Identity Obfuscation in Graphs Through the Information Theoretic Lens

Francesco Bonchi[a,*], Aristides Gionis[b], Tamir Tassa[c]

[a] *Yahoo! Research, Avinguda Diagonal 177, 08018 Barcelona, Spain*
[b] *Department of Information and Computer Science, Aalto University, Finland*
[c] *Department of Mathematics and Computer Science, The Open University, Ra'anana, Israel*

Abstract

Analyzing the structure of social networks is of interest in a wide range of disciplines. Unfortunately, sharing social-network datasets is often restrained by privacy considerations. One way to address the privacy concern is to anonymize the data before publishing. Randomly adding or deleting edges from the social graph is one of the anonymization approaches that have been proposed in the literature. Recent studies have quantified the level of anonymity that is obtained by random perturbation by means of a-posteriori belief probabilities and, by conducting experiments on small datasets, arrived at the conclusion that random perturbation cannot achieve meaningful levels of anonymity without deteriorating the graph properties.

We offer a new information-theoretic perspective on the question of anonymizing a social network by means of random edge additions and deletions. We make an essential distinction between image and preimage anonymity and propose a more accurate quantification, based on entropy, of the anonymity level that is provided by the perturbed network. We explain why the entropy-based quantification, which is global, is more adequate than the previously used local quantification that was based on a-posteriori belief probabilities. We also prove that the anonymity level as quantified by means of entropy is always greater than or equal to the one based on a-posteriori belief probabilities. In addition, we introduce and explore the method of random sparsification, which randomly removes edges, without adding new ones.

Extensive experimentation on several very large datasets shows that randomization techniques for identity obfuscation are back in the game, as they may achieve meaningful levels of anonymity while still preserving properties of the original graph.

As the methods we study add and remove edges, it is natural to ask whether an adversary might use the disclosed perturbed graph structure to reconstruct, even partially, the original graph. We thus study the resilience of obfuscation by random sparsification to adversarial attacks that are based on *link prediction*. Given a general link prediction method, with a predefined level of prediction accuracy, we show how to quantify the level of anonymity that is guaranteed by the obfuscation. We empirically prove that even for very accurate link prediction methods, the level of anonymity guaranteed remains very close to the one before the attack.

Finally, we show how the randomization method may be applied in a distributed setting, where the network data is distributed among several non-trusting sites, and explain why randomization is far more suitable for such settings than other existing approaches.

Keywords: Anonymity, Data Publishing, Graphs, Social Networks, Randomization, Information Theory

*Corresponding author
Email addresses: bonchi@yahoo-inc.com (Francesco Bonchi), aristides.gionis@aalto.fi (Aristides Gionis), tamirta@openu.ac.il (Tamir Tassa)

1. Introduction

A social network is a graph structure holding information on a set of entities and the relations between them. Such information is of interest in a wide range of disciplines, including sociology, psychology, market research, and epidemiology. Very often social-network data cannot be published in their raw form since they contain sensitive information. The immediate first step to respect the privacy of individuals is to remove identifying attributes like names or social security numbers from the data. However, such a naïve anonymization is far from being sufficient. As shown by Backstrom et al. [2], the mere structure of the released graph may reveal the identity of the individuals behind some of the vertices. Hence, one needs to apply a more substantial procedure of sanitization on the graph before its release.

The methods for identity obfuscation in graphs fall into three main categories. The methods in the first category provide k-anonymity in the graph via edge additions or deletions [10, 33, 51, 61, 63]. The second category includes methods that add noise to the data in the form of random additions, deletions or switching of edges, in order to prevent adversaries from identifying their target in the network, or from inferring the existence of links between vertices [17, 22, 34, 50, 55, 56, 57]. The methods in the third category do not alter the graph data like the methods of the two previous categories; instead, they group together vertices into super-vertices of size at least k, where k is the required threshold of anonymity, and then publish the graph data in that coarser resolution [9, 12, 13, 20, 21, 42, 58].

In this paper we focus on the second category: changing the graph structure via random perturbations. Algorithms in this category usually utilize one of two graph perturbation strategies: random addition and deletion of edges, or random switching of edges. In the first strategy one randomly adds h non-existing edges after randomly deleting h existing edges; such techniques preserve the total number of edges in the graph [22, 55]. In the second strategy, one selects h quadruples of vertices $\{u, v, x, y\}$ where (u, v) and (x, y) are edges and (u, x) and (v, y) are not, and switches between them, so that the two former edges become non-edges and the two latter non-edges become edges [55, 56, 57]; such techniques preserve the degree of all vertices in the graph.

Hay et al. [22] investigated methods of random perturbation in order to achieve identity obfuscation in graphs. They concentrated on re-identification of vertices by their degree. Given a vertex v in the real network, they quantified the level of anonymity that is provided for v by the perturbed graph as $(\max_u\{\Pr(v \mid u)\})^{-1}$, where the maximum is taken over all vertices u in the released graph and $\Pr(v \mid u)$ stands for the belief probability that u is in fact the target vertex v. By performing experimentation on the Enron dataset, using various values of h (the number of added and removed edges), they found out that in order to achieve a meaningful level of anonymity for the vertices in the graph, h has to be tuned so high that the resulting features of the perturbed graph no longer reflect those of the original graph.

