \leq \prod_{j=1}^{|\mathcal{A}(\mathcal{E}^*)|} \mathbb{C}(\mathcal{A}(\mathcal{E}^*)_j).$$

After utilizing the distance-1 neighbors from the entity, let $\mathbb{C}(L_i^*)$ denote the *homogeneous link cardinality*, which is the actual observed number of distinct combined attribute values describing the link L_i^*. Likewise, the maximum value of L_i^* is the product of each attribute cardinality of the link type L_i^*:

$$\mathbb{C}(L_i^*) \leq \prod_{j=1}^{\left|\mathcal{A}(L_i^*)\right|} \mathbb{C}(\mathcal{A}(L_i^*)_j).$$

Since entities are connected to one another via different target meta paths, *heterogeneous link cardinality* is no greater than the product of each homogeneous link cardinality:

$$\mathbb{C}(\mathcal{L}^*) \leq \prod_{i=1}^{|\mathcal{L}^*|} \mathbb{C}(L_i^*).$$

Thus, the number of distinct values that an entity can have when distance-1 neighbors are utilized is:

$$\mathbb{C}(T_G^*)_1 = \mathbb{C}(T_G^*)_0 \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*))^k.$$

By utilizing neighbors of next distance iteratively, generally when max. distance of utilized neighbors from target entities $n > 0$,

$$\mathbb{C}(T_G^*)_n = \mathbb{C}(T_G^*)_{n-1} \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}. \tag{4.1}$$

Based on the distribution function of power law for the out-degree $P_K(k) = ck^{-\alpha}$, we estimate the expected value $\mathbb{E}[\mathbb{C}(T_G^*)_n]$ of (4.1) as follows:

$$\begin{aligned}
\mathbb{E}[\mathbb{C}(T_G^*)_n] &= \mathbb{C}(T_G^*)_{n-1} \cdot \mathbb{E}[(\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}] \\
&\geq \mathbb{C}(\mathcal{E}^*) \cdot \mathbb{E}[(\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}] \\
&= \mathbb{E}[\mathbb{C}(\mathcal{E}^*) \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}] \\
&= \sum_{k=1}^{N} P_K(k) \cdot \mathbb{C}(\mathcal{E}^*) \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n} \\
&> \sum_{k=2}^{N} ck^{-\alpha} \cdot \mathbb{C}(\mathcal{E}^*) \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n} \\
&\geq \sum_{k=2}^{N} ck^{-\alpha} \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}.
\end{aligned}$$

Let $f = ck^{-\alpha} \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}, \quad k \in \mathbb{R}, \;\; 2 \leq k \leq N,$

$$\begin{aligned}
\frac{\partial f}{\partial k} &= \frac{c(\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{k^n}(nk^n ln(\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n) - \alpha)}{k^{\alpha+1}} \\
&> 0 \qquad (nk^n ln(\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n) > \alpha).
\end{aligned}$$

Hence,

$$\mathbb{E}[\mathbb{C}(T_G^*)_n] \;>\; 2^{-\alpha}(N-1)c \cdot (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{2^n}.$$

Since the vertex size N is given, the lower bound of the expected network cardinality is

$$\Omega\{\mathbb{E}[\mathbb{C}(T_G^*)_n]\} = (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{2^n}. \tag{4.2}$$

To establish the upper bound of the expected network cardinality, since $k \leq N$ and we assume N is large, solving the recursion of (4.1) we have

$$\begin{aligned}
\mathbb{C}(T_G^*)_n &\leq \mathbb{C}(\mathcal{E}^*)^{\frac{N^{n+1}-1}{N-1}} \mathbb{C}(\mathcal{L}^*)^{\frac{N^{n+1}((N-1)n+1)-N}{(N-1)^2}} \\
&\approx (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{N^n}.
\end{aligned}$$

Hence the upper bound of the expected network cardinality is the same as that of the network cardinality when all k is set to N:

$$O\{\mathbb{E}[\mathbb{C}(T_G^*)_n]\} = (\mathbb{C}(\mathcal{E}^*)\mathbb{C}(\mathcal{L}^*)^n)^{N^n}. \tag{4.3}$$

(4.2) and (4.3) complete the proof. □

Recalling the positive linear relationship between privacy risk and cardinality from Theorem 1, we obtain the following corollary.

Corollary 1. *For power-law distribution of the user out-degree, the lower and upper bounds for the expected privacy risk of a heterogeneous information network grows faster than double exponentially with respect to the max. distance of utilized neighbors.*

Corollary 1 substantiates the privacy risk growth in a heterogeneous information network as observed in Section 4.1. It should be emphasized that, it is the heterogeneity of information network links, which is in the mathematical form of $\mathbb{C}(\mathcal{L}^*)^n$, that makes both bounds even a higher order than double exponential growth.

4.4 Discussions of the Analysis

While it may be tempting to conclude that, as long as the max. distance of utilized neighbors grows infinitely, the dimensions for each entity will grow more than double exponentially until the privacy risk $\mathfrak{R}(t)$ becomes 1; it should be pointed out that it is not feasible in practice.

First, the assumption that N is large and all possible distinct values describing $\mathcal{E}^*$ appear in T_G^* may not hold. Then the observed cardinality depends on how to sample from a pool of all possible distinct values. The extreme case is that such "sampling" is so biased that each entity is assigned a value from a very small subset of the pool. However, such a "sampling" bias hardly happens because both $\mathbb{C}(\mathcal{E}^*)$ and $\mathbb{C}(\mathcal{L}^*)$ are actual observed cardinalities which are generally of reasonable sizes in practice.

Second, the assumption that in-degree is at most 1 may not hold and a large-scale information network in practice often has small average diameters [22]. For instance, in Figure 4.2, if user v_1' and user v_2' have the same attribute-metapath-combination value after utilizing their distance-1 neighbors, further utilizing their

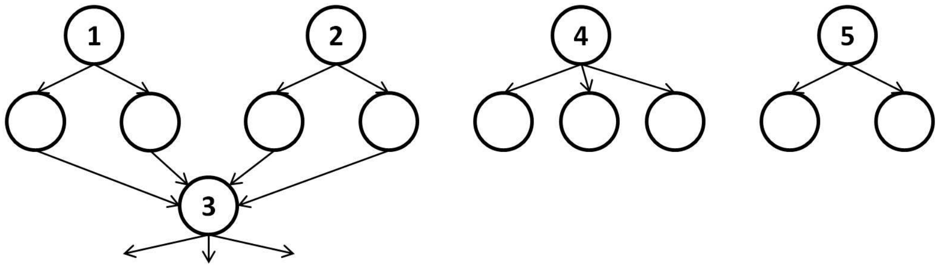


Figure 4.2: The bottleneck scenarios

longer-distance neighbors will not make them unique from each other since they will share the same neighbors of distances longer than 1. In addition, the existence of leaf nodes which do not have outgoing edges also prevents utilizing longer-distance of entity neighbors, such as user v'_4 and v'_5 in Figure 4.2. However, in Section 6 we show in practice this concern can be addressed because a slight increase of n renders the actual cardinality very close to N.

We show the empirical findings in Table 6.1 and Figure 6.1 that $\mathfrak{R}(t)$ grows very fast when $n \in \{0, 1\}$ and after $n > 1$, $\mathfrak{R}(t)$ grows towards 1 asymptotically until the bottleneck scenarios keep $\mathfrak{R}(t)$ from growing. Nonetheless, the growth order of bounds is consistent with the actual growth during $n \in \{0, 1\}$ so $\mathfrak{R}(t)$ can soon get very close to 1.

4.5 Practical Implications to Reduce Privacy Risk

To reduce privacy risk, following the two bounds established in (4.2) and (4.3), either the entity cardinality $\mathbb{C}(\mathcal{E}^*)$ or link cardinality $\mathbb{C}(\mathcal{L}^*)$ has to be reduced. Since preventing users from sharing their profile information may restrain the growth of online communities, practical efforts should focus on reducing $\mathbb{C}(\mathcal{L}^*)$ which makes both bounds grow more than double exponentially. Instead of making heterogenous types of links fully accessible from the public, online forums may only allow premium users to access all or partial types of relationships, so $\mathbb{C}(\mathcal{L}^*)$ decreases.

CHAPTER 5

DE-ANONYMIZATION ALGORITHM

To exploit the privacy risk in a heterogeneous information network as identified in Section 4, a de-anonymization algorithm is presented with a threat model.

5.1 Threat Model

In the privacy risk analysis, we assume the auxiliary dataset is exactly the non-anonymized counterpart of the target dataset. Although this assumption may hold in real attack scenarios, we consider a more challenging scenario where there is a time gap between the time data publishers release the target dataset and the time adversaries start to collect the auxiliary dataset from the web. Since a social network generally grows over time, we assume the later collected auxiliary dataset contain all the target users and links among them. Other or newly formed users and links can be included in the auxiliary dataset as well.

We emphasize that de-anonymizing with the auxiliary dataset larger than the target dataset is a non-trivial and more challenging task than both datasets are of the same size, especially when allowing certain attribute values and links to grow. First, when the auxiliary dataset becomes a superset of the target dataset without increasing the cardinality of each tuple from the target dataset, the actual risk should be lower because each tuple t_i in the target dataset has potentially more matches with users in the auxiliary dataset. Second, allowing certain attribute or link growth gives rise to potentially more candidate users in the auxiliary dataset that may match a certain target user. For instance, for a user in the target dataset that posted 3 tweets and only followed 5 users, any user in the auxiliary dataset with more than 3 tweets and more than 5 followees could be a candidate match if we consider number of tweets and number of followers grow over time. Section 6 demonstrates that the proved privacy risk can still be exploited even when the task is more challenging.

