Limiting Link Disclosure in Social Network Analysis through Subgraph-Wise Perturbation

Amin Milani Fard
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
milanifard@cs.sfu.ca

Ke Wang
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
wangk@cs.sfu.ca

Philip S. Yu
Dept. of Computer Science
University of Illinois at Chicago
Chicago, Illinois, USA
psyu@cs.uic.edu

ABSTRACT

Link disclosure between two individuals in a social network could be a privacy breach. To limit link disclosure, previous works modeled a social network as an undirected graph and randomized a link over the entire domain of links, which leads to considerable structural distortion to the graph. In this work, we address this issue in two steps. First, we model a social network as a directed graph and randomize the destination of a link while keeping the source of a link intact. The randomization ensures that, if the prior belief about the destination of a link is bounded by some threshold, the posterior belief, given the published graph, is no more than another threshold. Then, we further reduce structural distortion by a *subgraph-wise perturbation* in which the given graph is partitioned into several subgraphs and randomization of destination nodes is performed within each subgraph. The benefit of subgraph-wise perturbation is that it retains a destination node with a higher retention probability and replaces a destination node with a node from a local neighborhood. We study the trade-off of utility and privacy of subgraph-wise perturbation.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining; K.4.1 [**Public Policy Issues**]: Privacy

General Terms

Algorithms, Measurement, Design, Experimentation

Keywords

Social networks, Privacy-preserving data publishing, Graph node partitioning

1. INTRODUCTION

Social network analysis is gaining more attention in the study of many human and natural phenomena. However, publishing the structure of a social network for such analysis may reveal sensitive information about participants. A compromise is often reached between the data publisher and the data miner, resulting in both parties agreeing to some methods that will be used to *anonymize* the network data prior to its publication. Graph anonymization is more challenging than relational data because the adversary's background knowledge can be any information about nodes, links, subgraphs, and etc. [3]. Another challenge is that measuring information loss of graph anonymization can be tricky because two graphs with the same number of vertices and the same number of links can vary considerably in their global properties such as betweenness, diameter, etc., that are central to social network analysis [17].

Several types of privacy disclosure for social network data have been identified: *identity disclosure*, *link disclosure*, and *content disclosure* [15]. Content disclosure refers to sensitive disclosure of content information associated with each individual, such as age, gender, sex orientation, etc. Such disclosures are typically addressed by data anonymization [1][11]. Even if no such content information is released, identity disclosure can occur, which refers to the re-identification of the node of a target individual using background knowledge on neighborhood structure such as the degree of a node [3]. Identity disclosure is typically limited by node anonymization, which renders the neighborhood of several nodes similar. Such techniques include degree k-anonymity [4][15][23], k-automorphism [24], and k-isomorphism [9]. Link disclosure refers to inference about the existence of a link between two known individuals and is the focus of this work.

1.1 Motivation

In this work, we consider the problem of limiting link disclosure, that is, we want to limit the ability of an adversary to infer the existence of a link between two individuals. The work in [22] shows that knowledge about the links of a node can undo node anonymization. In this sense, limiting link disclosure is more fundamental than limiting identity disclosure. One powerful technique for limiting link disclosure is *link perturbation* [13][18], where some number of randomly selected links are deleted from the graph and some number of randomly selected new links are inserted into the graph. The number of links deleted and the number of links inserted dictate the trade-off between utility and privacy. The analysis in [6] show that link perturbation may achieve meaningful levels of identity obfuscation while still preserving characteristics of the original graph.

All previous works on link perturbation have the following characteristics (or limitations).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
EDBT 2012, March 26-30, 2012 Berlin, Germany
Copyright 2012 ACM 978-1-4503-0790-1/12/03 ...$10.00.

- First, despite the fact that many real life social networks (e.g., Twitter followers, e-mail networks, Google+ circles, Facebook) are directed graphs, previous works in the field of privacy preserving data publishing model a social network as an undirected graph. For example, a directed link (u, v) in a trust network means that u trusts (or distrusts) v, whereas a link (v, u) means that v trusts (or distrusts) u. The privacy implication of disclosing the two links is different. Dealing with only undirected graphs, previous link perturbation [13][18] will randomize both the source node u and the distinction node v of a link (u, v), even though, to hide the link (u, v), it suffices to hide either the source u or the destination v. Consequently, considerable structural distortion is incurred.

- Second, the prior belief of a destination node is not factored in inferring a destination node. The long-tail degree distribution of social networks implies that a small number of nodes are highly popular destination nodes of links [5]. Hiding the links to such celebrity nodes is more difficult or even unrealistic because of the prior knowledge due to such a popularity. A reasonable privacy notion should take this prior into account and protect a destination node only if the prior is bounded by some threshold.

- Third, the deleted/inserted links are selected randomly from the *entire* graph without considering the structural proximity of nodes. For example, adding a link between two nodes that are far apart would make the graph useless for shortest path analysis, or deleting the only link connecting two parts of the graph would make the two parts disconnected.

1.2 Contributions

To address the above issues, we propose several important deviations from previous works. Our contributions are summarized as follows.

Contribution 1 We model a social network as a directed graph and hide the existence of a link by perturbing the destination node of the link while keeping the source node intact. Specifically, for each (directed) link (u, v), we retain (u, v) with a certain probability p and replace (u, v) with a link (u, w) with probability $1-p$, where w is randomly selected from all destination nodes in the graph. We name this solution *graph-wise perturbation*. Note that being able to handle directed graphs is a generalization, not a restriction, because undirected graphs can always be modeled by directed ones.

Contribution 2 We formalize the privacy of a link through (ρ_1,ρ_2)-privacy proposed in [10]. Informally, (ρ_1,ρ_2)-privacy, where $0 < \rho_1 < \rho_2 < 1$, says that if the adversary's *prior* belief (before seeing the published graph) that a node v is the destination of a link is no more than ρ_1, his *posterior* belief (after seeing the perturbed graph) that v is the true destination of a link is no more than ρ_2. In other words, publishing the data changes the belief of the adversary by at most $\rho_2 - \rho_1$. We give the maximum retention probability p for satisfying a given (ρ_1, ρ_2)-privacy requirement.

Contribution 3 We propose *subgraph-wise perturbation* in which the given graph is partitioned into local subgraphs and graph-wise perturbation is applied to each subgraph. By confining the choice of a randomized node to a smaller subgraph, this approach is able to preserve more structure of the graph by retaining a destination node of a link with a higher retention probability and by replacing a destination node with a node from a local neighborhood. We present an analysis on the impact of graph partitioning on privacy, and a way to reduce the impact. We study empirically the trade-off between utility and privacy.

The rest of the paper is organized as follows. Related work, preliminaries, and problem description are presented in Section 2, 3, and 4. Our subgraph-wise perturbation, experiments, and conclusion appear in Section 5, 6, and 7.

2. RELATED WORK

Most previous works consider limiting identity disclosure, e.g., [4][15][23][24]. As shown in [9], limiting identity disclosure does not limit link disclosure. Our work is most related to link disclosure. The first group of works in this area achieves some form of edge anonymity by transforming the graph to have some structural similarity. k-isomorphism [9] is a way to achieve such structural similarity where the original graph is transformed into k disconnected pairwise isomorphic subgraphs through link insertion and deletion. Another approach is partitioning by nodes into equivalence classes and inducing edge equivalence classes grouping links between node classes[20]. In general, considerable structural distortion is required to provide the desired structural similarity.

The second group of works is based on random link perturbation. The work in [18][13] considered randomly adding m non-existing links, randomly deleting m existing links, or randomly switching m pairs of links (by switching their endpoints). As explained earlier, since the graph is considered undirected and since the deleted links and inserted links are chosen from the entire graph, such perturbation incurs considerable structural distortion.

The third group of works aims to hide the existence of links from an adversary who employs link prediction techniques to infer the presence of sensitive links from non-sensitive links. [21] is a work in this group. To publish a graph, their method removes all sensitive links and some non-sensitive links in order to limit the adversary's ability of predicting (removed) sensitive links. We assume that all links in a graph are sensitive; therefore, if their method is applied to our graphs, no link will be published.