Those methods were revisited by Ying et al. [54], who compared this perturbation method to the method of k-degree anonymity due to Liu and Terzi [33]. They too used the a-posteriori belief probabilities to quantify the level of anonymity. Based on experimentation on two modestly sized datasets (Enron and Polblogs) they arrived at the conclusion that the deterministic approach of k-degree anonymity preserves the graph features better for given levels of anonymity.

Other authors have confirmed that topological features (e.g., clustering coefficient, or the largest eigenvalue of the adjacency matrix, that we also assess in Section 8.2) are "significantly lost in the randomized graph when a medium or high perturbation is applied" [50, 55]. Those studies too use the same modestly sized datasets (Enron and Polblogs).

1.1. Our contributions

We provide a new information-theoretic perspective on the strategy of random additions and deletions of edges in graphs. Our main contribution is showing that randomization techniques for identity obfuscation are back in the game, as they may achieve meaningful levels of obfuscation while still preserving characteristics of the original graph. We prove our claim by means of a principled theoretical analysis and a thorough experimental assessment. In particular, we make the following contributions:

Types of obfuscation. We introduce a fundamental distinction between two different forms of privacy and two corresponding measures of the level of anonymity achieved by the perturbed network. One measure is

called k-obfuscation, or *k-image obfuscation*, and it is a measure of privacy against an adversary who tries to locate in the perturbed graph the image of a specific individual. The second is called *k-preimage obfuscation*; it is a measure of privacy against an adversary who does not have any particular target individual, but she tries to examine the released graph and deduce the identity of any of the vertices in that graph.

More accurate measures of obfuscation. Our measures are defined by means of the entropy of the probability distributions that are induced on the vertices of the perturbed graph (in the case of k-obfuscation) or those which are induced on the vertices of the original graph (in the case of k-preimage obfuscation). This is in contrast to Hay et al. [22] who based their definition of k-candidate anonymity on a-posteriori belief probabilities. While the a-posteriori belief probability is a *local* measure that examines, for each vertex in the perturbed graph, the probability that the vertex originated from the target individual in question, the entropy is a *global* measure that examines the entire distribution of those belief probabilities. We explain and exemplify why the entropy-based measure is more accurate than the a-posteriori belief probability, where accuracy means that it distinguishes between situations that the other measure perceives as equivalent. Moreover, we prove that the obfuscation level quantified by means of the entropy is always no less than the one based on a-posteriori belief probabilities. We derive formulas to compute those entropies in the case where the background knowledge of the adversary is the degree of the target individual.

Extensive and large-scale experimentation. We conduct thorough experimentation on three large datasets and measure many properties of the perturbed graphs. We compare the distribution of the obfuscation levels as measured by our entropy-based measure to those measured by the previous one, and show that one may achieve meaningful levels of obfuscation while preserving the properties of the original graph.

Random sparsification. We also introduce the method of random sparsification, which only removes edges from the graph, without adding new ones. We compare it to random perturbation in terms of the trade-off between utility of the perturbed graphs and the levels of obfuscation that they provide. Somehow surprisingly, sparsification maintains better the characteristics of the graph than perturbation at the same anonymity levels. Indeed, removing an edge affects the structure of the graph to a smaller degree than adding an edge. This is partially due to the small-world phenomenon: adding random long-haul edges brings everyone closer, while removing an edge does not bring vertices so much apart as there usually exist alternative paths.

Resilience to link-prediction attacks. As the methods that we study add and remove edges, it is natural to ask whether an adversary might use the disclosed perturbed graph structure to even partially reconstruct the original graph. Estimating the likelihood of the existence of an edge between two vertices based on observed links is a well-studied data-mining task, called *link prediction* [32, 35]. In this paper we propose adversarial attacks based on link-prediction methods, and we formally study the resilience of perturbation methods to such attacks. In particular, for the sake of simplicity, we focus only on random sparsification. Given a general link-prediction method, with a predefined prediction accuracy, we show how to quantify the level of anonymity guaranteed by the obfuscation. Then, we empirically prove that even for very accurate link-prediction methods, more accurate than the state-of-the-art methods reported in the literature [35], the level of anonymity guaranteed remains very close to the one before the attack.

Secure distributed obfuscation protocols. In some settings, the network data is split between several data holders, or players. For example, the data in a network of email accounts, where two vertices are connected if they exchanged a minimal number of email messages, might be split between several email service providers. As another example, consider a transaction network where an edge denotes a financial transaction between two individuals; such a network would be split between several banks. In such settings, each player controls some of the vertices (his clients) and he knows only the edges that are adjacent to the vertices under his control. It is needed to devise distributed protocols that would allow the players to arrive at a sanitized version of the unified network, without revealing to them sensitive information on clients of other players. The recent survey by X. Wu et al. about privacy-preservation in graphs and social networks concludes by recommendations for future research in this emerging area [52]. One of the proposed directions is distributed privacy-preserving social-network analysis, which "has not been well reported in literature". The randomization method, as we explain later on, is most suitable for such settings and we use it in order to design a privacy-preserving protocol for anonymizing distributed social networks.

A preliminary version of this manuscript appeared previously in [8]. The present manuscript extends the preliminary version, mainly by including an analysis of resilience of obfuscation by random perturbation to attacks based on link-prediction methods, and by addressesing the problem of designing secure protocols for obfuscating distributed social networks.