Algorithm 1: De-anonymizing entity v' in a Heterogeneous Information Network: DeHIN (G, G', T_G^*, v', n)

```
Input: $G = (V, E)$: auxiliary graph, $G' = (V', E')$: target graph,
       $T_G^* = (\mathcal{E}^*, \mathcal{L}^*)$: target network schema, $v' \in G'$: target entity, $n$:
       max. distance of utilized neighbors
Output: $C$: candidate set from the auxiliary dataset matching $v'$
begin
    $C \overset{set}{\longleftarrow} \emptyset$;
    foreach $v \in V$
        if $entity\_attribute\_match(v', v, \mathcal{E}^*)$
            if $n > 0$
                if $link\_match(n, v', v, G, G', T_G^*)$
                    $C \overset{add}{\longleftarrow} v$;
            else
                $C \overset{add}{\longleftarrow} v$;
    return $C$;
```

5.2 Algorithm

In Algorithm 1 we formulate a general de-anonymization algorithm **DeHIN** to prey upon the risk of a heterogeneous information network as identified in Section 4.

The attribute values of the target entity and the entity from the auxiliary dataset is compared by function *entity_attribute_match*. This function can be configured by users depending on different scenarios. We consider the auxiliary dataset grows from the target dataset in the threat model. So some attribute values may grow over time, such as number of tweets.

The recursive Algorithm 2 is to assist DeHIN to compare the distance-n neighbors from a target entity and an entity in the auxiliary dataset whose attributes are matched with those of the target. Likewise, function *link_attribute_match* compares the attribute values of target meta paths (links in the target network schema), if any, and is configurable. The challenge lies in how to compare the neighbors of two entities, after their own entity and link attribute values are matched. Consider the case depicted in Figure 5.1, the target entity v'_8 is matched with entity v_9 in the auxiliary dataset for function *entity_attribute_match*, and the target's neighbor v'_5 is matched with v_1 and v_2 (entity v_9's neighbors) via the same type of link for

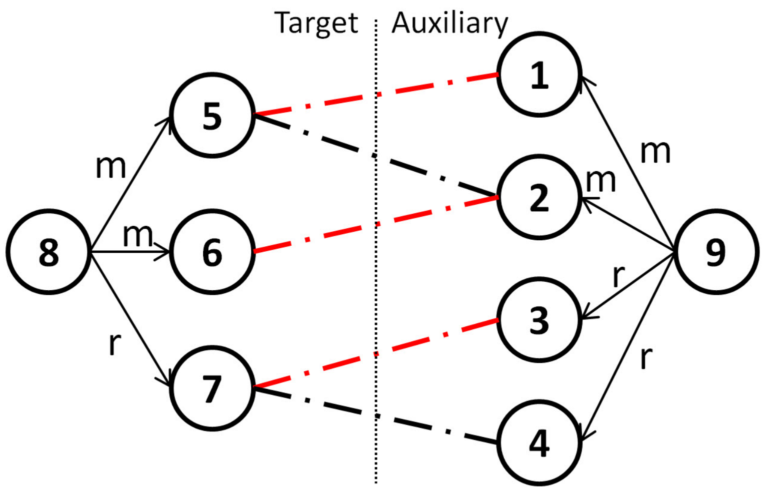


Figure 5.1: Comparing neighbors via multiple types of target network schema links from target and auxiliary datasets

the same function, v'_6 matched with v_2, v'_7 matched with v_3 and v_4. For a growing network, v_9 in the auxiliary dataset may be the "grown" target: v_9 itself matches v'_8 in profile attributes, v_9's neighbors v_1 and v_2 in fact are the non-anonymized v'_5 and v'_6, who are the neighbors of the target via the same type of link. Although v'_7 may be either v_3 or v_4 since they are matched via the same type of link, we can consider the remaining neighbor of v_9, either v_4 or v_3, to be the newly developed relationships during the time gap of the target and auxiliary datasets. Therefore, it is a maximum bipartite matching problem in graph theory (the candidate set for v'_5, $C(v'_5) = \{v_1, v_2\}$, $C(v'_6) = \{v_2\}$, $C(v'_7) = \{v_3, v_4\}$), and the most efficient Hopcroft-Karp algorithm is employed to decide whether such a maximum bipartite matching exists [23]. As long as a maximum bipartite matching exists (*e.g.*, v'_5, v'_6 and v'_7 match v_1, v_2 and v_3 respectively; or v'_5, v'_6 and v'_7 match v_1, v_2 and v_4 respectively), v_9 is considered as a candidate of v'_8. Finally DeHIN returns a *candidate set* containing all entities from the auxiliary dataset that may be the target entity. If the size of the correct candidate set is 1, a unique matching is found and the target entity is successfully de-anonymized.

It should be pointed out that, DeHIN is suitable for the general information network and is also applicable to a homogeneous information network, when it is considered as a special case of the general information network whose number of entity type and link type are 1. Besides, DeHIN does not employ isomorphism testing algorithms due to its high computational cost although we believe it can

Algorithm 2: Comparing neighbors of entities v' and v via heterogeneous links: $link\_match(n, v', v, G, G', T_G^*)$

```
Input: $n$: max. distance of utilized neighbors, $v' \in G'$: target entity, $v$: the
       entity in auxiliary graph under comparison, $G = (V, E)$: auxiliary
       graph, $G' = (V', E')$: target graph, $T_G^* = (\mathcal{E}^*, \mathcal{L}^*)$: target network
       schema
Output: $is\_match$: a boolean value
begin
    $is\_match \xleftarrow{set} true$;
    $G_B \xleftarrow{set} \emptyset$ (The bipartite graph modeling neighborhood matching);
    $\mathcal{N}_b(v', L_i^*) \xleftarrow{set} v'$'s neighbors via the link type $L_i^*$;
    $\mathcal{N}_b(v, L_i^*) \xleftarrow{set} v$'s neighbors via the link type $L_i^*$;
    foreach link type $L_i^* \in \mathcal{L}^*$
        foreach neighbor $b_i' \in \mathcal{N}_b(v', L_i^*)$
            $\emptyset \leftarrow C(b_i')$; ($C(b_i')$: candidate set for $b_i'$);
            foreach neighbor $b_i \in \mathcal{N}_b(v, L_i^*)$
                if $link\_attribute\_match(b_i', b_i)$
                    if $entity\_attribute\_match(b_i', b_i)$
                        if $n = 1$
                            $C(b_i') \xleftarrow{add} b_i$;
                        else
                            if $link\_match(n-1, v', v, G, G', T_G^*)$
                                $C(b_i') \xleftarrow{add} b_i$;
            $G_B \xleftarrow{add} C(b_i')$;
        if $max\_bipartite\_match(G_B) \neq |\mathcal{N}_b(v', L_i^*)|$
            $is\_match \xleftarrow{set} false$;
    return $is\_match$;
```

further enhance the accuracy. In the next section, we show DeHIN is effective in the settings of a heterogeneous information network even without incorporating isomorphism tests.

CHAPTER 6

EVALUATION

In this section, we evaluate the privacy risk and DeHIN performance on *t.qq* dataset. Then we show DeHIN is able to beat the investigated graph anonymization algorithms in the settings of a heterogeneous information network, while further sacrificing utility is able to defend the attack. It is also shown that DeHIN undermines the notion of "security by obscurity" for privacy preservation.

6.1 Case Study of t.qq Dataset

Following the motivating example in Section 1.1, we first evaluate the privacy risk as formalized in Section 4. Details of the anonymized KDD Cup 2012 *t.qq* dataset is depicted in Section 1.1 and Section 3. 500 target graphs of 1,000 vertices are sampled from *t.qq* dataset where vertices are randomly sampled and all the edges among them are preserved. Although a power-law out-degree distribution is assumed in the analysis (Section 4), since increasing privacy risk requires more edges to utilize different distances of neighbors from a target user, the privacy risk may vary when in reality heterogeneous information networks are of different densities:

$$density = \frac{|E|}{m\,|V|^2 + (|\mathcal{L}| - m)\,|V|\,(|V| - 1)} \tag{6.1}$$

In (6.1), $|E|$ and $|V|$ are the number of edges and vertices in the network. $|\mathcal{L}|$ indicates the total number of link types in the network and m denotes the number of link types which allow nodes to self-link. The denominator of (6.1) represents the maximum possible number of edges in the network and the value of density is always between 0 and 1.

57 of the sampled target graphs have density 0.01. The average cardinality of gender, yob, number of tweets, and number of tags for these 57 samples are 3, 87, 643, and 11 respectively. Considering the relatively small size of the target

Table 6.1: Privacy Risk of the Anonymized t.qq Dataset (density: 0.01, size: 1000) increases as the amount of utilized target network schema link types increases (in percentage)

Types of Links	max. distance 1	max. distance 2	max. distance 3
f	84.4	93.8	93.8
m	85.4	93.6	93.8
c	87.6	93.6	93.9
r	90.2	94.2	94.3
f-m	96.0	98.5	98.6
f-c	95.6	98.5	98.5
f-r	96.8	98.5	98.5
m-c	89.9	94.0	94.2
m-r	91.2	94.4	94.5
c-r	91.8	94.4	94.5
f-m-c	96.5	98.5	98.6
f-m-r	96.9	98.6	98.6
f-c-r	96.8	98.6	98.6
m-c-r	92.3	94.5	94.6
f-m-c-r	96.9	98.6	98.6

*f: follow; m: mention; r: retweet; c: comment

**Max. Distance n*: max. distance of utilized neighbors to target entities

*$n = 0$: only target entities' profiles are utilized and risk is always 1.1%

dataset, to better observe the growth of risk and variation in terms of different amounts of link types, only the number of tags is used in computing the entity cardinality $\mathbb{C}(\mathcal{E}^*)$. Results in Table 6.1 and Figure 6.1 (Figure 6.1 averages the privacy risk utilizing the same amount of link types) show that privacy risk calculated by Theorem 1 increases as the utilized heterogeneity information grows, which is the amount of target network schema link types. The drastic growth from distance 0 to 1 is consistent with the established order of growth in (4.2) and (4.3), then risk grows asymptotically towards 1 until it remains unchanged. Recall Section 4.5, the results also justify the practical efforts of reducing accessible link types is able to reduce $\mathbb{C}(\mathcal{L}^*)$ and hence privacy risk. When no link information is accessible, $n = 0$ and privacy risk is reduced efficiently given that the entity cardinality is not large as compared with the entity size.