To our knowledge, our work is the first that models a social network as a directed graph in the field of privacy preserving data publishing, adopts (ρ_1, ρ_2)-privacy [10] to factor in the prior belief of a destination node, and retains more structure of a graph by subgraph-wise perturbation. Note that [10] does not consider social network data or link privacy.

3. PRELIMINARIES

In this section we introduce our data model, assumptions, perturbation operator, and privacy/utility measures.

3.1 Social Network Data

We represent a social network by a simple *directed* graph $G = (V, E)$, where V is the set of nodes $\{1, \ldots, |V|\}$ representing users, and E is the link table on two columns $(N1, N2)$ and contains one row for each link (u, v) representing a relationship from user u to user v. u is the source node of the link and v is the destination node of the link. Each node is associated with a unique (pseudonym) node ID and no content information about a node will be published. $Src(G)$ denotes the set of all source nodes and $Dst(G)$ denotes the set of all destination nodes in G. $d^{in}(u)$ denotes the *in-degree* of u, i.e., the number of links with u being the destination node, and $d^{out}(u)$ denotes the *out-degree* of u, i.e., the number of links with u being the source node. An undirected graph can be represented by replacing each undirected link (u, v) with two directed links (u, v) and (v, u). Figure 1(a) shows the link table E for the graph G in Figure 1(c).

Assumptions. (1) We assume that the destination of a link in G is chosen independently at random according to some underlying probability distribution (such as the power law of in-degree). This assumption is reasonable because no content information of a node is released, thus, can be used to correlate nodes. The underlying probability distribution, in the form of the in-degree distribution of nodes $Pr[X = x]$, is not private, and in fact, the data miner is allowed to learn it (more details in Section 3.3 and 3.4). (2) An adversary has access to the published graph and the algorithm and parameters used to produce the published graph, including the retention probability for perturbing the destination node of a link. With this information and the independence assumption in (1), the adversary tries to infer the true destination node of a link in the original graph, by computing the posterior probability of the true destination, given that the link is observed in the published graph (which was randomized by our algorithm). We do not assume that an adversary never identifies the node of a target individual; in fact, making this assumption would result in a weak privacy model. Rather, our privacy goal is to hide the existence of a link between two individuals by bounding the posterior probability of inferring the true destination of an observed link.

3.2 Link Perturbation

To hide the presence of a link (u, v), it suffices to hide either the source u or the destination v because knowing one but not the other does not reveal the existence of the link (u, v). We consider hiding the destination node, but the same method can be used to hide the source node by first reversing the direction of each link.

Specifically, for each link $(u, v) \in E$, we retain the destination v with some *retention probability* p and perturb the destination v to a node w randomly chosen from $Dst(G)$ with the probability $\frac{1-p}{m}$, where $m = |Dst(G)|$. Note that this perturbation process is done for each link independently. These probabilities can be represented by a perturbation matrix $P_{m \times m}$: the i^{th} column represents probabilities P_{ji} that the original destination node i is perturbed to a new destination node j. P_{ji} is defined as:

$$P_{ji} = \begin{cases} p + (1-p)/m & \text{if } j{=}i \text{ (retain } i\text{)} \\ (1-p)/m & \text{if } j{\neq}i \text{ (perturb } i \text{ to } j\text{)} \end{cases} \quad (1)$$

Let G^* denote the perturbed version of G and let E^* denote the link table for G^*. Note that this perturbation operator always preserves the out-degree of a node because the source node of a link is never perturbed. Figure 1(b) shows the perturbed table E^* with perturbed values in shaded cells, and Figure 1(d) shows the perturbed graph G^*.

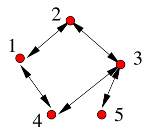
During the perturbation process, duplicate links and self-loops may be generated. Such links are called *singular links*

N1	N2
1	2
1	4
2	1
2	3
3	2
3	4
3	5
4	1
4	3
5	3

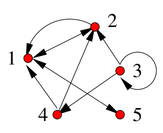
(a) Table E

N1	N2
1	2
1	5
2	1
2	1
3	2
3	3
3	4
4	1
4	2
5	1

(b) Table E^*



(c) Graph G



(d) Graph G^*

Figure 1: A simple social network graph

because they do not occur in the original graph G. We will keep all singular links because they are part of the perturbation and are useful for reconstructing the degree distribution (Section 3.4) and performing social network analysis (Section 6). We will discuss the privacy implication of such links and ways to deal with them in Section 5.3.

3.3 Privacy Model

We want to perturb the destination of a link to achieve some uncertainty of inferring the true destination of a published link. This goal is not achievable if a destination node is overly popular among the links. Therefore, privacy protection should be relative to the "prior" of a node as a destination of links. To capture such a relative notion of privacy, we adapt the (ρ_1, ρ_2)-privacy originally proposed by Evfimievski el al [10]. Consider the link table $E(N1, N2)$ for G and the link table $E^*(N1, N2)$ for G^*. Let r^* in E^* be the corresponding link of a link r in E. X and Y denote the random variables for the destination node x in $r[N2]$ and y in $r^*[N2]$ respectively. $Pr[X{=}x]$ denotes the *prior probability* that x is the destination of a link in G.

In the absence of further knowledge, we equate $Pr[X{=}x]$ with the fraction of source nodes linking to x, i.e., $\frac{d^{in}(x)}{|Src(G)|}$, where $d^{in}(x)$ is the in-degree of x in G. $Pr[X{=}x|Y{=}y]$ denotes the *posterior probability* that, given that a link is observed to have the destination y in G^*, x is the true destination of the link in G. Since the source node of a link is never perturbed, the probability $Pr[X{=}x|Y{=}y]$ represents the adversary's ability to infer the presence of a link between two individuals in G. We limit this ability by (ρ_1, ρ_2)-privacy [10].

Definition 1. (*(ρ_1, ρ_2)-privacy*): *Let $0{<}\rho_1{<}\rho_2{<}1$. There is an upward (ρ_1,ρ_2)-privacy breach w.r.t a value x for X if for some y for Y, $Pr[X{=}x]{\leq}\rho_1$ and $Pr[X{=}x|Y{=}y]{>}\rho_2$. (ρ_1,ρ_2)-privacy holds if upward privacy breach is eliminated.* □

Essentially, (ρ_1,ρ_2)-privacy says that whenever the prior does not exceed ρ_1, the posterior must not exceed ρ_2. For a more elaborated discussion, please see [10]. This definition is more restricted than the original definition in [10] which considers the prior $Pr[Q(X)]$ and posterior $Pr[Q(X)|Y{=}y]$ for a general predicate $Q(X)$ on X. We consider the restricted form $X = x$ of $Q(X)$ because we are concerned with the risk of inferring an *individual* destination node x of a link.

The values of ρ_1 and ρ_2 are set by the data publisher. For a destination node x, (ρ_1, ρ_2)-privacy requirement imposes the bound ρ_2 on $Pr[X{=}x|Y{=}y]$ if and only if $Pr[X = x] \leq \rho_1$. The *power-law* or *long-tail* distribution of degrees for social networks suggests that a small number of nodes may have a very large degree whereas the rest have a small degree [5]. Therefore, (ρ_1, ρ_2)-privacy will bound the posterior by ρ_2 for the majority of nodes, i.e., those with $Pr[X = x] \leq \rho_1$.

For the small number of nodes with $Pr[X = x] > \rho_1$, the prior is considered too high and link disclosure comes as a result of "prior belief", which occurs before the data release. Since such "celebrity" nodes are commonly referenced by other nodes, the existence of a link to such nodes is a "fact of life" and less sensitive to infer. Indeed, even if all links to celebrity nodes are removed from the published graph, the prior belief cannot be removed and thus no algorithm could prevent link disclosure for such nodes. For this reason, we focus on link disclosure mainly due to the release of data, like in most previous works. With the pre-condition $Pr[X = x] \leq \rho_1$, our privacy goal is to limit the inference of destination nodes that have a low in-degree. This in turn limits the adversary's ability to reconstruct the original graph.