1.2. Roadmap

The paper is organized as follows. In Section 2 we introduce the problem setting, we review the adversarial assumptions made by the relevant related literature, and we define the notion of k-anonymity for graphs. The methods of random sparsification and random perturbation are introduced in Section 3, while the concepts of k-obfuscation and k-preimage obfuscation are formalized in Section 4. In Section 5 we describe how to compute the probability distributions behind k-obfuscation and k-preimage obfuscation. In Section 6 we explicitly derive the formulas to measure the level of anonymity for the case in which the adversarial knowledge and the corresponding anonymity concept are based on the degrees. In Section 7 we discuss the case in which the property being traced is the neighborhood. In Section 8 we conduct our experimental analysis on three very large datasets, measuring a multitude of features of the perturbed graph. Our study on the resilience of obfuscation by sparsification to adversarial attacks based on link-prediction methods is presented in Section 9. Using the same three very large datasets used in the previous section, we empirically confirm that the randomization methods are indeed resilient to such attacks, even when assuming a very accurate link-prediction method. In Section 10 we study the distributed setting, deriving secure multiparty protocols for social network anonymization based on random perturbation methods. In Section 11 we summarize the most relevant related literature, and we conclude the discussion in Section 12.

Table 1 summarizes the notation that we use throughout the paper.

2. Preliminaries

Let $G = (V, E)$ be a simple undirected graph that represents some social network, i.e., each vertex v in V represents an individual and an edge (v, v') in E represents some relation between v and v'. The goal is to release the graph for the sake of public scrutiny while preserving the anonymity of the individuals, in the sense of limiting the ability of an adversary to re-identify vertices in the released graph.

2.1. Adversarial assumptions

When studying anonymity methods for preventing identity disclosure, it is commonly assumed that the adversary knows some structural property of the vertex representing the target individual in the real graph. Hence, if the adversary wishes to locate her target individual, Alice, in the anonymized graph, the adversary may use her prior knowledge of that structural property in order to do so. Anonymization methods aim at modifying the original graph to an anonymized graph in which the assumed property induces equivalence classes of size at least k, where k is the required level of anonymity.

Liu and Terzi [33] considered the case where the property that the adversary uses is the degree $d(v)$ of the vertex v. Namely, it is assumed that the adversary knows the degree of Alice, and armed with that knowledge the adversary embarks on a search for her in the network. The algorithms presented by Liu and Terzi [33] use edge additions, and possibly also deletions, in order to make the graph k-degree anonymous in the sense that every vertex in the graph has at least $k-1$ other vertices with the same degree. Those algorithms attempt to achieve k-degree anonymity with the minimal number of edge additions and deletions.

Zhou and Pei [61] assumed a stronger property; they considered the case where the adversary knows the distance-one neighborhood of the target vertex, $N(v_i)$, namely, the sub-graph that is induced by v_i and its immediate neighbors in G. Another enhancement of the degree property was proposed by Hay et al. [20, 21], and Thompson and Yao [45], where they considered a sequence of properties that could be used by the adversary. Assume that $\mathcal{H}_1(v)$ is the degree of v; then, if $t = \mathcal{H}_1(v)$, and $v_1, \ldots, v_t$ are the neighbors of v, the quantities $\mathcal{H}_{i+1}(v) = \{\mathcal{H}_i(v_1), \ldots, \mathcal{H}_i(v_t)\}$ are defined for all values of $i \geq 1$. So, while $\mathcal{H}_1$ is just the degree property, $\mathcal{H}_2$ is the property that consists of the degrees of the neighbors, $\mathcal{H}_3$ consists of the degrees of the neighbors of the neighbors, and so forth.

Table 1: Notation used in this paper.

$G = (V, E)$	the input undirected graph with set of vertices V and edges E
$n = \|V\|$	number of vertices
$m = \|E\|$	number of edges
P	a property of vertices
$\sim$	the equivalence relation induced by P on V
$d(\cdot)$	the degree property of vertices
p	probability with which edges are removed by the randomization process
q	probability with which edges are added by the randomization process
$G_p = (U, E_p)$	the randomized graph
$X_v(u)$	probability that $u \in U$ is the image of $v \in V$ in the randomized graph G_p
$X_u(v)$	probability that $v \in V$ is the preimage of $u \in U$ in the original graph G
$f(v; u)$	probability that $v \in V$ s.t. $P(v)$ was converted in $u \in U$ s.t. $P(u)$ by the randomization process
$\mathcal{W}_h(G_p)$	the set of all graphs that could have been the pre-image of G_p under that randomization model
$M_{\mathrm{LP}}(y, z)$	outcome of a link-prediction model M_{LP} for a pair of vertices (y, z) (Section 9)
η	false positive rate of M_{LP}
θ	false negative rate of M_{LP}
V^i	set of vertices under control of the ith player (Section 10)

Zou et al. [63] and Wu et al. [51] assumed that the adversary knows all of the graph G, and the location of v_i in G; hence, she can always identify v_i in any copy of the graph, unless the graph has other vertices that are automorphically-equivalent to v_i. The algorithms presented in those works ensure that every vertex in the released graph has at least $k - 1$ other vertices that are automorphically-equivalent to it.