To evaluate the performance of DeHIN proposed in Section 5 on *t.qq* dataset, the entire anonymized *t.qq* dataset is used as the auxiliary dataset while the target dataset is the sampled 500 target graphs and none of them contains cliques of size over 3. We will show DeHIN works effectively without the need to cre-

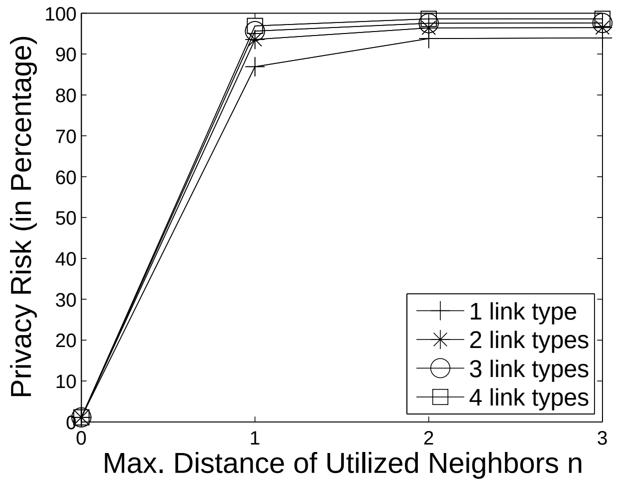


Figure 6.1: Privacy risk increases with more link types

Table 6.2: Performance of DeHIN on t.qq anonymized dataset (in percent)

Density	Max. Distance 0		Max. Distance 1		Max. Distance 2		Max. Distance 3	
	Precision	Reduction	Precision	Reduction	Precision	Reduction	Precision	Reduction
0.001	4.1	99.836	12.6	99.848	12.6	99.848	12.6	99.848
0.002	5.1	99.925	22	99.947	22.7	99.948	22.7	99.948
0.003	6.5	99.917	32.8	99.944	33.5	99.945	33.5	99.945
0.004	4.3	99.907	39.4	99.941	40.8	99.942	40.9	99.942
0.005	4.3	99.927	48.7	99.969	49.8	99.969	49.9	99.969
0.006	7	99.920	59.4	99.979	61.6	99.980	61.7	99.980
0.007	5.1	99.908	65.6	99.977	68.8	99.978	68.9	99.978
0.008	5.3	99.921	76.6	99.989	78.8	99.989	79	99.989
0.009	6.4	99.914	86.2	99.997	88.6	99.997	88.8	99.997
0.01	5.4	99.892	92.5	99.989	95.6	99.990	95.7	99.990

**Max. Distance n*: max. distance of utilized neighbors to target entities; when $n = 0$, only target entities' profile attributes are utilized

**Reduction*: Reduction Rate

ate any “sybil nodes” or to rely on easily-detectable graph structures in a large-scale network as required in the existing attacks [7, 8]. The anonymized user IDs (randomly assigned strings) in both target and auxiliary datasets are not used for attribute value matching. After DeHIN employs the remaining attribute and link information described in the motivating example (*user profile, mention, retweet, comment, follow data*) to establish the unique matching between the target user in the target dataset and a user in the auxiliary dataset, the anonymized user IDs will serve as the ground truth to decide if the unique matching is correct.

Since a social network generally grows over time, we intentionally consider attributes such as *tweet count*, *mention strength*, *retweet strength*, *comment strength* may grow between the time gap of the auxiliary and target datasets. Therefore, the attribute matching functions are configured to allow any user entity in the aux-

Table 6.3: Performance of DeHIN on t.qq anonymized dataset (density: 0.01) improves as the amount of utilized target network schema link types increases (in percent)

Types of Links	Max. Distance 1		Max. Distance 2		Max. Distance 3	
	Precision	Reduction	Precision	Reduction	Precision	Reduction
f	68.1	99.982	77.6	99.983	77.7	99.983
m	80.9	99.976	87.8	99.976	88	99.976
c	82.8	99.975	88.7	99.976	88.8	99.976
r	81.1	99.976	88.7	99.976	88.9	99.976
f-m	89.3	99.989	94.2	99.990	94.2	99.990
f-c	90.1	99.989	94.6	99.990	94.6	99.990
f-r	89.2	99.989	94.9	99.990	95	99.990
m-c	84.7	99.976	89.6	99.976	89.7	99.976
m-r	83.2	99.976	89.5	99.977	89.7	99.977
c-r	85.2	99.976	90.3	99.976	90.5	99.976
f-m-c	91.6	99.989	94.8	99.990	94.8	99.990
f-m-r	90.6	99.989	95.1	99.990	95.2	99.990
f-c-r	91.5	99.989	95.4	99.990	95.5	99.990
m-c-r	86.5	99.977	91	99.977	91.2	99.977
f-m-c-r	92.5	99.989	95.6	99.990	95.7	99.990

*f: follow; m: mention; r: retweet; c: comment
**Max. Distance n*: max. distance of utilized neighbors to target entities; when $n = 0$, only target entities' profile attributes are utilized
*$n = 0$: only target entities' profiles are utilized—precision and reduction rate are always 5.4% and 99.892%
**Reduction*: Reduction Rate

Table 6.4: Performance of DeHIN on t.qq dataset of complete graph anonymity (in percent)

Density	Max. Distance 0		Max. Distance 1		Max. Distance 2		Max. Distance 3	
	Precision	Reduction	Precision	Reduction	Precision	Reduction	Precision	Reduction
0.001	4.1	99.836	11.5	99.847	11.9	99.847	11.9	99.847
0.002	5.1	99.925	19.7	99.941	20.9	99.941	20.9	99.941
0.003	6.5	99.917	29.8	99.938	31.6	99.938	31.6	99.938
0.004	4.3	99.907	35.8	99.936	38.3	99.936	38.4	99.936
0.005	4.3	99.927	44.1	99.963	47.1	99.963	47.1	99.963
0.006	7	99.921	54.3	99.973	57.8	99.973	57.9	99.973
0.007	5.1	99.908	59.5	99.971	64.2	99.971	64.2	99.971
0.008	5.3	99.921	70.3	99.978	74.8	99.978	74.8	99.978
0.009	6.4	99.914	78.1	99.985	83.4	99.986	83.5	99.986
0.01	5.4	99.892	84.4	99.976	89.8	99.976	89.8	99.976

**Max. Distance n*: max. distance of utilized neighbors to target entities; when $n = 0$, only target entities' profile attributes are utilized
**Reduction*: Reduction Rate

iliary dataset with values of these attributes greater than or equal to those of the target user to be a candidate. Likewise, we also intentionally consider links may be newly formed in the auxiliary dataset for link matching. These considerations make the de-anonymization scenario more practical and more challenging since they will potentially introduce more candidates comparing with the exact attribute or link value matching.

The entire auxiliary dataset contains 2,320,895 user entities. With random guessing, the adversary may de-anonymize a user from the target dataset with probability no higher than $\frac{1}{2{,}320{,}895}$. If the candidate size can be reduced to 100 in-

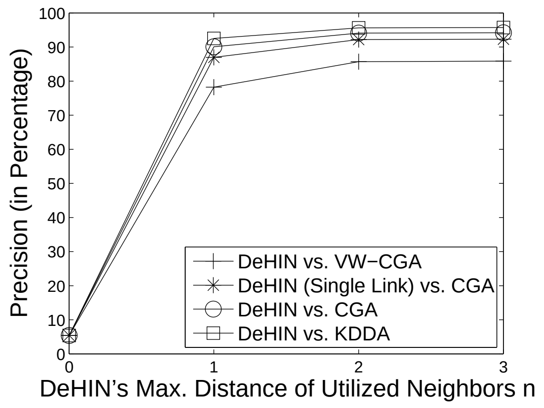


Figure 6.2: DeHIN Precision Improves with More Link Types

cluding the target, the random guessing may be correct with a drastically increased chance of $\frac{1}{100}$. If the candidate size is exactly 1 and such a unique matching is correct, the de-anonymization is successful. Hence, we define two metrics for the experiments:

$$Precision = \frac{\sum_{i=1}^{|V'|} s(v_i')}{|V'|},$$

$$Reduction\ Rate = \frac{1}{|V'|}\sum_{i=1}^{|V'|}(1 - \frac{|C(v_i')|}{|V|}),$$

where $|V'|$ and $|V|$ are the size of the target and auxiliary dataset, $s = 1$ if $v_i' \in V'$ is successfully de-anonymized, otherwise $s = 0$, and $|C(v_i')|$ is the size of candidate set for the target v_i'.

The performance of DeHIN on target datasets of different densities is shown in Table 6.2. Clearly, the general performance improves as the density of the target dataset increases because higher density indicates DeHIN may be able to utilize more neighbors to expand the dimensions of each target user to achieve unique matchings. It reveals an important problem that, if a group of people have rich social connections, they may have higher social values and may cause adversaries' attention; however, their privacy can be compromised more easily. Generally, the reduction rate looks promising as compared with the original candidate size of 2.3 million; so even when precision is relatively low on a low-density network, high reduction rate makes manual investigation of matched candidates possibly practical. For a certain density level, precision increases drastically when distance-1 neighbors are utilized, particularly for a higher-density network where there may be more neighbors. Due to the bottleneck scenarios discussed in Section 4.3 and Figure 4.2, the performance improves much more slowly or remains unchanged when DeHIN utilizes neighbors of longer distances.