We should mention that [10] does not consider social network data or privacy notions for links. We borrow from [10] the standard notion of (ρ_1,ρ_2)-privacy, but all social network related contributions are new.

3.4 Utility Metrics

The graph for a social network is published to serve a variety of analysis purposes. We consider three types of utility metrics in this regard: (i) common graph metrics such as centrality, clustering coefficient, diameter, and etc. [17]; (ii) the destination retention probability p (Equation (1)), which indicates the portion of original links preserved; (iii) the reconstruction of the in-degree of each node, which has applications such as popularity based ranking and influence of nodes. Note that our perturbation does not alter the out-degree of a node. Since (i) and (ii) are well defined, we explain only (iii).

Let $D = < d_1, \cdots, d_m >$ be the in-degree vector of destination nodes in G, and $O = < o_1, \cdots, o_m >$ be the observed in-degree vector of destination nodes in G^*, where $m = |Dst(G)|$. The data miner has access to O but not D, and wants to reconstruct an estimate of D. Directly using O as the estimate of D would give a large distortion because it does not take into account of the perturbation of destination nodes. A better estimate can be obtained by treating O as the result of applying the perturbation to the original graph, which is characterized by the perturbation matrix P in Equation (1) (recall that P is public). Let $E(O) = < E(o_1), \cdots, E(o_m) >$, where $E(o_j)$ is the mean of o_j, $\Sigma_{i=1}^{m} P_{ji} \times d_i$. We have $E(O) = P \times D$. Approximating $E(O)$ by the observed instance O, we get an estimate of D as $D' = P^{-1} \times O$, where P^{-1} is the inversion of P. D' can be computed by the *iterative Bayesian reconstruction* [2].

DEFINITION 2. (*Reconstruction error*): *Let D and D' be the actual and estimated in-degree vectors for G and G^* respectively. The reconstruction error is* $\frac{\sum_i |d_i - d'_i|}{\sum_i d_i}$. □

4. PROBLEM DESCRIPTION

The next question is what retention probability p is required to provide (ρ_1,ρ_2)-privacy on destination nodes. We propose two solutions, graph-wise perturbation and subgraph-wise perturbation.

4.1 Graph-wise Perturbation

In *graph-wise perturbation*, the replacing destination node is randomly selected from the *entire domain* of the destination nodes, $Dst(G)$. In this case, $m = |Dst(G)|$ for the perturbation matrix P in Equation (1). We can determine the maximum retention probability p in Equation (1) for ensuring (ρ_1,ρ_2)-privacy from a constraint called *amplification condition* due to [10]. Intuitively, this condition says that for any two destination nodes k and i, their probabilities of being perturbed to the same node j, i.e., P_{jk} and P_{ji}, must not differ by a factor of more than γ. Precisely, let X be a variable for a true destination node of a link in G, and Y be a variable for an observed destination node of a link in G^*. A perturbation matrix P is said to satisfy the γ-*amplification condition* [10] for a value j of Y if for all values k, i of X

$$\frac{P_{jk}}{P_{ji}} \leq \gamma \tag{2}$$

Therefore, when this condition holds, the small difference between P_{jk} and P_{ji} implies uncertainty that both k and i are likely to be the original destination node, when observing a destination node j in a link in the perturbed graph. The next theorem, also due to [10], relates the γ-amplification condition to (ρ_1, ρ_2)-privacy.

THEOREM 1. *(ρ_1, ρ_2)-privacy is guaranteed if the perturbation matrix P satisfies the γ-amplification condition for all values j of Y, where $\gamma \leq \frac{\rho_2}{\rho_1} \times \frac{1-\rho_1}{1-\rho_2}$.*

The proof of Theorem 1 can be found in [10]. Let us derive the maximum retention probability for guaranteing (ρ_1, ρ_2)-privacy from Theorem 1. Following Equation (1), P_{ji} is maximized if $i = j$, i.e., $p + (1-p)/m$, and P_{ji} is minimized if $i \neq j$, i.e., $(1-p)/m$. So the left-hand side of Equation (2) is maximized at $\frac{p+(1-p)/m}{(1-p)/m}$, when $j = k$ and $j \neq i$, and Equation (2) reduces to $\frac{p+(1-p)/m}{(1-p)/m} \leq \gamma$. To maximize the retention probability p, we replace $\leq$ with $=$ in the inequality and get $p = \frac{(\gamma-1)}{(m-1+\gamma)} = \frac{1}{m/(\gamma-1)+1}$. p is maximized by the maximum value of γ, which, according to Theorem 1, is

$$\gamma = \frac{\rho_2}{\rho_1} \times \frac{1-\rho_1}{1-\rho_2} \tag{3}$$

So Equation (1) with the maximum p that guarantees (ρ_1, ρ_2)-privacy has the form

$$P_{ji} = \begin{cases} \gamma/(m-1+\gamma) & \text{if } j = i \\ 1/(m-1+\gamma) & \text{if } j \neq i \end{cases} \tag{4}$$

with γ being given by Equation (3). Note that P_{ji} is completely determined by m, ρ_1, ρ_2.

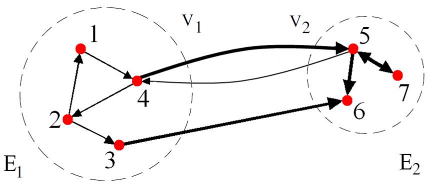
COROLLARY 1. *The perturbation matrix given by Equations (3) and (4) provides (ρ_1, ρ_2)-privacy.*

4.2 Subgraph-wise Perturbation

The above graph-wise perturbation has two limitations. First, the retention probability $p = \frac{\gamma-1}{(m-1+\gamma)}$ can be very low because $m = |Dst(G)|$ is large, easily thousands or more for real life social network graphs. Second, the unrestricted choice of the replacing destination node from the entire domain $Dst(G)$ means that the structure of the perturbed graph could be considerably different from the original graph.

N1	N2
2	1
4	2
2	3
1	4
5	4

E$_1$



E$_2$

N1	N2
4	5
7	5
3	6
5	6
5	7

Figure 2: Link partitioning

$$P^1 = \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{pmatrix} 0.43 & 0.19 & 0.19 & 0.19 \\ 0.19 & 0.43 & 0.19 & 0.19 \\ 0.19 & 0.19 & 0.43 & 0.19 \\ 0.19 & 0.19 & 0.19 & 0.43 \end{pmatrix} \quad P^2 = \begin{array}{c} 5 \\ 6 \\ 7 \end{array} \begin{pmatrix} 0.53 & 0.23 & 0.23 \\ 0.23 & 0.53 & 0.23 \\ 0.23 & 0.23 & 0.53 \end{pmatrix}$$

Figure 3: Subgraph-wise perturbation operators

A low retention probability due to a large randomization domain was first pointed out in [8] for relational data, but their work does not address graph data and the second limitation.

To tackle these limitations, we propose a *subgraph-wise perturbation* approach so that the replacing destination node is chosen from some local part of the graph. First, we partition the graph G into *link partitioned* subgraphs $G_1, \ldots, G_k$. In contrast to the usual node partitioning of a graph, the link partitioning scheme partitions the links into subgroups, so that each link in G will be contained in exactly one subgraph G_t. A node may appear in multiple subgraphs and the subgraphs may not be connected. Such link partitioning ensures that, when randomizing the links in each subgraph G_t, each link in G gets *exactly one* chance of being randomized.

Additionally, the link partitioning is expected to satisfy two properties: (i) the nodes in the same G_t are close to each other and (ii) each subgraph G_t has a smaller set $Dst(G_t)$ than $Dst(G)$. (i) ensures that a randomized destination node will not be structurally far away from the original destination node, and (ii) ensures a smaller domain size $m = |Dst(G_t)|$ of destination nodes in a subgraph G_t, compared to the domain size $m = |Dst(G)|$ of destination nodes in the whole graph G, in order to increase the retention probability for G_t.