2.2. Anonymization

Let $G = (V, E)$ be a graph and P be a property of the vertices in the graph, as exemplified above. That property induces an equivalence relation $\sim$ on V, where $v_i \sim v_j$ if and only if $P(v_i) = P(v_j)$. For example, for the degree property, $v_i \sim v_j$ if and only if $d(v_i) = d(v_j)$; or, in the case that was studied in [51, 63], $v_i \sim v_j$ if and only if they are automorphically-equivalent. Since P is assumed to hold all of the information that the adversary has regarding the target vertex, she cannot distinguish between two vertices in the same P-equivalence class. This motivates the following definition.

Definition 1. A graph G is called k-anonymous with respect to property P if all equivalence classes in the quotient set $V/\sim$ of the property P are of size at least k.

Given an integer k, one may transform the input graph G into a k-anonymous graph $G_A = (V_A, E_A)$ by adding or removing edges and vertices from G. For example, in the model of k-degree anonymity [33], the anonymization algorithm adds edges to the graph (or, in another version of the algorithm, adds and removes edges) until each degree in $G_A = (V_A = V, E_A)$ appears at least k times. In k-neighborhood anonymity [61], edges are added until each neighborhood in G_A appears at least k times. In k-symmetry anonymity [51, 63], new edges and vertices are added until every vertex in G_A has at least $k-1$ other vertices that are indistinguishable from it based on the graph structure, since they are all automorphically-equivalent.

By releasing such a k-anonymized graph, the adversary will not be able to track Alice down to subsets of vertices of cardinality less than k. Moreover, as all vertices in that set are equivalent in the eyes of the adversary (since they all share the same property that is the only weapon that the adversary has), they are all equally probable as being Alice. Hence, if we model the knowledge of the adversary regarding the location of Alice in V_A as a random distribution on V_A, where each vertex v in V_A has an associated probability of being Alice, given the a-priori knowledge of the adversary and the observed G_A, then k-type anonymity models dictate that the entropy of that random distribution is at least $\log_2 k$.

We proceed to propose a probabilistic version of that type of k-anonymity.

3. Obfuscation by randomization

In this paper we study identity obfuscation methods based on either random perturbation or random sparsification.

3.1. Obfuscation by random sparsification

Obfuscation by random sparsification is performed in the following manner. The data owner selects a probability $p \in [0,1]$ in a way that will be discussed later on. Then, for each edge e in E the data owner performs an independent Bernoulli trial, $B_e \sim B(1,p)$. He will remove the edge in the graph in case of success (i.e., $B_e = 1$) and will leave it otherwise ($B_e = 0$).

Letting $E_p = \{e \in E \mid B_e = 0\}$ be the subset of edges that passed this selection process, the data owner will release the subgraph $G_p = (U = V, E_p)$. The idea is that such a graph offers some level of identity obfuscation for the individuals in the underlying population, while maintaining sufficient utility in the sense that many features of the original graph may be inferred from looking at G_p.

The set of vertices in G_p will be denoted by U, even though it equals the set of vertices in G, which is denoted by V. The introduction of a different notation will be needed in our analysis later on, in order to distinguish between the set of vertices, as observed by the adversary in G_p, and the set of vertices in the original graph G.

3.2. Obfuscation by random perturbation

Obfuscation by random perturbation is a process that consists of two phases – edge deletions followed by edge additions. One way of performing this process is as follows: in the first phase the data owner selects an integer h and then he randomly picks a subset of h edges out of the m edges in E and removes them. In the second phase the data owner randomly picks h pairs of vertices that were not connected in E, and adds edges that connect them.

We consider here random perturbations that are obtained by a similar process using a sequence of Bernoulli trials. In the first phase the data owner selects a probability $p \in [0,1]$ and then, for each edge e in E, he removes it in probability p. In the second phase the data owner selects another probability q and then adds an edge e in $\binom{V}{2} \setminus E$ with probability q. Edges that were removed in the first phase are not considered as candidates for addition in the second phase. In order to guarantee that the expected number of edges in the resulting graph equals $m = |E|$, the probability q should be selected so that

$$(1-p)m + q\left(\binom{n}{2} - m\right) = m,$$

or equivalently,

$$q = q(p) = \frac{mp}{\left(\binom{n}{2} - m\right)}. \tag{1}$$

As we shall always set q to be a function of p through Equation (1), we shall denote the resulting randomized graph, as before, by $G_p = (U = V, E_p)$.

4. k-Obfuscation and k-preimage obfuscation

Here, we define our two privacy notions. The first one protects against adversaries who try to locate a specific individual in the randomized graph. The second one protects against a different type of adversarial attack which is not targeted against a specific individual.[1]

[1] A similar distinction was made by Gionis et al. [14] between $(1,k)$-anonymity and $(k,1)$-anonymity in the context of anonymizing databases by means of generalization.

4.1. k-Obfuscation

We assume hereinafter that the adversary knows the randomization method and the value of the selected randomization parameter p. The goal of the adversary is to locate the image in U of a specific vertex v in V. Due to randomization, the adversary cannot determine which of the vertices u in U is the image of v in V; however, based on her background knowledge and the observed G_p the adversary may associate a probability with every vertex u in U as being the sought-after image of v in V. Let us denote the corresponding random variable that is defined by v and G_p on U by X_v; namely, for every u in U, $X_v(u)$ is the probability that u is the image of v in G_p.