To evaluate whether the heterogeneity of an information network improves the performance, we selectively employ different types of links in DeHIN and gradually increase the number of links in de-anonymizing the target dataset with potentially a higher social value (density = 0.01). The results in Table 6.3 and Figure 6.2 (Figure 6.2 averages the precision of DeHIN utilizing the same amount of link types) justifies that the performance improves as the utilized heterogeneity information grows, which is the amount of target network schema link types. Moreover, the observed growth trend is consistent to that of privacy risk in Figure 6.1.

6.2 Beating Complete Graph Anonymity

The utility of *t.qq* dataset has to be preserved to a certain level to ensure effective recommendation algorithms can be designed. We now lower their utility and apply the state-of-the-art graph anonymization algorithms in Section 2.3 on *t.qq* dataset. Since adding edges to link all the users will make the entire network safer from all the structural attacks as identified in the work of k-degree, k-neighborhood, k-symmetry, k-automorphism, and k-security, to ensure the best case of defence, we formulate *complete graphs* under different types of links. *Complete Graph Anonymity* can be considered as one of the best case for the investigated graph anonymization algorithms. For instance, when the graph becomes a complete graph after fake links are added, the k turns to be the largest possible value, which is the number of vertices in the graph, for anonymization like k-degree, k-neighborhood, *etc,* as surveyed in Section 2.3. To be consistent with these original algorithms that do not consider short-circuited features and to preserve certain utility, we set short-circuited attribute values to be the same random number and keep the existing short-circuited attribute values.

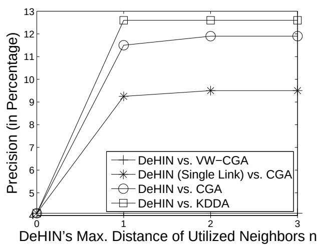
To address the enhanced anonymity, DeHIN is now re-configured to remove all the links with the majority short-circuited attribute value in the entire network before taking effect. Since a social network is generally of density lower than 0.5, it can almost be ensured that all the newly added fake links will be removed from the target dataset. However, this step will mistakenly remove the real links that have the same short-circuited attribute values as the fake links from the target dataset and $\mathbb{C}(\mathcal{L}^*)$ decreases in (4.2) and (4.3); thus the performance of DeHIN degrades slightly as shown in Table 6.4 and Figure 6.3(a)—Figure 6.3(j). In Figure 6.3(a)—Figure 6.3(j), complete graph anonymity is able to lower the attack precision effectively when DeHIN only utilizes a single homogeneous link. However, DeHIN still poses great threats to complete graph anonymity, when heterogeneous links are fully utilized.

6.3 Defending DeHIN by Sacrificing Utility

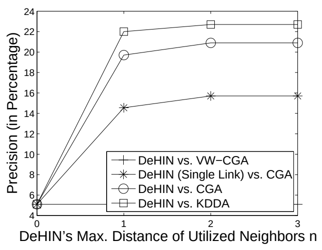
To enhance preserved privacy against DeHIN, we have to further lower the utility of the target dataset by assigning randomly generated varying weights to the short-circuited attributes of each newly added fake links. It can be observed from Figure 6.3(a)—Figure 6.3(j) that this *Varying Weight Complete Graph Anonymity* renders DeHIN ineffective when utilizing neighbors because most faked links are still preserved in the target dataset and n is clear to 0 in (4.2) and (4.3). However, varying weight values in the fake links cause much higher information loss than assigning the same values; thus the anonymized data utility is sacrificed much more.

6.4 "Security by Obscurity"?

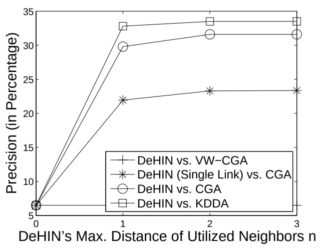
While DeHIN can be launched successfully against certain anonymization (*e.g.*, DeHIN v.s. KDD Cup Original anonymization), it may be (slightly) less effective against other anonymizations (*e.g.*, complete graph anonymity) even when it is re-configured as in Section 6.2. Researchers might be tempted to suggest that, because the adversary might not know what anonymity is employed, he might not be able to launch an attack. Here, we hope to dispel this notion. Suppose an adversary always uses the re-configured DeHIN in Section 6.2, the performance on the original *t.qq* anonymization will be exactly the same as that of complete graph anonymity because likewise only the real edges of the same majority attribute values will be affected during de-anonymization. Since DeHIN still poses great threats, this is an extremely important indication that privacy preservation requires more attention from researchers.



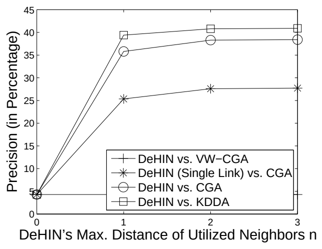
(a) Density: 0.001



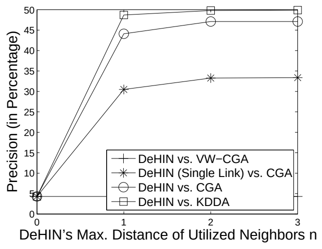
(b) Density: 0.002



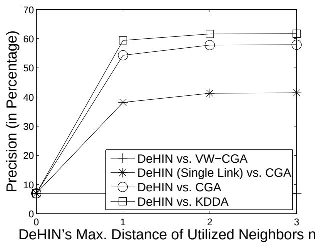
(c) Density: 0.003



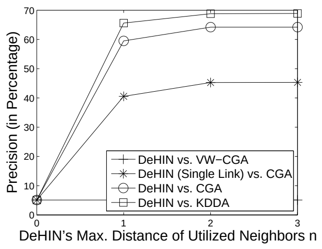
(d) Density: 0.004



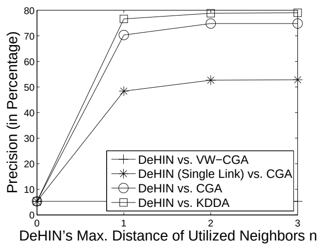
(e) Density: 0.005



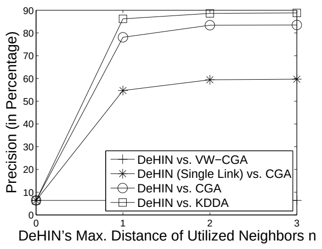
(f) Density: 0.006



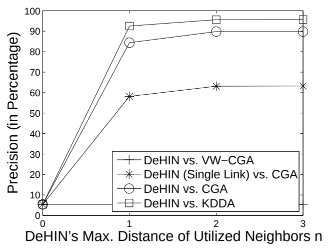
(g) Density: 0.007



(h) Density: 0.008



(i) Density: 0.009



(j) Density: 0.01

Figure 6.3: Precision of DeHIN against different anonymized heterogeneous information networks of different densities (CGA: Complete Graph Anonymity; VW-CGA: Varying Weight Complete Graph Anonymity; KDDA: KDD Cup 2012 t.qq Original Anony-mization)

CHAPTER 7

EXTENSION TO GENERAL GRAPHS

We have demonstrated that privacy risk can be exploited theoretically and empirically in anonymized heterogeneous information networks. A natural question is, are we safe in general graphs that are not heterogeneous? To address this question, we further extend our theoretical and empirical analysis to general graphs. We categorize general graphs into two broad classes: simple graphs and rich graphs. Specifically, rich graphs contain rich information on nodes and links but are not necessarily heterogeneous information networks or directed; thus, rich graphs are still more general than heterogeneous information networks.

We start with new concepts, definitions, and notations in the technical discussions.

7.1 Preliminaries

Social Graphs Two models for social networks are described in [24]. 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 $\{\text{Alice}, \text{female}, 1990\}$. If user i marks user j as a "friend" and has sent the friend 5 messages, the edge attributes $Y(i, j)$ can be represented as $\{\text{friend}, 5\}$.

Node Similarity There are a number of works on graph node similarity for purposes other than de-anonymization in literature. Blondel *et al.*, [25] proposed a graph node similarity measurement which is calculated iteratively by summing up the similarity scores between the neighbors of two nodes. The "SimRank" proposed by Jeh and Widom [26] compares nodes in the same graph, so it is infeasible for de-anonymization (in which two graphs are compared). Melink *et al.* [27] 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 above two measurements. In our initial experiments, we evaluated the above measurements, and found empirically that they are not suitable for de-anonymization (Section 7.2.2), so we decided to find a new node similarity measurement that is effective in de-anonymization.

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, *e.g.*, 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 nearby 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 do 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 de-anonymization 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.

7.2 De-anonymizing 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 sub-isomorphic to the other. In theory, the graph can be perfectly de-anonymized by solving the subgraph isomorphism problem. However, as the subgraph isomorphism problem is known to be NP-complete, this approach is infeasible for large-scale graphs. Once both edge addition and deletion are involved, the problem becomes complicated and the sub-isomorphism no longer holds. The basic assumption in this paper 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 de-anonymization. Finally, we discuss the relation between node similarity and privacy risks. Different from [28], no initial seed mappings are required in our approach.

7.2.1 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 de-anonymization, 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 7.1.

2. Two nodes are as similar as their neighbors are. This intuitive property is commonly assumed by all the node similarity measures we investigated above.