Example 1. Figure 2 shows a link partitioning $\{G_1, G_2\}$ and the corresponding link table partitioning $\{E_1, E_2\}$. The links in each partition are marked with a different thickness. The two circles depict destination nodes in the two partitions: $Dst(G_1) = \{‘1’,‘2’,‘3’,‘4’\}$ and $Dst(G_2) = \{‘5’,‘6’,‘7’\}$.□

DEFINITION 3. (*Subgraph-wise perturbation*): *For a given k, we want to find a linkpartitioning of G, $G_1, \cdots, G_k$, such that, for $1 \leq t \leq k$, the nodes within each partition G_t are close to each other and $|Dst(G_t)|$ are as small as possible.* □

In subgraph-wise perturbation we will randomize the links in each G_t independently, $1 \leq t \leq k$, and produce the perturbed version G_t^*, using the $m_t \times m_t$ perturbation matrix P^t defined in Equation (4), where $m_t = |Dst(G_t)|$. To reconstruct the in-degree distribution D' for G, we can reconstruct the in-degree distribution D'_t from each G_t^* using the method described in Section 3.4 and aggregate D'_t over $1 \leq t \leq k$. Subgraph-wise perturbation helps address the two limitations of graph-wise perturbation mentioned at the beginning of Section 4.2 because m_t is smaller than m, where $m = |Dst(G)|$, and the replacing destination node of a link in G_t is chosen from destination nodes in a local neighborhood, $Dst(G_t)$. This point is illustrated by the next example.

Example 2. Consider Figure 2 again. Let ρ_1=0.4 and ρ_2=0.6, thus we have γ= 2.25 (Equation (3)). With graph-wise perturbation, m=$|Dst(G)|$=7 and $P_{jj} = \frac{\gamma}{|Dst(G)|-1+\gamma} = \frac{2.25}{7-1+2.25} = 0.27$ (Equation (4)). With subgraph-wise perturbation, $|Dst(G_1)|$=4 and $|Dst(G_2)|$=3, and Figure 3 shows the perturbation matrixes for the subgraphs in Figure 2. $P^1_{jj} = \frac{\gamma}{|Dst(G_1)|-1+\gamma} = \frac{2.25}{4-1+2.25}$=0.43 and P^2_{jj}=0.53. This shows that the retention probabilities for the subgraphs are significantly higher than the retention probability for the original graph. □

Remark 1. A question is how to publish the perturbed graphs $G_1^*, \cdots, G_k^*$. If the publication is for reconstructing the in-degree distribution of nodes, as in Section 3.4, publishing each G_t^* separately will facilitate a more accurate reconstruction because each G_t^* is produced by a different perturbation matrix P^t. In this case, the in-degree distribution, i.e., the prior probability, of a node in G_t, $Pr_t[X = x] = \frac{d_t^{in}(x)}{|Src(G_t)|}$, may differ from that in the original graph G. We will discuss the privacy implications due to this difference in Section 5.2. In the case that the data is published for social network analysis, such as the study on closeness and betweenness centrality, clustering coefficient, graph diameter, eigenvalues, and shortest paths, it suffices to publish all G_t^* in a *single merged graph* G^* through common nodes without separating them apart. For example, the merged graph is obtained in Figure 2 by ignoring the different thickness of links for different partitions. In this case, the prior probability of a destination node remains unchanged.

5. SUBGRAPH-WISE PERTURBATION

In this section, we present an algorithm for subgraph-wise perturbation problem. This is shown in Algorithm 1. Given the input graph $G(V, E)$, the number of partitions k, the privacy setting ρ_1 and ρ_2, and the sparsity parameter ℓ (explained later), the algorithm consists of following main steps.

- Partition the node set V into k subsets $U_1, \cdots, U_k$ (Line 1). U_i contains the destination nodes for subgraph G_i. This partition ensures the destinations in each G_i are close to each.
- Induce k link partitioned subgraphs $G_1, \cdots, G_k$ using $U_1, \cdots, U_k$ (Line 2). That is, G_i contains all links to the destinations in U_i.
- Balance the in-degree of nodes in $G_1, \cdots, G_k$ to limit the prior probability of destination nodes in G_t (Lines 3-4), which will be explained in Section 5.2.
- Enforce a notion of ℓ-sparsity on each G_t (Line 5), which will be explained in section 5.3.
- Lastly, the algorithm perturbs the destination nodes of links in each subgraph G_t independently and outputs the perturbed G_t^* and P^t (Lines 6-11).

Steps 3 and 4 address some privacy issues introduced by the graph partitioning. The details will be discussed in Sections 5.2 and 5.3.

Algorithm. 1 Subgraph-wise Perturbation

Input: A directed graph $G(V, E)$, number of partitions k, privacy settings (ρ_1, ρ_2), sparsity parameter ℓ
Output: Perturbed subgraphs G^*_t, matrices P^t; $1 \leq t \leq k$

```
 1: $\{U_1, \ldots, U_k\} \leftarrow \mathit{VertexPartition}(G, k)$
 2: $\{G_1, \ldots, G_k\} \leftarrow \mathit{InduceSubgraphs}(U_1, \ldots, U_k)$
 3: Build the sorted list of exposed nodes $\mathit{Exposed}$
 4: $\mathit{DegreeBalance}(G_1, \ldots, G_k, \rho_1, \mathit{Exposed})$
 5: $\ell$-$\mathit{sparsity}(G_1, \ldots, G_k, \ell)$
 6: $\gamma \leftarrow (\rho_2 \times (1 - \rho_1))/(\rho_1 \times (1 - \rho_2))$
 7: for each subgraph $G_t$ do
 8:     Build $P^t$ using Equation (4) where $m = |Dst(G_t)|$
 9:     Using $P^t$, perturb destination nodes of links in $G_t$
10:     Output the perturbed subgraph $G^*_t$ and $P^t$
11: end for
```

Theorem 2. *Algorithm 1 is in $O(|E| + |V| \cdot log\ |V|)$.*

Proof. The multilevel k-way partitioning [14] (line 1), and inducing a link partitioning (line 2), are both in $O(|E|)$. Sorting exposed nodes (line 3) is in $O(|V| \cdot log\ |V|)$. Degree balancing (line 4) according to Theorem 3 in Section 5.2 is in $O(|E|)$ and ℓ-sparsity (line 5) is in $O(|V| \cdot \ell)$. Finally generating G^* (lines 7-11) is in $O(|E|)$. Since $log\ |V|$ is typically larger than ℓ, the overall complexity is $O(|E| + |V| \cdot log\ |V|)$. □

Let us explain Lines 1-5 in Algorithm 1 in details.

5.1 Link Partitioning (Lines 1-2)

Lines 1 and 2 partition G into link partitioned subgraphs $G_1, \cdots, G_k$ such that highly connected nodes are in the same subgraph and $|Dst(G_t)|$ is as small as possible. In Line 1, *VertexPartition*(G, k) partitions the set of nodes V into k disjoint sets $U_1, \ldots, U_k$ of balanced size such that the number of links with the endpoints in different sets is minimized. This is the standard node partitioned graph partitioning. We recommend *METIS*[1], an open source software for partitioning graphs [14]. Intuitively, each U_i contains the destination nodes in the subgraph G_i. This partitioning ensures that the destination nodes in each G_i are close to each other.

In Line 2, *InduceSubgraphs*$(U_1, \ldots, U_k)$ induces a link partitioning $G_1(V_1, E_1), \ldots, G_k(V_k, E_k)$ from $U_1, \ldots, U_k$. For $1 \leq t \leq k$, E_t contains all links with the destination nodes being in U_t and V_t is the set of nodes on either sides of the links in E_t. Since each U_t contains highly connected nodes, the destination nodes within each G_t are close to each other. This property is important because we want a replacing destination node to be structurally close to the original destination node. At this point, $G_1, \ldots, G_k$ is both a link partitioning and a destination-partitioning, but is not necessarily a source-partitioning, as shown in the next example. In other words, a node may appear in multiple partitions as source nodes, but appear in one partition as a destination node.