Definition 2 (k-Obfuscation). A perturbed graph G_p respects k-obfuscation if for every vertex v in V, the entropy of the random variable X_v over U is at least $\log k$.

We used the term obfuscation, rather than anonymity, because traditionally anonymity is associated with cases in which every item (vertex or record) in the released data (graph or table) belongs to an equivalence class of items of size at least k. Randomization does not produce such outputs, hence the different term.

Hay et al. [21] defined a different notion: k-candidate anonymity. Reformulated in our terms, a randomized graph offers k-candidate anonymity if

$$X_v(u) \leq \frac{1}{k}, \text{ for all } v \text{ in } V \text{ and } u \text{ in } U. \tag{2}$$

The logic behind that definition, as implied by the term *k-candidate* anonymity, is that condition (2) guarantees that for each vertex $v \in V$ there are at least k candidate vertices $u \in U$ that might be its image. Hence, k-candidate anonymity attempts at guaranteeing a lower bound on the amount of uncertainty that the adversary would have when she tries to locate the image of the target individual in the perturbed graph. However, we claim that this definition does not measure correctly the amount of uncertainty that the adversary has regarding the correct identification of the target individual. For example, such a definition does not distinguish between the following two situations:

(1) $X_v(u_1) = X_v(u_2) = \frac{1}{2}$, and
$X_v(u_i) = 0$ for all $3 \leq i \leq n$;
(2) $X_v(u_1) = \frac{1}{2}$,
$X_v(u_i) = \frac{1}{2t}$ for all $2 \leq i \leq t+1$, and
$X_v(u_i) = 0$ for all $t+2 \leq i \leq n$.

Both cases respect 2-candidate anonymity since the maximal probability in both is $\frac{1}{2}$. However, it is clear that in the second case, where there are $t+1$ suspects, the amount of uncertainty regarding the identity of the correct image of the target individual is much higher than in the first case. In addition, the efforts that are needed to complete the identification in the second case are higher, as there are more suspects to examine ($t+1$ instead of 2).

The correct way to measure uncertainty is the entropy. Indeed, the entropy distinguishes between the above two cases: by applying Definition 2 to those two cases, we find that the first one respects 2-obfuscation, while the second one respects $(2\sqrt{t})$-obfuscation.

The notion of ℓ-diversity was introduced, independently and under different terms, in [36] and [48]. In similarity to obfuscation, it is a measure of the amount of uncertainty that the adversary has regarding some sensitive attribute of the target individual. [36] based their definition on the entropy, while [48] based theirs on the maximal probability. The entropy-based definition of diversity as appears in [36] remains the more accepted and prevalent definition.

Proposition 1. *The obfuscation level of a given perturbed graph G_p is always greater than or equal to the corresponding candidate anonymity level.*

PROOF. Fix $v \in V$ and let $\mathbf{p}(v) = (p_1, \ldots, p_n)$ denote the probability distribution X_v; i.e., if the vertices in the perturbed graph are $U = \{u_1, \ldots, u_n\}$ then $p_i = X_v(u_i)$. The obfuscation level offered by G_p is then

$$k_o = \min_{v \in V} 2^{H(\mathbf{p}(v))},$$

(where $H(\mathbf{p}(v))$ is the entropy of the probability distribution $\mathbf{p}(v)$), while the candidate anonymity level is

$$k_c = \min_{v \in V} \left(\max_{p_i \in \mathbf{p}(v)} p_i \right)^{-1}.$$

For any fixed $v \in V$ we have

$$H(\mathbf{p}(v)) = \sum_i p_i \log\left(\frac{1}{p_i}\right) \geq \sum_i p_i \log\left(\frac{1}{\max p_i}\right) = \log\left(\frac{1}{\max_{p_i \in \mathbf{p}(v)} p_i}\right).$$

Therefore,

$$2^{H(\mathbf{p}(v))} \geq \left(\max_{p_i \in \mathbf{p}(v)} p_i \right)^{-1}, \text{ for all } v \in V.$$

The last inequality implies that $k_o \geq k_c$. □

In experiments conducted by Hay et al. [21] on the Enron dataset, it was shown that in order to achieve a reasonable k-candidate anonymity it is necessary to use values of the perturbation probability p for which most of the graph features (e.g., diameter, path length, closeness, betweenness) are severely altered. They concluded that randomness cannot achieve at the same time a reasonable privacy preservation and an acceptable loss of utility. Our claim here is that, in light of the observation that k-obfuscation is the correct measure, and that such a measure is satisfied by larger values of k than k-candidate anonymity, random perturbation is back in the game of privacy preservation in social networks.

4.2. k-preimage obfuscation

The definitions of k-obfuscation and k-candidate anonymity reflect the goal of protecting against an adversary who wishes to reveal sensitive information on a specific target individual. Those definitions aim to limit the possibility of the adversary to identify the image of the target individual in the released network. Such adversarial attacks are the ones that are more commonly discussed and analyzed [2, 24].

However, there is another type of adversarial attacks that one should consider: when the adversary is interested in re-identifying any entity in the released data (say, in order to find possible victims to blackmail). Such an attack works in the opposite direction: focusing on an item in the released (and presumably anonymized) corpus of data, the adversary tries to infer its correct preimage.