Based on the above 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 effort to maximum the overall similarity score. The similarity score $\mathcal{S}(i,j)$ is then assigned with such an overall score. Because the similarity score is calculated by matching neighbors, we refer to the de-anonymization algorithm as `NeighborMatch` (see Algorithm 3 for pseudocode).

Algorithm 3: The `NeighborMatch` algorithm

```
Input  : Auxiliary graph $G_1$, target graph $G_2$, number of expected
         mappings $M$
Output: Identity disclosres in the form of 1-1 mappings
foreach $(i,j) \in V_1 \times V_2$
    $\mathcal{S}^{(0)}(i,j) \leftarrow 1$
$k \leftarrow 1$
repeat
    foreach $(i,j) \in V_1 \times V_2$
        $B_{i,j} \leftarrow (N_1(i), N_2(j), N_1(i) \times N_2(j))$ where $w(i',j') = \mathcal{S}^{(k)}(i',j')$
        $\theta_{i,j} \leftarrow \texttt{MaxMatch}(B_{i,j})$
        $\mathcal{S}^{(k+1)}(i,j) \leftarrow \sum_{l \in N_1(i)} \mathcal{S}^{(k)}(l, \theta_{i,j}(l))$
    $k \leftarrow k+1$
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 $\theta$ with largest scores)
```

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 $B_{i,j} = (N_1(i), N_2(j), N_1(i) \times N_2(j))$, 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 $B_{i,j}$. We denote the match as a bijection $\theta_{i,j}$, where node $l \in N_1(i)$ is matched to node $\theta_{i,j}(l) \in N_2(j)$. Finally, $\mathcal{S}^{(k+1)}(i,j)$ is calculated as the sum of matched neighbors' similarity scores:

$$\mathcal{S}^{(k+1)}(i,j) = \sum_{l \in N_1(i)} \mathcal{S}^{(k)}(l, \theta_{i,j}(l)) \tag{7.1}$$

where the match $\theta_{i,j}$ is re-calculated in every iteration with the node similarity scores $\mathcal{S}^{(k)}$. The above procedures are repeated until the normalized scores converge, and the normalization is done by dividing $\mathcal{S}^{(k)}$ by the maximum $\mathcal{S}^{(k)}(i,j)$. As the diameter of a social graph is usually small, we found it unnecessary to repeat the iteration that many times, so we limited the number of iterations to 5 in experiments.

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 which 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 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. 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 that expected in our algorithm.

7.2.2 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, *i.e.*, 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 *self-similarity score* $\mathcal{T}(i) = \mathcal{S}(i, i)$. We study the property of $\mathcal{T}(i)$ and find:

Theorem 3. *A node is always among the most similar candidates to itself,* i.e.*,* $\mathcal{T}^{(k)}(i) \geq \mathcal{S}^{(k)}(i, j)$ *for any nodes* $i \in V_1, j \in V_2$ *in* k^{th} *iteration.*

Proof. Theorem 3 is proved by induction. For $k = 0$, every $\mathcal{S}^{(0)}(i, j)$ equals 1, so 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 $B_{i,j}$ in the $(k+1)^{\text{th}}$ iteration. The assumption indicates edge (l, l) has the maximum weight among $\{(l, l')|l' \in N_2(j)\}$, so one of the optimal matches for $\mathcal{T}^{(k)}(i)$ is $\theta_{i,i}(l) = l$. As all optimal matches have the same overall scores, the updated self-similarity score is

$$\begin{aligned} \mathcal{T}^{(k+1)}(i) &= \sum_{l \in N_1(i)} \mathcal{S}^{(k)}(l, l) \\ &= \sum_{l \in N_1(i)} \mathcal{T}^{(k)}(l) \end{aligned} \tag{7.2}$$

We can then devise

$$\begin{aligned} \mathcal{S}^{(k+1)}(i, j) &= \sum_{l \in N_1(i)} \mathcal{S}^{(k)}(l, \theta_{i,j}(l)) \\ &\leq \sum_{l \in N_1(i)} \mathcal{T}^{(k)}(l) \\ &= \mathcal{T}^{(k+1)}(i) \end{aligned}$$

□

Theorem 3 shows that $\mathcal{S}(i, j)$ is bounded by $\mathcal{T}(i)$. The following theorem further explains what $\mathcal{T}$ really is.

Theorem 4. *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 (7.2) 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 which 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. □

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 3 and 4 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. Above discussions motivate a possible anonymization approach against our attack which 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 de-anonymization in Section 7.2.1. The first property is proven to be satisfied directly by Theorem 3, and apparently the second property is also satisfied. For the node similarity measurements mentioned in Section 7.1, 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 de-anonymization. For example, the node similarity proposed by Blondel *et al.* [25] can only identify less than 2% nodes in a co-author graph from Microsoft Academic Search, once the graph is modified by any of the anonymization algorithms listed in Section 8.1.2. Only when the graph structure is not modified at all (naïve anonymization), the measurement could identify about 45% nodes. We also obtained similar result for "Similarity Flooding" [27].

7.3 De-anonymizing Rich Graphs

In order to incorporate graph structure and attributes for de-anonymization, we extend the algorithm described in Section 7.2 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.

7.3.1 Attribute Similarity

The *node-attribute similarity* $\mathcal{S}_X(i,j)$ represents 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 both measurements range from 0 to 1 inclusively. Determining the exact definition of similarity measurement requires the comprehension and domain knowledge of the attributes. Therefore, we do not make any assumption on how the attributes are measured or combined but only specify that two attribute sets are more similar if the corresponding similarity score is larger, *e.g.*, value 0 indicates completely different and 1 suggests possible equivalence.

In directed graphs, there could be two edges of opposite directions between two 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) is 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. Again, determining the exact definition of relation similarity is non-trivial, so we only assume that it ranges from 0 to 1 and relies on edge-attribute similarity.

Basically, we try to utilize the attribute information of graphs by introducing the above similarity measurements. $\mathcal{S}_X$ and $\mathcal{S}_Y$ are supposed to measure similarity of nodes and edges respectively, 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.

7.3.2 Generalized Node Similarity

Given 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 $j \in V_2$, which utilizes both structural and attribute information. 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 order to calculate $\mathcal{S}(i, j)$, a bipartite graph $B_{i,j} = (N_1(i), N_2(j), N_1(i) \times N_2(j))$ is constructed by weighting edge $(i', j') \in N_1(i) \times N_2(j)$ as $\mathcal{S}(i', j')\mathcal{S}_R(i, i', j, j')$. Denoting the maximum matching as $\theta_{i,j}$, the generalized node similarity is calculated iteratively as

$$\mathcal{S}^{(k+1)}(i,j) = \alpha \cdot \sum_{l \in N_1(i)} \mathcal{S}^{(k)}(l, \theta_{i,j}(l)) \cdot \mathcal{S}_R(i,l,j,\theta_{i,j}(l)) + \mathcal{S}_X(i,j) \tag{7.3}$$

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 7.2.1. 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 7.2.2, that Theorem 3 still holds in this case, so the generalized node similarity score still guarantees that a node is always one of the most similar candidates to itself in this case.

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 which 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 their similarity scores only. 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, clustering coefficients [29], centralities [30], and recursive structural features proposed in [31]. For rich graphs, additional attribute information can be combined with structural features to obtain $\mathcal{C}$. Denoting d_c as the upper bound of the node degree that is counted only with respect to the nodes in $\mathcal{C}$, the running time of a single iteration is reduced to $O(Kd_c^2 \log d_c)$, and the space requirement for similarity scores is reduced to $O(K)$.

CHAPTER 8

FURTHER EVALUATION ON GENERAL GRAPHS

In this section, we start by introducing experiment settings like 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 eigenvector centrality.

8.1 Experiment Settings

8.1.1 Dataset

We used four public datasets for evaluation: a co-author 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 (of 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 at least one paper.

The `lj` dataset was analyzed in [32] 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, from which we randomly extracted a subgraph for our evaluation, and it contains 10,000 nodes and

72,831 edges.

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 and 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 naïvely anonymized, *i.e.*, all keywords 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.

8.1.2 Anonymization

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

Naïve Anonymization The naïve 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 de-anonymization result for this approach.

k**-degree Anonymity**(k) The k-degree anonymity requires that for every node i in the anonymized graph, 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 [33]: 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", ***i.e.***, (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 $p|E|$ edge additions and $p|E|$ edge deletions.

Spectrum Preserving(p) Ying and Wu [34] proposed two spectrum preserving randomization algorithms which can be viewed as variants of random perturbation and switching, and they were shown empirically to keep better data utility. We implemented the two algorithms and used parameter p to control the number of modifications as the above.

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.

8.1.3 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.

8.1.4 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 3), 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.

8.2 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 by Narayanan et al. [35, 28]. Empirical results about the relation between privacy risks and eigenvector centrality were also reported.