Example 3. Figure 2 shows a link partitioning $\{G_1, G_2\}$. The first step produces the node partitioning $U_1 =$\{'1','2','3','4'\} and $U_2 =$\{'5','6','7'\}. The second step produces the link partitioning $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$, where the ith partition E_i includes all the links that have U_i as the destination node set. Note that a node (such as the node '4') may appear in more than one partition as a source node, but only in one partition as a destination node. □

[1]http://glaros.dtc.umn.edu/gkhome/metis/metis/download

5.2 Degree Balancing (Lines 3-4)

(ρ_1, ρ_2)-privacy bounds the posterior probability $Pr[X = x|Y = y]$ only if the prior probability $Pr[X = x] = \frac{d^{in}(x)}{|Src(G)|}$ is $\leq \rho_1$. If each G^*_t is published separately, the destination node x in a subgraph G_t has a new prior $Pr_t[X = x] = \frac{d^{in}_t(x)}{|Src(G_t)|}$, where $d^{in}_t(x)$ denotes the in-degree of x in G_t. If $Pr[X = x] \leq \rho_1$ but $Pr_t[X = x] > \rho_1$, (ρ_1,ρ_2)-privacy w.r.t. G_t will not impose the bound ρ_2 on the posterior. We call such nodes "exposed nodes". The degree balancing step aims to reduce the number of exposed nodes.

Definition 4 (Exposed nodes). *For a destination node x in G_t, if $Pr[X = x] \leq \rho_1$ and if $Pr_t[X = x] > \rho_1$, x is called an* exposed node *in G_t.* □

Remark 2. The long-tail degree distribution of social networks implies that most nodes x have a low in-degree and the prior probability $Pr[X = x]$ is well below ρ_1, consequently, $Pr_t[X = x]$ in G_t remains below ρ_1. A few nodes that have a prior $Pr[X = x]$ "close enough" to ρ_1 in G may become exposed nodes after link partitioning. For example, consider the *Newman*'s scientific collaboration network [16] with 16,264 destination nodes. Let $\rho_1 = 0.01$. The highest in-degree is 107, thus, the highest prior probability is $Pr[X = x] = \frac{107}{16{,}264} \leq \rho_1$. If we partition the graph (using *METIS*) into $k = 5$ subgraphs with an (almost) equal number of destination nodes, the number of source nodes within each subgraph would be 3886, 4147, 4136, 4481, and 4273. Note that the partitioning is not source disjoint. To avoid becoming an exposed node, the maximum in-degree for a node in each subgraph is $\rho_1 \times |Src(G_t)|$, that is, 38.86, 41.47, 41.36, 44.81, and 42.73, for G_1, ..., G_5. Only 67 nodes have an in-degree above this threshold, so $\frac{67}{16{,}264}$=0.41% of all nodes are exposed nodes.

To reduce the number of exposed nodes x in G_t, the *DegreeBalance* step reduces the prior $Pr_t[X = x] = \frac{d^{in}_t(x)}{|Src(G_t)|}$ in two ways: either decrease $d^{in}_t(x)$ by moving out some links to x in G_t to other subgraphs $G_{t'}$ while keeping $|Src(G_t)|$ unchanged, or increase $|Src(G_t)|$ by moving some links from other subgraphs $G_{t'}$ with *new* source nodes to G_t while keeping $d^{in}_t(x)$ unchanged. This move is called a *prior-reducing* move for G_t and the subgraph $G_{t'}$ is called the *donor* of the move. Additionally, such moves must not yield new exposed nodes in the donor $G_{t'}$. We say that a prior-reducing move for G_t involving the donor $G_{t'}$ is *safe* if the move does not increase the number of exposed nodes in $G_{t'}$.

DegreeBalance reduces the number of exposed nodes by a sequence of safe prior-reducing moves, or simply "safe-moves". This is implemented by Algorithm 2. It gets an input list of exposed nodes x in the ascending order of in-degree, *Exposed*, and iteratively reduces the prior probability $\frac{d^{in}_t(x)}{|Src(G_t)|}$ by safe-moves for G_t. Exposed nodes are considered in the ascending order of in-degree because nodes with lower in-degree are more sensitive and are considered first. Each iteration has two phases.

Algorithm. 2 DegreeBalance

```
Input: Subgraphs $G_1,\cdots,G_k$, parameter $\rho_1$, the list of exposed nodes in ascending order of in-degree, Exposed
Output: Balanced subgraphs $G_1,\ldots,G_k$
1: repeat
2:    Out-phase:
3:    for all $x \in Exposed$ where $x$ is exposed in $G_t$ do
4:       Safe-move as much as possible but no more than $\alpha_t(x)$ links $(y,x)$ from $G_t$ to some $G_{t'}$ with maximum $|Src(G_{t'})|$
5:       Remove $x$ from Exposed if $x$ is not exposed in any subgraph
6:    end for
7:    In-phase:
8:    for all $x \in Exposed$ where $x$ is exposed in $G_t$ do
9:       Safe-move as much as possible but no more than $\beta_t(x)$ links $(w,z)$ from some $G_{t'}$ to $G_t$, where $z \neq x$ and $w \notin Src(G_t)$
10:      Remove $x$ from Exposed if $x$ is not exposed in any subgraph
11:   end for
12: until no safe-move is done or Exposed is empty
```

- *Out-phase* (line 2-6) reduces the in-degree $d_t^{in}(x)$ for an exposed node x in G_t by safe-moving out links to x as much as possible, but no more than $\alpha_t(x)$. $\alpha_t(x)$ is the minimum number required to satisfy $\frac{d_t^{in}(x)-\alpha_t(x)}{|Src(G_t)|} \leq \rho_1$, i.e., $\alpha_t(x) = \lceil d_t^{in}(x) - \rho_1 \times |Src(G_t)| \rceil$. Given choices, we choose the donor subgraph $G_{t'}$ with the largest $|Src(G_{t'})|$ because such subgraphs have a small increase in the prior probability $\frac{d_{t'}^{in}(x)}{|Src(G_{t'})|}$ after accepting more links to x.
- *In-phase* (line 7-11) increases $|Src(G_t)|$ for G_t containing some exposed nodes x by safe-moving in links (w,z) with *new* source nodes w as much as possible, but no more than $\beta_t(x)$, which is the minimum number required to satisfy $\frac{d_t^{in}(x)}{|Src(G_t)|+\beta_t(x)} \leq \rho_1$. $\beta_t(x) = \lceil d_t^{in}(x)/\rho_1 - |Src(G_t)| \rceil$.

The next theorem, shows nice properties of degree balancing in Algorithm 2.

THEOREM 3. *Algorithm 2 never increases the number of exposed nodes and will terminate in $O(|E|)$.*

PROOF. It never increases the number of exposed nodes because it only makes safe-moves. For an exposed node x in G_t, each safe-move either reduces $d_t^{in}(x)$ (i.e., out-phase), or increases $|Src(G_t)|$ (i.e., in-phase). Each such safe-move brings $\frac{d_t^{in}(x)}{|Src(G_t)|}$ closer to ρ_1, so the iterative process is guaranteed to terminate. The number of safe moves required to finish the algorithm is in $O(\sum_{\forall x}\alpha_t(x) + \sum_{\forall x}\beta_t(x))$. Since $\sum_{\forall x}\alpha_t(x)$ and $\sum_{\forall x}\beta_t(x)$ are both in $O(|E|)$, Algorithm 2 is in $O(|E|)$. □

Example 4. Consider the two subgraphs in Figure 2. Let ρ_1=0.4 and ρ_2=0.6. $Pr_1[X=\text{'1'}]=Pr_1[X=\text{'2'}]=Pr_1[X=\text{'3'}]=\frac{1}{4}$, $Pr_1[X=\text{'4'}]=\frac{2}{4}$, $Pr_2[X=\text{'5'}]= Pr_2[X=\text{'6'}]=\frac{2}{4}$, $Pr_2[X=\text{'7'}]=\frac{1}{4}$. Since the prior probability of node '4' in G_1 and the prior probability of nodes '5' and '6' in G_2 are $> \rho_1$, and their prior probability in G is $\leq \rho_1$, these nodes are exposed and *Exposed*=<'4','5','6'>.