This kind of attack was also at the basis of the well-known AOL crisis. On August 4, 2006, AOL Research released a compressed text file containing twenty million search keywords for more than 650,000 users over a 3-month period, intended for research purposes. Even though all records were stripped of the identity of the person that made the query, certain keywords contained personally identifiable information. Since each user was identified on that list by a unique sequential key, it was possible to compile a search history for each given user. The New York Times [3] was able to locate an individual from the released and anonymized search records by cross referencing them with phonebook listings. These considerations motivate the following definition.

Definition 3 (k-Preimage Obfuscation). Let $G = (V, E)$ and $G_p = (U, E_p)$ be an original and perturbed graphs, respectively. For each $u \in U$ let X_u denote the corresponding random variable that is defined on V, i.e., $X_u(v)$ is the probability that v is the preimage of u in G. Then the perturbed graph G_p respects k-preimage obfuscation if for every vertex $u \in U$, the entropy of the random variable X_u on V is at least $\log k$.

Similarly, we define k-preimage candidate anonymity as follows: a randomized graph offers k-preimage candidate anonymity if

$$X_u(v) \leq \frac{1}{k}, \text{ for all } v \text{ in } V \text{ and } u \text{ in } U.$$

5. Quantifying the level of obfuscation

The definitions in the previous section involved two ensembles of probability distributions: $\{X_v(\cdot) : v \in V\}$, defined on U, for k-obfuscation, and $\{X_u(\cdot) : u \in U\}$, defined on V, for k-preimage obfuscation. The randomized graph G_p respects k-obfuscation (resp. k-preimage obfuscation) if the entropy of each of the probability distributions in the first (resp. second) ensemble is greater than or equal to $\log k$. Here we describe how to compute those distributions. We separate the discussion to the two notions of obfuscation.

5.1. Verifying k-obfuscation

Let $P(\cdot)$ denote the property that the adversary knows about the target vertex $v \in V$. By looking at the released graph G_p and the values of $P(u_i)$ for each of the vertices $u_i \in U$ in G_p, the adversary may associate a probability $X_v(u)$ for the event that the vertex $u \in U$ is the image of v.

Let v and u be vertices in V and U respectively. Let $f(v;u)$ be the probability that a vertex with property $P(v)$ was converted, under the known randomization model, to a vertex with property $P(u)$. For example, in the case of the degree property, $P(\cdot) = d(\cdot)$, $f(v;u)$ is the probability that a vertex with degree $d(v)$ was converted to a vertex with degree $d(u)$, given the method and parameters of randomization. For the sake of illustration, if the method of randomization was sparsification, then $f(v;u) = 0$ whenever $d(v) < d(u)$, since by only deleting edges it is impossible that the degree of a vertex would increase.

As discussed earlier, the property P induces an equivalence relation, denoted $\sim$, on both V and U. Therefore, we may compute the probabilities $f(v;u)$ only on the Cartesian product of the two equivalence classes, $(V/\sim) \times (U/\sim)$. Those computed values would give the values of $f(v;u)$ for all $v \in V$ and $u \in U$. We arrange those values in an $n \times n$ matrix F where $F_{i,j} = f(v_i;u_j)$, $1 \le i,j \le n$. Each row in this matrix corresponds to an original vertex $v \in V$ and gives the related probabilities $f(v;u)$ for all $u \in U$. Similarly, each column corresponds to an image vertex $u \in U$. The matrix F enables us to compute the probability distributions $X_v(\cdot)$, for all $v \in V$, by looking at its rows. The probability distributions are obtained by normalizing the corresponding row in the matrix F:

$$X_{v_i}(u_j) = \frac{F_{i,j}}{\sum_{1 \le \ell \le n} F_{i,\ell}}, \quad 1 \le i,j \le n. \tag{3}$$

For example, if in the randomization process we use $p = 0$, then $G_p = G$. In that case, $f(v;u) = 1$ if $P(v) = P(u)$ and $f(v;u) = 0$ otherwise. Therefore, for any given $v \in V$, the entries in the corresponding row in F would be 1 in columns that correspond to vertices $u \in U$ with $P(u) = P(v)$, and 0 otherwise, since by using $p = 0$ (no randomization), the properties of the vertices remain unchanged. For each $v \in V$, the set of probabilities $\{f(v;u) : u \in U\}$ is then converted into a probability distribution by means of normalization. If there are ℓ vertices $u \in U$ for which $f(v;u) = 1$, then each one of them is the image of v with probability $1/\ell$. In that case, $X_v(\cdot)$ associates the probability $1/\ell$ for each of those ℓ vertices in U, and zero to the remaining $n - \ell$ vertices.

To illustrate that, assume that G and G_p have 7 vertices and $p = 0$ (so that $G_p = G$). Assume that the degree sequence in the graph is $(2,2,2,3,4,4,5)$. Then Figure 1 gives the matrix F is this case; the first column indicates the vertices in G and their degrees, while the first row indicates the vertices in G_p and their degrees. Take, for instance, the row of v_1. It has four 0s and three 1s. Then, by dividing the entries of that row by 3, we infer that each of the vertices u_1, u_2, u_3 is the image of v_1 with probability 1/3, while u_4, u_5, u_6, u_7 cannot be the image of v_1.