8.2.1 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

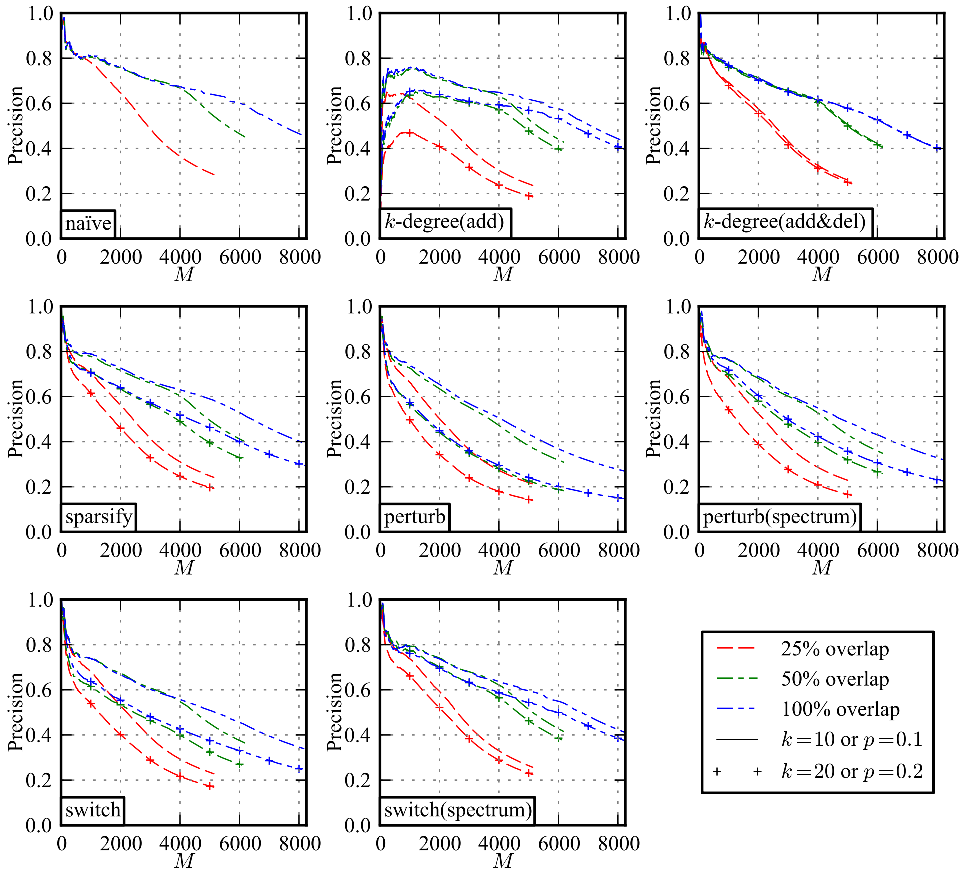


Figure 8.1: Precision of attacks with different M (the number of mappings) and node overlap on `msas` graph

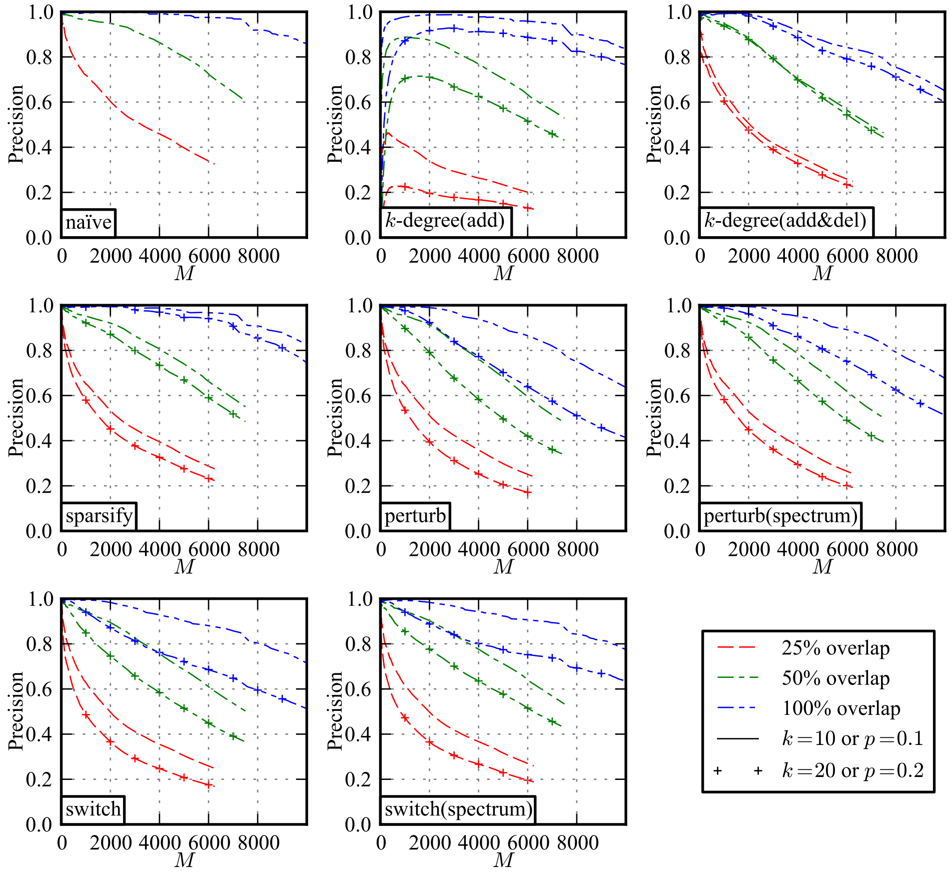


Figure 8.2: Precision of attacks with different M (the number of mappings) and node overlap on `lj` graph

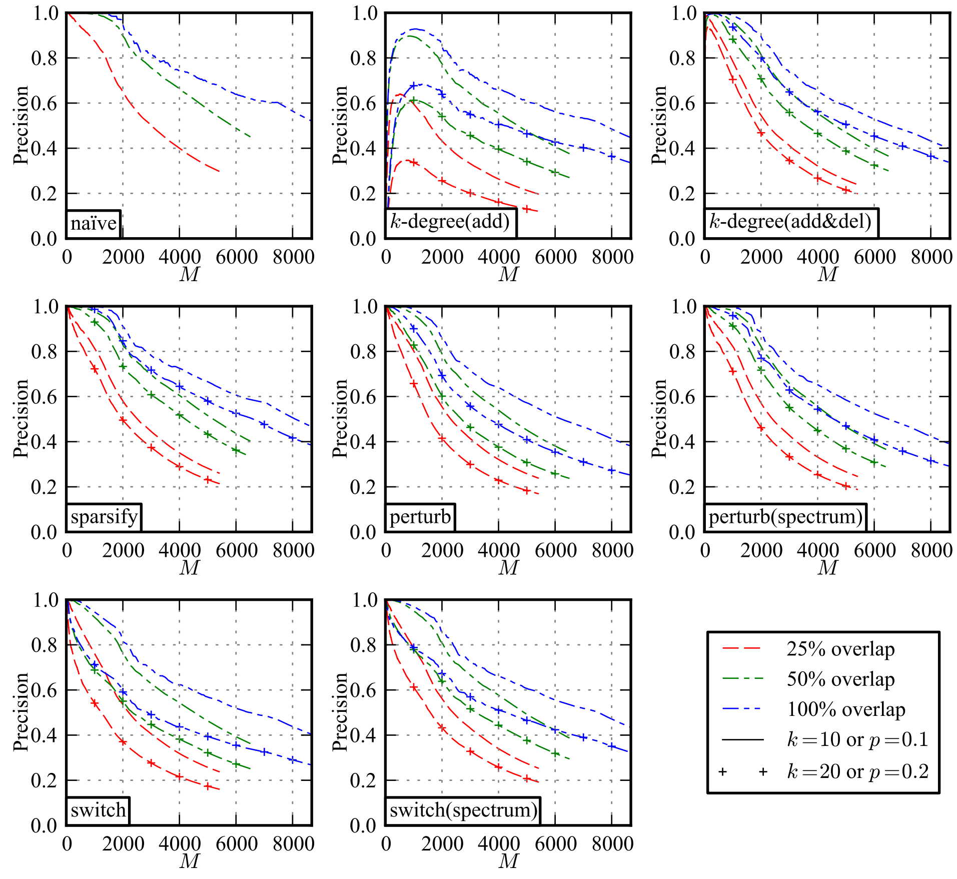


Figure 8.3: Precision of attacks with different M (the number of mappings) and node overlap on `enron` graph

pairs with different overlap (25%, 50%, 100%). Although the auxiliary graph does not have to be equal sized with the target graph in a practical attack, we took the case where the two graphs were equal sized for illustration, *i.e.*, 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 was sanitized with different anonymization algorithms. The results (Figure 8.1, 8.2, and 8.3) showed that the precision and the recall (see Section 8.1.4) 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 identified with a probability of at most $1/k$. However, with extra structure knowledge, such probability 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 outputted 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, *e.g.*, about half of the first 100 mappings corresponded to the top 100 degree nodes. As node degree in a real-world graph is usually skewed [36], 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%. Therefore, the neighborhood of the top degree nodes were greatly changed and they were difficult to be identified.

For a given randomization parameter p, the sparsification seemed to be the easiest one to be attacked, 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 perturbation, it provided less protection. We believed it is due to 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 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

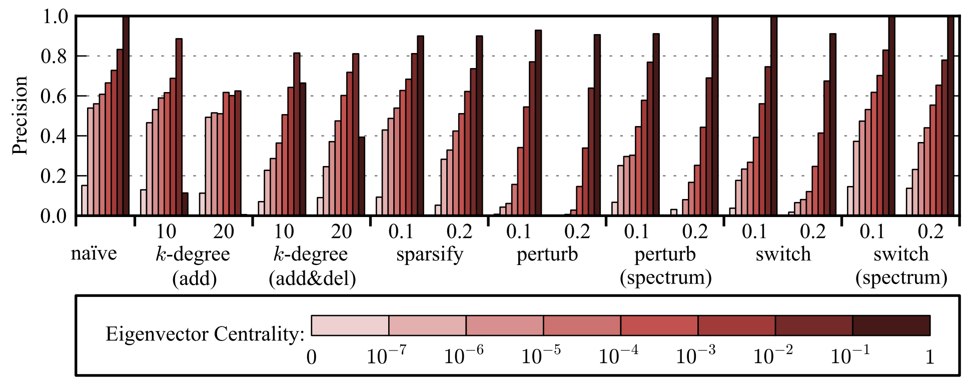


Figure 8.4: Precision of identity disclosure grouped by the auxiliary graph's normalized eigenvector centrality (100% overlap, msas). Results for other graphs were similar.

the result showed that the nodes could still be identified with high precision even if 20% of the edges were modified.