Out-phase: Consider node '4' first. There is no safe-move out of G_1 for links (1,4) and (5,4) because their source nodes have out-degree of 1. Consider node '5'. Similarly, there is no safe-move out of G_2 for links (4,5) and (7,5) because their source nodes have out-degree of 1. Consider node '6'. We can safe-move the link (5,6) from G_2 to G_1. Since α_2('6')=1, it suffices to move one link, so node '6' is removed from *Exposed.*

In-phase: Consider node '4'. Since all links in G_2 have a source node with out-degree of 1, there is no safe-move of any link from G_2 into G_1. Consider node '5'. We can safe-move the link (2,1) from G_1 to G_2 to increase $|Src(G_2)|$. Since β_2('5')=1, node '5' is removed from *Exposed.*

In the second iteration, *Exposed*=<'4'>. We can safe-move out (5,4) from G_1 to G_2. Since α_1('4')=1, node '4' is removed from *Exposed.* There is no more exposed node. The final two subgraphs have the links E_1={(4,2), (2,3), (1,4), (5,6)} and E_2={(4,5), (5,4), (7,5), (3,6), (5,7), (2,1)}.□

Remark 3. Algorithm 2 provides no guarantee to eliminate all exposed nodes. In this sense, subgraph-wise perturbation serves a trade-off between utility and privacy. As discussed in Remark 2, social networks are highly sparse and follow a long-tail degree distribution. Thus, we expect that only nodes with a very high in-degree are likely to exceed ρ_1 after the linking-partitioning, but such nodes are naturally less sensitive because they are commonly referenced. Furthermore, as discussed in Remark 1, for many social network applications, it suffices to publish all G_t^* in a single merged graph G^* without separating them apart. In this case, the prior probability of destination nodes remains unchanged, so the degree balancing step in Algorithm 1 is not needed.

5.3 Anonymizing Singular Links (Line 5)

While the problem in Section 5.2 is caused by a large in-degree of destination nodes in a subgraph, we now consider a problem caused by a large out-degree of source nodes in a subgraph. Consider E_2 in Figure 2 where node '5' is connected to all nodes in $Dst(G_2)$ but itself. Suppose that the link (5,7) is perturbed to (5,5) or (5,6), we have either a self-loop or duplicate links. Such links are called *singular links.* Since singular links do not appear in the original graph and since node '7' is the only unused destination node in G_2^*, it is immediate that the self-loop or the multiple link must come from the original link (5,7).

In general, if a source node x is connected to *all or most* destination nodes in G_t (for a large G_t, this is very unlikely), there is an increased chance to infer the true destination of a singular link originating at x since each singular link will leave exactly one destination node to which x has no link in G_t^*. This node in fact is the true destination node for the singular link.

Simply discarding the singular links from the perturbed subgraph G_t^* will interfere with the perturbation intended for providing (ρ_1,ρ_2)-privacy. Our solution is to keep all singular links in G_t^* but remove some links from G_t *prior to* the perturbation of G_t. Specifically, we want to ensure that each source node in $Src(G_t)$ has no link to some minimum number, say $\ell - 1$, of nodes in $Dst(G_t)$. This condition implies that for a singular link in G_t^*, there are at least ℓ choices for its true destination node, thus, bounding the guessing probability by $\frac{1}{\ell}$.

Definition 5. (*ℓ-sparsity*): *Let $Dst(G_t)\backslash x$ denote $Dst(G_t)$ without node x. A subgraph G_t is ℓ-sparse if each source node $x \in Src(G_t)$ is not connected to at least $\ell-1$ destination nodes in $Dst(G_t)\backslash x$. In other words, the out-degree of any node in $Src(G_t)$ is at most $|Dst(G_t)\backslash x|+1-\ell$.* □

Lemma 1. *If a subgraph G_t is ℓ-sparse, the probability of guessing the true destination node for each singular link in G_t^* is no more than $\frac{1}{\ell}$.*

ℓ-sparsity with a value of ℓ from 5 to 10 should provide sufficient uncertainty. Given that a social network graph is typically large and sparse, we expect that ℓ-sparsity typically holds for a subgraph G_t without further modification to G_t, unless G_t is very small. In the rare case that G_t is not ℓ-sparse, some node x in $Src(G_t)$ has an out-degree more than $|Dst(G_t)\backslash x|+1-\ell$. We can satisfy the condition of ℓ-sparsity by deleting no more than $\ell-1$ out-going links (x, v) from G_t. If v becomes isolated after deleting the link (x, v), $Dst(G_t)$ still contains v. Note that this link removal does not reduce $|Src(G_t)|$ because x remains a source for other links, thus, does not increase the prior probability of any node in G_t and does not create new exposed nodes. The step *ℓ-sparsity*$(G_1, \ldots, G_k, \ell)$ in Algorithm 1 is based on this idea.

6. EXPERIMENTS

We implemented the proposed algorithms to evaluate the utility of perturbed graphs on a system with core-2 Duo 2.99GHz CPU with 3.83 GB memory. The implementation was done in VC++. For social network analysis we used *UCINET* software [7].

6.1 Experiment Setup

6.1.1 Data sets

Our first data set is the e-mail network of *University Rovirai Virgili* (URV) with 1,132 nodes and 10,900 links [12]. This graph shows e-mail interchanges between members of the university. Data providers eliminated e-mails that were sent to more than 50 different recipients (spam ignored) and also only considered bidirectional interchanges.

The second data set is the *Newman*'s scientific co-authorship network [16] with 16,264 nodes and 95,188 links (47,594 bidirectional links), which is a co-authorship network of scientists posting preprints on the High-Energy Theory in *arXiv* E-Print Archive between 1995 and 1999. Figure 4 shows the long-tail degree distribution of these data sets.

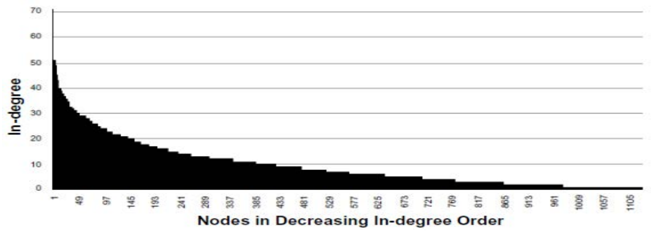
6.1.2 Parameters

For (ρ_1, ρ_2)-privacy, we set ρ_1 to 0.01, and ρ_2 to 0.1, 0.2, 0.3, and 0.4. We set the number of partitions k to 1, 10, 50 and 100 for the *URV* network, and to 1, 100, 500, and 1000 for the *Newman*'s network. For both data sets, the generated subgraphs are ℓ-sparse for ℓ=5 and all settings of k. Therefore, the step *ℓ-sparsity*$(G_1, \ldots, G_k, \ell)$ in Algorithm 1 can be skipped.

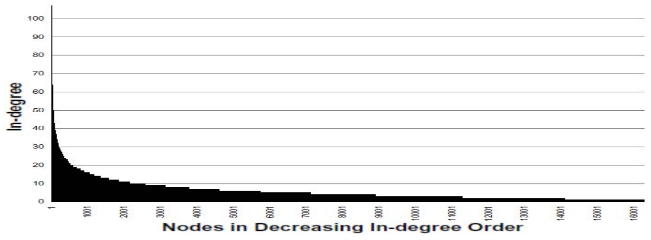
6.1.3 Evaluation methods

We compare the following approaches:

Subgraph-Wise Perturbation, with and without degree balancing. This is the proposed Algorithm 1.