The computation of the probability distributions $X_v(\cdot)$ that was carried out by Hay et al. [22] was different, but the basic idea is similar. In that study, the randomization was made by first removing h of the existing edges, thus arriving at an interim graph, and then adding h of the edges that do not exist in the interim graph, thus arriving at G_p. Given a graph G_p, the set of possible worlds $\mathcal{W}_h(G_p)$ is the collection of all graphs over the same set of vertices that could have been the pre-image of G_p under that randomization model. Each of the graphs G in $\mathcal{W}_h(G_p)$ was associated with a probability, $\Pr(G)$. Then, they defined $f(v;u)$ as the sum of $\Pr(G)$ for all possible worlds $G \in \mathcal{W}_h(G_p)$ in which the property of the candidate

	u_1:2	u_2:2	u_3:2	u_4:3	u_5:4	u_6:4	u_7:5
v_1:2	1	1	1	0	0	0	0
v_2:2	1	1	1	0	0	0	0
v_3:2	1	1	1	0	0	0	0
v_4:3	0	0	0	1	0	0	0
v_5:4	0	0	0	0	1	1	0
v_6:4	0	0	0	0	1	1	0
v_7:5	0	0	0	0	0	0	1

Figure 1: Example of the matrix F.

vertex u equals the known property of the target vertex v. Finally, they normalized $f(v;u)$ in the same way that we did in order to get a probability distribution $X_v(\cdot)$ on U.[2]

5.2. Verifying k-preimage obfuscation

Assume that the adversary knows the value of the property $P(\cdot)$ for all vertices $v \in V$. Then by looking at the released graph G_p and fixing $u \in U$, he may associate, for every $v \in V$, a probability $X_u(v)$ for the event that v is the preimage of u. Namely, every $u \in U$ induces a probability distribution on V.

Let u and v be vertices in U and V respectively. Let $f'(v;u)$ be the probability that a vertex with property $P(u)$ originated from a vertex with property $P(v)$. As discussed earlier, we may compute $f'(v;u)$ on the Cartesian product $(V/\sim)\times(U/\sim)$ of the two equivalence classes, and then construct an $n\times n$ matrix F' where $F'_{i,j} = f'(v_i;u_j)$, $1 \le i,j \le n$. That matrix enables us to compute the probability distributions $X_u(\cdot)$, for all $u \in U$, by looking at its columns. The probability distributions are obtained by normalizing the corresponding column in the matrix F':

$$X_{u_j}(v_i) = \frac{F'_{i,j}}{\sum_{1\le\ell\le n} F'_{\ell,j}}, \quad 1 \le i,j \le n. \tag{4}$$

Considering the example that was given above in Section 5.1, if we normalize the column of u_5 in Figure 1 we infer that u_5 originated from v_5 or v_6 with probability 1/2 each, and could not have originated from any of the other vertices.

We proceed to derive explicit formulas for the case where the property is the degree. The goal is to arrive at a condition that p has to satisfy so that G_p will be k-obfuscated or k-preimage obfuscated. In Section 7 we discuss stronger properties.

6. *k*-degree obfuscation

Here we consider the case of the degree property $P(\cdot) = d(\cdot)$ and derive explicit formulas for $f(\cdot;\cdot)$ and $f'(\cdot;\cdot)$. From those values one may compute the levels of k-degree obfuscation and k-degree preimage obfuscation as described in the previous section. Hereinafter, if $v \in V$ and $u \in U$, and u is the image of v, we denote this relation by $v \mapsto u$.

Let $v \in V$ be a target vertex in V and assume that the adversary knows that its degree is $d(v) = a$. Let $u \in U$ be a candidate vertex in U whose degree is $d(u) = b$. Then $f(v;u)$ equals the following conditional probability,

$$f(v;u) = \Pr(d(u) = b \mid d(v) = a, v \mapsto u); \tag{5}$$

[2]The probability distribution that was associated by Hay et al. [22] to every possible world was uniform. However, this should be corrected since some possible graphs may be converted to the observed randomized graph G_p in more than one way. Specifically, if the intersection of the edge set in the possible graph G with the edge set in G_p is of size $m - h + j$, where $0 \le j \le h$, then the number of different ways in which G could be converted to G_p is $\binom{m-h+j}{j}$. As a result, the probability of the possible world G is $\Pr(G) = \binom{m-h+j}{j} / \left[\binom{m}{h}\binom{\binom{n}{2}-m+h}{h}\right]$.

namely, given that v has a degree a and its image in G_p is u, $f(v;u)$ is the probability that u's degree is b. (In order to avoid cumbersome notations we shall drop the notation $v \mapsto u$ from the conditional probabilities henceforth; it is assumed always that v and u are a preimage and image pair.)