8.2.2 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, and were thus omitted here.

8.2.3 Eigenvector Centrality

Bonacich [30] proposed a family of centrality measurements, which 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 eigenvector centrality and calculated the average de-anonymization 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 8.4) showed that important nodes (with high eigenvector centrality) were more likely to be identified.

We studied the distribution of eigenvector centrality of the `msas`, `lj`, 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 7.2.2, 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 identified.

8.2.4 Efficiency

We implemented our algorithm in C++ with multi-threading. The experiments were performed on a server equipped with an Intel Xeon X5660 CPU (6 cores, 2.8 GHz), and 2 GB 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 7.2.1, we limited the number of iterations to 5. Taking the experiment on the `lj` graph presented in Section 8.2.1 for example, where the auxiliary and 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.

8.2.5 Comparison

Narayanan and Shmatikov [28] proposed a de-anonymization algorithm (denoted as `NS`) which requires a certain number of *seed mappings* to invoke large-scale re-identification. Collecting seed mappings can be viewed as a kind of small-scale de-anonymization 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, *i.e.*, 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%, to 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 8.5) showed that for `NS`, both precision and recall increased as the quality of seed mappings increased. However, its recall was low even 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 such 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 the 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. [35] 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 ($n \leq 100$), since the top degree nodes of the both graphs were observed to 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 the 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 8.1.2, the result showed that 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, while the highest accuracy of the `SA` algorithm, which turned out to be obtained against the

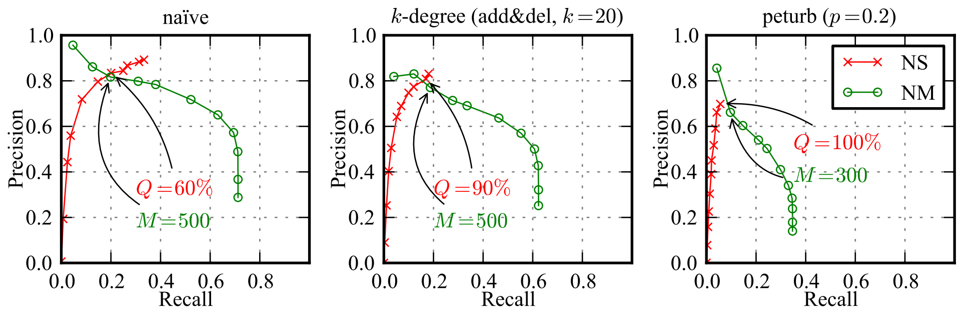


Figure 8.5: Performance of NeighborMatch(NM) and NS attacks (25% overlap, msas). The curves for NM and NS attacks are generated by varying the number of mappings (M = 100, 300, 500, 800, 1000, 1500, 2000, 2500, 3000, 4000, 5000) and the quality of seed mappings (Q = 0%, 10%, ..., 100%) respectively. Note that the curves for NS attacks are not standard precision/recall curves due the different parameters for different data points. Results for other anonymization algorithms and datasets are similar and thus omitted.

naïve anonymization, was 32%, 84%, and 60%. For the k-degree anonymization algorithm which only adds edges, both of the algorithms performed poorly on the top 20 nodes ($\leq$ 8%), and this was expected as discussed in Section 8.2.1. We also tried other values for n and 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 varies from 2% to 76% depending on datasets and anonymization algorithms, but the recall was always lower than 10%.

8.3 Rich Graphs

We further explored the performance of our algorithm on rich graphs. As de-anonymizing the actual `tweibo` graph was infeasible, we simulated the process of auxiliary graph collection and de-anonymization instead.

8.3.1 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 by $r(x, y) = \min\{x, y\}/\max\{x, y\}$. The node-attribute similarity score $\mathcal{S}_X$ 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 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 $\mathcal{S}_{Y2} = \mathcal{S}_Y(j_1, i_1, j_2, i_2)$. $\mathcal{S}_R$ was then taken as the average of $\mathcal{S}_{Y1}$ and $\mathcal{S}_{Y2}$, if both edge pairs were presented. Otherwise, the pair with a missing edge was omitted, or zero if too many edges were missing.

8.3.2 Parameters

We kept the top $K = 10^7$ candidate pairs (see Section 7.3.2) to reduce running time and space requirement. In order to determine the proper parameter α (see Equation (7.3)), we carried out an experiment over different choices of α to find the best α in the sense 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 showed that any value greater than 10^{-4} would decrease the overall precision. It turned out that a smaller value would not decrease the overall precision significantly, but it disregarded the importance of node-attributes, so we took $\alpha = 10^{-4}$ as a modest choice.

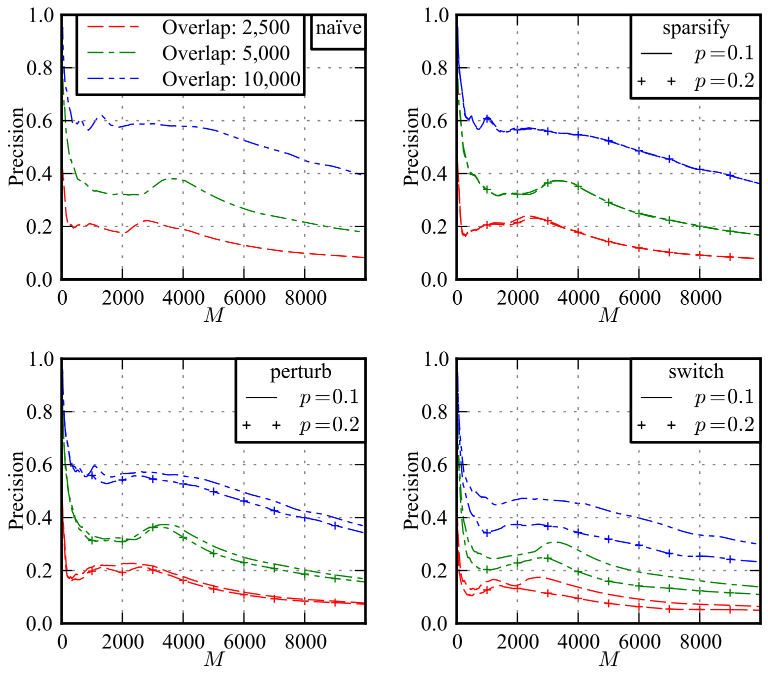


Figure 8.6: Attack precision with different node overlap (tweibo, $K = 10^7$)

8.3.3 Evaluation

For rich graphs, we were interested in evaluating the impact of altering graph structure and attributes on the de-anonymization performance. We modified the graph structure by applying anonymization algorithms described in Section 8.1.2. 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 [28]. 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 $(1-q)^2/(1-q^2) = 1-p$.

Considering the node overlap is between only 0.1% and 0.4%, the result (Figure 8.6) showed reasonable precision and recall in de-anonymizing rich graphs. Modifying the graph structure did not bring much difficulty to de-anonymization compared with the naïve approach, and different randomization strategies did not differ much in terms of precision. Further result (Figure 8.7) showed that attributes played important roles in de-anonymization and attribute perturbation decreased the precision to a considerable extent. As the attributes of nodes and edges provided meaningful information to distinguish them, it was expected that attacking

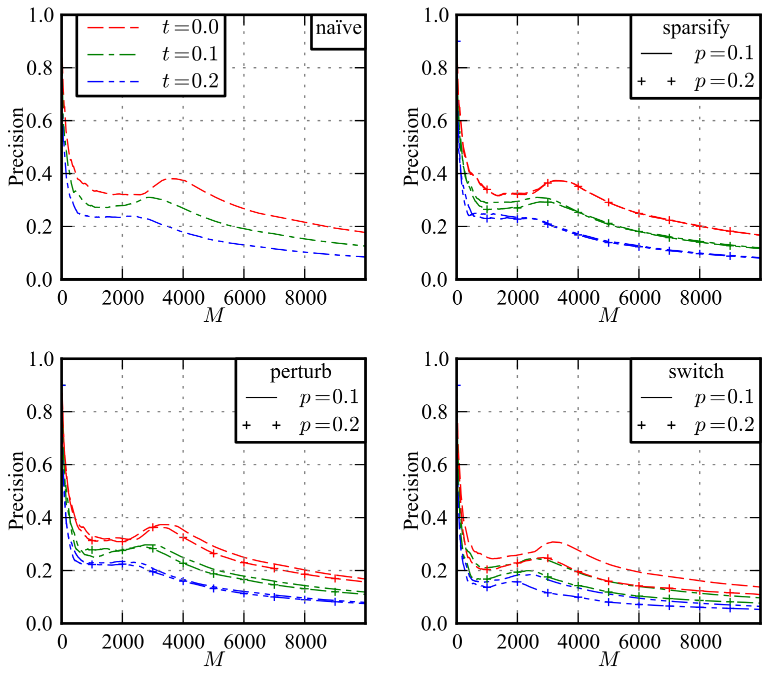


Figure 8.7: Attack precision with different attribute perturbation (tweibo, $K = 10^7$, 5,000 overlap). Results for other overlaps show similar behavior and are thus omitted.

would be easier with such additional information even if the auxiliary graph was rather smaller than the target graph.

8.3.4 Efficiency

According to Section 7.3.2, the running time highly depends on K (the number of maintained top candidate pairs). We used naïve anonymization as a case study to demonstrate the performance of our algorithm with different values of K. The result (Figure 8.8) 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.