(a) *URV* email network



(b) *Newman*'s co-authorship network

Figure 4: Network degree distribution

Table 1: Average values for the *URV* email network

k	1	10	50	100
$\lvert E_i\rvert$	10900	1090	218	109
$\lvert Dst(G_i)\rvert$	1132	113.19	22.63	11.32
$\lvert Dst(G_i)\rvert$ *(balancing)*	1132	291.39	60.41	26.21
$\lvert Src(G_i)\rvert$	1132	320	111.1	70.25
$\lvert Src(G_i)\rvert$ *(balancing)*	1132	466.2	142.08	81.83
P_{ii}	0.05	0.37	0.75	0.85
P_{ii} (balancing)	0.05	0.23	0.67	0.83
Coverage	100%	41.53%	11.98%	4.85%
Coverage (balancing)	100%	99.75%	56.82%	41.66%

Table 2: Average values for the *Newman*'s network

k	1	100	500	1000
$\lvert E_i\rvert$	95188	951.88	190.37	95.188
$\lvert Dst(G_i)\rvert$	16264	162.63	32.52	16.26
$\lvert Dst(G_i)\rvert$ *(balancing)*	16264	269.70	43.38	21.30
$\lvert Src(G_i)\rvert$	16264	272.02	68.56	40.56
$\lvert Src(G_i)\rvert$ *(balancing)*	16264	387.62	95.01	53.39
P_{ii}	0.004	0.29	0.67	0.83
P_{ii} (balancing)	0.004	0.28	0.65	0.81
Coverage	100%	35.65%	3.42%	0.4%
Coverage (balancing)	100%	90.02%	64.52%	58.60%

Graph-Wise Perturbation: This is the special case of *Subgraph-Wise Perturbation* when k=1.

Random Add/Del: This is the implementation of the link perturbation approach in [18], which randomly adds n non-existing links and randomly deletes n existing links. n is derived from the method in [19] as follows. Let A and A^* be the adjacency matrix of G and G^* respectively. According to [19], both the link retention probability and the posterior belief $Pr[a_{ij}=1|a^*_{ij}=1]$ are equal to $\frac{|E|-n}{|E|}$. We bound $Pr[a_{ij}=1|a^*_{ij}=1]$ by ρ_2. Thus n=(1-ρ_2)$\times|E|$.

We investigate the following utility metrics.

- ***Link retention probability***. For *Subgraph-Wise Perturbation* and *Graph-Wise Perturbation*, this is P_{ii}, and for *Random Add/Del*, this is $\frac{|E|-n}{|E|}$, where n is the number of links deleted.
- ***Reconstruction error***. This is the reconstruction error of in-degree distribution in Definition 2. Only *Graph-Wise Perturbation* and *Subgraph-Wise Perturbation* use this metric.
- ***Common graph metrics***. We study social network analysis metrics including degree/closeness/betweenness centralities, clustering coefficient, graph diameter, and largest eigenvalue. For detailed definitions, see [17]. These metrics apply to all methods.

6.2 Results

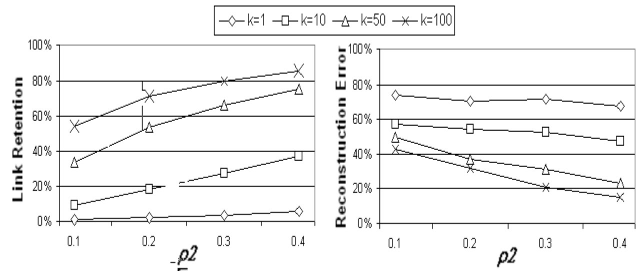
6.2.1 The number of exposed nodes

First, we examine the impact of *Subgraph-Wise Perturbation* on the number of exposed nodes, i.e., the destination nodes that have the prior $\leq \rho_1$ in G but have the prior $> \rho_1$ in some subgraph G_t. This can be measured by the percentage of destination nodes that have the prior $\leq \rho_1$ in all subgraphs G_t after graph partitioning, among those destination nodes that have the prior $\leq \rho_1$ in G. With our setting of $\rho_1 = 0.01$, before graph partitioning, 73% of destination nodes in the *URV* networks have the prior $\leq \rho_1$, and 100% of destination nodes in the *Newman*'s network have the prior probability $\leq \rho_1$. The last two rows in Tables 1 and 2 show the percentage of destination nodes that have the prior $\leq \rho_1$ in all subgraphs G_t, with and without degree balancing, among the destination nodes that have the prior $\leq \rho_1$ in G.

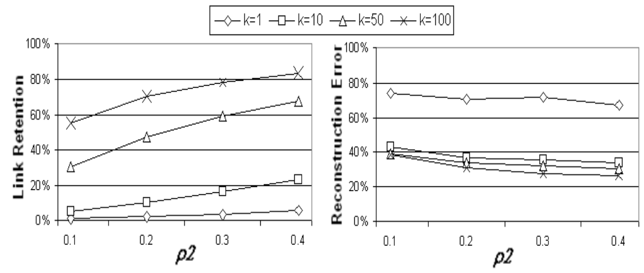
With a larger number of subgraphs, k, $|Src(G_t)|$ for each subgraph G_t becomes smaller, thus, more nodes become exposed after graph partitioning. In this case, degree balancing is effective to reduce the number of exposed nodes. For example, with $k = 10$ for *URV* and $k = 100$ for *Newman*, without degree balancing, 41.53% and 35.65% of those with the prior $\leq \rho_1$ in G also have $\leq \rho_1$ in all subgraphs G_t, but with degree balancing, these percentages are 99.75% and 90.02%, respectively. In other words, except for a very large k, most nodes that have their posterior bounded by ρ_2 in G^* also have their posterior bounded by ρ_2 in all subgraphs G^*_t

6.2.2 Link retention probability

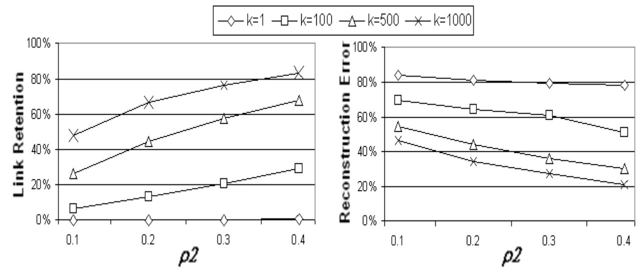
Retention probability is a direct indicator of how much original links are preserved. For *Subgraph-Wise Perturbation*, the left side of Figure 5 shows the retention probability vs ρ_2 and the number of subgraphs k. The retention probability increases as ρ_2 increases, and a larger k results in a significant higher retention probability. In fact, a larger k implies that the domain size $|Dst(G_t)|$ for destination nodes in a subgraph G_t is reduced, and according to Equation (4), this contributes to a larger retention probability. Tables 1 and 2 show the detailed statistics on $|E_t|$, $|Dst(G_t)|$, $|Src(G_t)|$, and retention probability P_{ii}, with respect to k. ρ_1 is set to 0.01 and ρ_2 is set to 0.4.