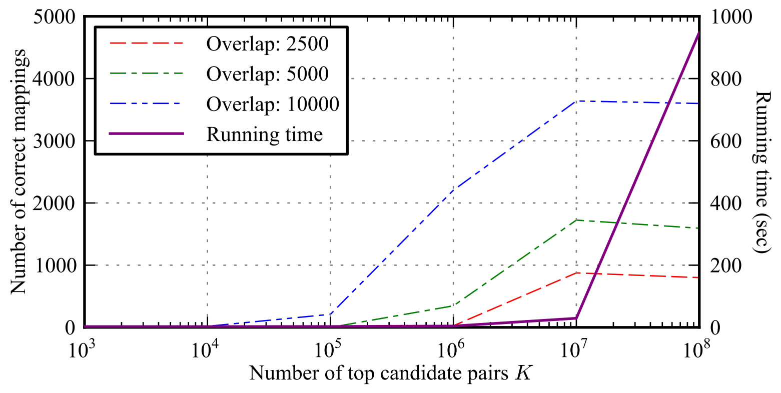


Figure 8.8: Precision and running time of attacks for naïve anonymization (tweibo, $K = 10^3, 10^4, ..., 10^8$)

CHAPTER 9

CONCLUSIONS AND FUTURE WORK

Heterogeneous information networks abound in real life but privacy preservation in such new settings has not received the due attention. In this work, we defined and identified privacy risk in anonymized heterogeneous information networks and presented a new de-anonymization attack that preys upon their risk. We further experimentally substantiated the presence of privacy risk and successfully tested the attack in the KDD Cup 2012 *t.qq* dataset. In addition, we also extended the ideas of exploiting privacy risk and de-anonymizing heterogeneous information networks to more general graphs both theoretically and empirically. One might find surprising the ease with which the devised attack can beat the investigated anonymization algorithms. While we have selected a small number of anonymization for this initial study, we have no reason to believe that other anonymization will prove impervious to this attack. Hence, our results make a compelling argument that privacy must be a central goal for sensitive heterogeneous information network publishers.

This thesis presents early results of our investigation. Planned future work includes: a) explore properties of the privacy risk metric and extend its applications; b) identify possible solutions for defending DeHIN, particularly without much utility loss.

A conference paper version of the key component of this thesis appears in Proceedings of the 17th International Conference on Extending Database Technology [37]. Two extensional papers on generalizing this key component appear in Proceedings of the 23rd International World Wide Web Conference (poster paper) [38] and ACM Transactions on Intelligent Systems and Technology [39].

REFERENCES

[1] J. Han, Y. Sun, X. Yan, and P. Yu, “Mining knowledge from data: An information network analysis approach,” in *IEEE 28th International Conference on Data Engineering (ICDE)*. IEEE, 2012, pp. 1214–1217.

[2] A. Narayanan and V. Shmatikov, “Robust de-anonymization of large sparse datasets,” in *IEEE Symposium on Security and Privacy (IEEE S & P)*. IEEE, 2008, pp. 111–125.

[3] “http://www.kddcup2012.org/c/kddcup2012-track1.”

[4] Y. Sun and J. Han, “Mining heterogeneous information networks: Principles and methodologies,” *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 2, 2012.

[5] C. Grier, K. Thomas, V. Paxson, and M. Zhang, “@ spam: the underground on 140 characters or less,” in *Proceedings of the 17th ACM conference on Computer and communications security (CCS)*. ACM, 2010, pp. 27–37.

[6] L. Sweeney, “k-anonymity: A model for protecting privacy,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[7] L. Backstrom, C. Dwork, and J. Kleinberg, “Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography,” in *Proceedings of the 16th international conference on World Wide Web (WWW)*, 2007, pp. 181–190.

[8] A. Narayanan and V. Shmatikov, “De-anonymizing social networks,” in *IEEE Symposium on Security and Privacy (IEEE S & P)*. IEEE, 2009, pp. 173–187.

[9] L. Sweeney, “Weaving technology and policy together to maintain confidentiality,” *The Journal of Law, Medicine & Ethics*, vol. 25, no. 2-3, pp. 98–110, 1997.

[10] S. Hansell, “Aol removes search data on vast group of web users,” *New York Times*, Aug 8 2006.

[11] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "l-diversity: Privacy beyond k-anonymity," *ACM Transactions on Knowledge Discovery from Data (ACM TKDD)*, vol. 1, no. 1, p. 3, 2007.

[12] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *IEEE 23rd International Conference on Data Engineering*. IEEE, 2007, pp. 106–115.

[13] A. Narayanan, H. Paskov, N. Gong, J. Bethencourt, E. Stefanov, E. Shin, and D. Song, "On the feasibility of internet-scale author identification," in *IEEE Symposium on Security and Privacy (IEEE S & P)*. IEEE, 2012, pp. 300–314.

[14] M. Newman, *Networks: an introduction*, 2009.

[15] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data (SIGMOD)*, 2008.

[16] B. Zhou and J. Pei, "Preserving privacy in social networks against neighborhood attacks," in *IEEE 24th International Conference on Data Engineering (ICDE)*, 2008.

[17] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM SIGKDD Explorations Newsletter*, vol. 10, no. 2, pp. 12–22, 2008.

[18] L. Zou, L. Chen, and M. Özsu, "K-automorphism: A general framework for privacy preserving network publication," *Proceedings of the VLDB Endowment (VLDB)*, vol. 2, no. 1, pp. 946–957, 2009.

[19] W. Wu, Y. Xiao, W. Wang, Z. He, and Z. Wang, "k-symmetry model for identity anonymization in social networks," in *Proceedings of the 13th international conference on extending database technology (EDBT)*. ACM, 2010, pp. 111–122.

[20] J. Cheng, A. Fu, and J. Liu, "K-isomorphism: privacy preserving network publication against structural attacks," in *Proceedings of international confonference on Management of data (SIGMOD)*, 2010.

[21] L. Sweeney, "Achieving k-anonymity privacy protection using generalization and suppression," *International Journal of Uncertainty, Fuzzy, and Knowledge-Based System*, vol. 10, no. 05, pp. 571–588, 2002.

[22] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*, 1994, vol. 8.

[23] J. Hopcroft and R. Karp, “An nˆ5/2 algorithm for maximum matchings in bipartite graphs,” *SIAM Journal on Computing*, vol. 2, no. 4, pp. 225–231, 1973.

[24] X. Wu, X. Ying, K. Liu, and L. Chen, “A survey of privacy-preservation of graphs and social networks,” *Managing and mining graph data*, pp. 421–453, 2010.

[25] V. D. Blondel, A. Gajardo, M. Heymans, P. Senellart, and P. V. Dooren, “A measure of similarity between graph vertices: applications to synonym extraction and web searching,” *SIAM Review*, vol. 46, no. 4, pp. 647–666, Apr. 2004.

[26] G. Jeh and J. Widom, “Simrank: a measure of structural-context similarity,” in *Proceedings of the 8th International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 538–543.

[27] S. Melnik, H. Garcia-Molina, and E. Rahm, “Similarity flooding: a versatile graph matching algorithm and its application to schema matching,” in *Proceedings of the 18th International Conference on Data Engineering*, 2002, pp. 117–128.

[28] A. Narayanan and V. Shmatikov, “De-anonymizing social networks,” in *Proceedings of the 2009 IEEE Symposium on Security and Privacy*, May 2009, pp. 173 –187.

[29] D. J. Watts and S. H. Strogatz, “Collective dynamics of ’small-world’ networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, June 1998.

[30] P. Bonacich, “Power and centrality: a family of measures,” *American Journal of Sociology*, pp. 1170–1182, 1987.

[31] K. Henderson, B. Gallagher, L. Li, L. Akoglu, T. Eliassi-Rad, H. Tong, and C. Faloutsos, “It’s who you know: graph mining using recursive structural features,” in *Proceedings of the 17th International Conference on Knowledge Discovery and Data Mining*, 2011, pp. 663–671.

[32] J. Yang and J. Leskovec, “Defining and evaluating network communities based on ground-truth,” in *Proceedings of the 2012 IEEE International Conference on Data Mining*, Dec 2012, pp. 745–754.

[33] K. Liu and E. Terzi, “Towards identity anonymization on graphs,” in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, 2008. [Online]. Available: http://doi.acm.org/10.1145/1376616.1376629 pp. 93–106.

[34] X. Ying and X. Wu, “Randomizing social networks: a spectrum preserving approach,” in *Proceedings of the 2008 SIAM International Conference on Data Mining*, 2008. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972788.67 pp. 739–750.

[35] A. Narayanan, E. Shi, and B. I. P. Rubinstein, “Link prediction by de-anonymization: How we won the kaggle social network challenge,” in *Proceedings of the 2011 International Joint Conference on Neural Networks*, 2011, pp. 1825–1834.

[36] R. Albert and A.-L. Barabási, “Statistical mechanics of complex networks,” *Reviews of modern physics*, vol. 74, pp. 47–97, Jan 2002. [Online]. Available: http://link.aps.org/doi/10.1103/RevModPhys.74.47

[37] A. Zhang, X. Xie, K. C.-C. Chang, C. A. Gunter, J. Han, and X. Wang, “Privacy risk in anonymized heterogeneous information networks.” in *Proceedings of the 17th International Conference on Extending Database Technology (EDBT)*, 2014, pp. 595–606.

[38] H. Fu, A. Zhang, and X. Xie, “De-anonymizing social graphs via node similarity,” in *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion (WWW)*, ser. WWW Companion ’14. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2014. [Online]. Available: http://dx.doi.org/10.1145/2567948.2577366 pp. 263–264.

[39] H. Fu, A. Zhang, and X. Xie, “Effective social graph de-anonymization based on graph structure and descriptive information,” *ACM Transactions on Intelligent Systems and Technology (ACM TIST)*, vol. 06, no. 04, 2015.