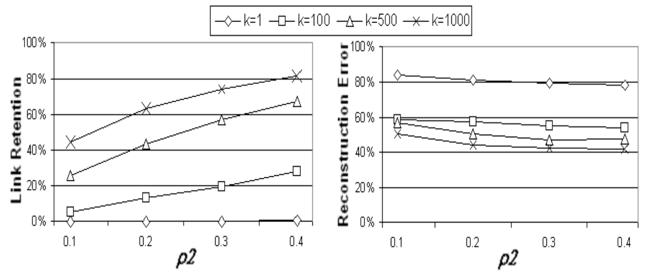
(a) *URV* email network - without degree balancing



(b) *URV* email network - with degree balancing



(c) *Newman*'s co-authorship network - without degree balancing



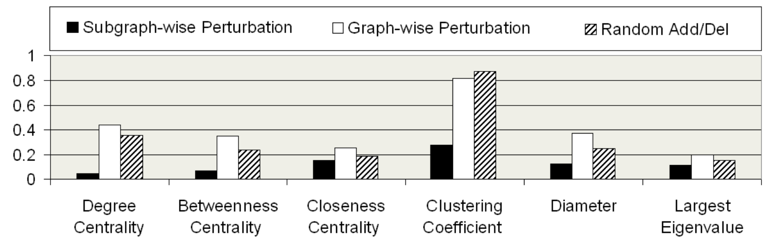
(d) *Newman*'s co-authorship network - with degree balancing

Figure 5: Link retention and reconstruction error comparison

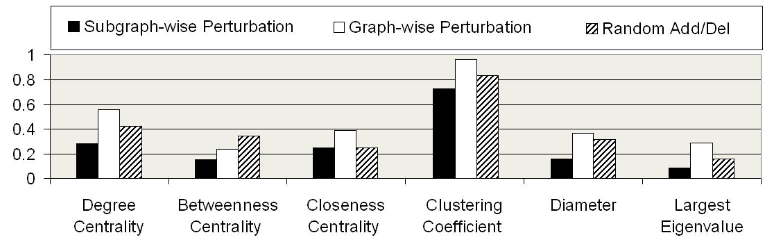
The link retention probability for *Random Add/Del* is equal to $\frac{|E|-n}{|E|}$, which is bounded by ρ_2, that is, 0.1, 0.2, 0.3 and 0.4. These probabilities are lower than those of *Subgraph-Wise Perturbation* for a large k. This study supports our claim that subgraph-wise perturbation better retains structural information than graph-wise perturbation.

6.2.3 In-degree reconstruction error

This experiment studies the accuracy of reconstructing the in-degree of each node from the published graph, where the accuracy is measured by the reconstruction error (Definition 2). The right side of Figure 5 shows the reconstruction error of in-degree distribution vs ρ_2 and k, where k is



(a) *URV* email network



(b) *Newman*'s co-authorship network

Figure 6: Relative error for social network analysis

the number of partitions. Without degree balancing, for a larger k, the increased retention probability contributes to a decreased reconstruction error. With degree balancing, the effect of increasing k to reduce the reconstruction error is less significant. This is because degree balancing increases $m = |Dst(G_t)|$ in a receiving subgraph G_t of a link, thus, reduces the retention probability in G_t according to Equation (4).

This study suggests that subgraph-wise perturbation is more effective in reducing the reconstruction error when protecting low in-degree nodes is the priority, in which case degree balancing is not needed.

6.2.4 Social network analysis

One important motivation of publishing social network data is to study a variety of social network behaviors. Figure 6 shows the relative error of four methods for common social network metrics. Subgraph-wise perturbation is run without the degree balancing step because we only publish a single merged graph G^* for performing social network analysis, as discussed in Remarks 1 and 3. For each method, we run 10 trials of randomization and reported their average results. The values of closeness/degree/betweenness centralities are the average over all nodes. In this experiment, ρ_1 is set to the default value, ρ_2 is set to 0.4, and for *Subgraph-Wise Perturbation*, k=100 for the email network and k=1000 for the co-authorship network.

For almost the metrics studied, *Subgraph-Wise Perturbation* gives more accurate results than *Graph-Wise Perturbation* and *Random Add/Del*. *Graph-Wise Perturbation* has a large error across most metrics used. Indeed, all these metrics depend on the local structure (such as degree centrality and clustering coefficient) and/or global structure (such as diameter and largest eigenvalue) of the graph to a varied degree. Compared to *Graph-Wise Perturbation*, *Subgraph-Wise Perturbation* better preserves such structures by retaining more original destination nodes of links and replacing an original destination node with a structurally close destination node when a link is perturbed. This study supports our claim that *Subgraph-Wise Perturbation* is superior to *Graph-Wise Perturbation* for general social network data analysis.

7. CONCLUSION

We presented a novel link perturbation algorithm, subgraph-wise perturbation, to limit link disclosure in publishing social network data. This algorithm addresses the excessive structural distortion in previous link perturbation methods through two novel modifications: It treats a social network as a directed graph and perturbs only the destination of a link, instead of both the source and the destination, and it partitions a large graph into several small subgraphs and perturbs the destination nodes within each subgraph. Compared to the previous graph-wise perturbation, subgraph-wise perturbation is able to preserve more graph structure while providing a given privacy guarantee by (1) completely preserving the source nodes of all links, (2) retaining more original destination nodes of links with a larger retention probability, and (3) confining the randomized destination node of a link to a local neighborhood. Additionally, we also factor the popularity of destination nodes through a privacy notion of a destination node relative to this popularity. To our best knowledge, this is the first work that preserves structural properties through link destination perturbation within a local subgraph and factors the popularity of destination nodes in privacy of such nodes.

8. ACKNOWLEDGMENT

The authors would like to thank reviewers of EDBT 2012 conference for their feedback. The work of Ke Wang was supported by Canada NSERC Discovery Grant, and the work of Philip S. Yu was supported by US NSF through grants CNS-1115234, and IIS-0914934, and Google Mobile 2014 Program.

9. REFERENCES

[1] C. C. Aggarwal, and P. S. Yu. Privacy-preserving data mining: models and algorithms. Vol. 34, Advances in Database Systems, Springer, 2008

[2] R. Agrawal, R. Srikant, and D. Thomas. Privacy preserving OLAP. In SIGMOD 2005

[3] L. Backstrom, C. Dwork, and J. M. Kleinberg. Wherefore art thou R3579X?: anonymized social networks, hidden patterns, and structural steganography. In WWW, 2007

[4] K. Bai, Y. Liu, and P. Liu. Prevent identity disclosure in social network data study. In ACM CCS 2009

[5] A.-L. Barabasi, R. Albert. Emergence of scaling in random networks. In Science, Vol 286, 509–512, 1999

[6] F. Bonchi, A. Gionis, and T. Tassa. Identity obfuscation in graphs through the information theoretic lens. In ICDE 2011.

[7] S.P. Borgatti, M.G. Everett, and L.C. Freeman. Ucinet for windows: software for social network analysis. Harvard, MA: Analytic Technologies 2002.

[8] R.Chaytor, and K.Wang.Small domain randomization: same privacy, more utility. In VLDB 2010

[9] J. Cheng, A. W. Fu, and J. Liu. K-isomorphism: privacy preserving network publication against structural attacks. In SIGMOD 2010

[10] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In PODS 2003

[11] B. C. M. Fung, K. Wang, A. W.-C. Fu, and P. S. Yu. Introduction to privacy-preserving data publishing: concepts and techniques. Data Mining and Knowledge Discovery Series. Chapman & Hall/CRC, 2010

[12] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt and A. Arenas. Self-similar community structure in a network of human interactions. In Physical Review E, Vol. 68, 2003

[13] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava. Anonymizing social networks. Technical report, University of Massachusetts Amherst, 2007

[14] G. Karypis, V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. In SIAM Journal on Scientific Computing, Vol. 20, 1999

[15] K. Liu, and E. Terzi. Towards identity anonymization on graphs. In ACM SIGMOD/PODS 2008

[16] M. E. J., Newman. The structure of scientific collaboration networks. In Proc. of the National Academy of Sciences of the USA, Vol. 98, No. 2. 404-409. 2001.

[17] S. Wasserman and K. Faust. Social network analysis: methods and applications. Cambridge University Press, 1994

[18] X. Ying, and X. Wu. Randomizing social networks: a spectrum preserving approach. In SDM 2008

[19] X. Ying and X. Wu. On link privacy in randomizing social networks. In PAKDD 2009

[20] L. Zhang, and W. Zhang. Edge anonymity in social network graphs. In IEEE Social Computing 2009

[21] E. Zheleva, and L. Getoor. Preserving the privacy of sensitive relationships in graph data. In PinKDD 2007

[22] E. Zheleva, and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In WWW 2009

[23] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In ICDE 2008

[24] L. Zou, L. Chen, and M. T. Ozsu. K-automorphism: a general framework for privacy preserving network publication. In VLDB 2009.