Under the random sparsification approach, $b \sim B(a, 1-p)$, where $B(a, 1-p)$ is the Binomial distribution over a experiments and success probability $1-p$. Under the random perturbation approach, $b \sim B(a, 1-p) + B(n-1-a, q(p))$. Hence, in the first model of random sparsification,

$$\Pr(d(u) = b \mid d(v) = a) = \begin{cases} \binom{a}{b}(1-p)^b p^{a-b} & b \leq a \\ 0 & b > a \end{cases} \tag{6}$$

As for the second model of random perturbation, let us denote by t the number of real edges adjacent to v that survived the randomization. Hence, $b-t$ is the number of phantom edges that were added by the random perturbation. Clearly, $t \leq a$ and $t \leq b$. Therefore, the conditional probability is given by

$$\Pr(d(u) = b \mid d(v) = a) = \sum_{t=0}^{\min\{a,b\}} \binom{a}{t}(1-p)^t p^{a-t} \binom{n-1-a}{b-t} q^{b-t}(1-q)^{n-1-a-b+t}\,. \tag{7}$$

Let

$$F(v) = \sum_{u \in U} f(v;u), \tag{8}$$

where $f(v;u)$ is given by Equations (5)+(6) or (5)+(7). Then

$$X_v(u) = \frac{f(v;u)}{F(v)}, \quad u \in U.$$

Hence, the level of obfuscation which is provided by using p is $k_o = \min_{v \in V} 2^{H(X_v)}$, while the candidate anonymity level is $k_c = \min_{v \in V} (\max X_v)^{-1}$.

Next, we turn to compute $f'(v;u)$, which is the inverse conditional probability. Assume that $d(v) = a$ and $d(u) = b$. Then

$$f'(v;u) = \Pr(d(v) = a \mid d(u) = b); \tag{9}$$

namely, given that v is the preimage of u and that the degree of u in G_p is b, $f'(v;u)$ is the probability that the degree of v is a. By Bayes Theorem,

$$\Pr(d(v) = a \mid d(u) = b) = \frac{\Pr(d(v) = a)}{\Pr(d(u) = b)} \cdot \Pr(d(u) = b \mid d(v) = a). \tag{10}$$

The value of $\Pr(d(v) = a)$ may be obtained from the frequencies of the degrees in G. The probabilities $\Pr(d(u) = b)$ are computed as follows:

$$\Pr(d(u) = b) = \sum_a \Pr(d(u) = b \mid d(v) = a) \Pr(d(v) = a). \tag{11}$$

Hence, the value of $f'(v;u)$ is given by Equations (9)–(11), together with Equations (6) or (7) which give the conditional probability $\Pr(d(u) = b \mid d(v) = a)$ in the two randomization models.

To summarize, we derived here the values of $f(v;u)$ and $f'(v;u)$ in the case of $P(\cdot) = d(\cdot)$. Those values enable to compute the probability distributions $X_v(\cdot)$ on U, for all $v \in V$, and $X_u(\cdot)$ on V, for all $u \in U$. The data owner needs to select a randomization parameter p such that the entropy of X_v is greater than or equal to $\log k$ for all $v \in V$ (in order to satisfy k-obfuscation), or a similar condition for the X_u, $u \in U$, probability distributions on V for the sake of k-preimage obfuscation. The goal is to select the smallest value of p (i.e., least amount of random perturbations) for which the required level of obfuscation is met, in order to retain as much as possible of the properties of the original graph. The minimal value of p that still respects k-obfuscation (or k-preimage obfuscation) may be found approximately by numerical means.

The complexity of the above described computation of the minimal entropy of X_v for all $v \in V$ is analyzed as follows. First, it is necessary to find the sets of degree values, denoted A and B, that are present in G and in G_p, respectively; that computation has complexity $O(n)$. Then, for each pair of degrees $(a, b) \in A \times B$ we need to compute the corresponding probability $\Pr(d(u) = b \mid d(v) = a)$ as given in Equation (6) or (7), depending on the perturbation model; the complexity of that computation depends only on $|A| \cdot |B|$, a value which is typically much smaller than n. Then, for a given $v \in V$, the computation of the entropy of $X_v(\cdot)$ on U can be done in time $O(|B|)$; indeed, even though that is a probability distribution over n vertices, it has only $|B|$ distinct probability values, and, consequently, the complexity of computing its entropy is only $O(|B|)$. Finally, that computation has to be repeated $|A|$ times for each of the degrees which are present in G. Therefore, the overall complexity of the computation that is needed to verify compliance with the k-obfuscation requirement is $O(n) + O(|A| \cdot |B|)$, what renders that computation practical even for very large graphs. A similar analysis applies also for computing the minimal entropy of X_u for all $u \in U$, which is needed in order to verify compliance with k-preimage obfuscation.

7. *k*-neighborhood obfuscation

Here we discuss briefly the case in which the traceability property is the neighborhood, $P(\cdot) = N(\cdot)$. In order to compute the probability distributions $X_v(\cdot)$, $v \in V$, on U, in this case, we need to compute, for every pair of neighborhoods η and θ, the conditional probability $\Pr(N(u) = \theta \mid N(v) = \eta, v \mapsto u)$. This is the probability that a vertex $v \in V$ that has a neighborhood η is converted under the randomization model to a vertex $u \in U$ with a neighborhood θ. In the case of randomization by sparsification, it is necessary to find all possible embeddings of θ in η, since there could be many ways in which η could be transformed into θ, to compute the probability of each such transformation and then add them up. Such a computation seems intricate even for moderately sized η and θ. The situation becomes more intricate in the case of random perturbation. Here, any neighborhood η could be converted to any neighborhood θ since any edge can be potentially removed and any non-edge (in the entire graph) can be potentially added.

Hence, it seems hard to measure precisely the level of obfuscation that is achieved when the property is not a simple one like the degree. The same difficulty also prevents the adversary from associating probabilities to the different vertices in the released graph as the possible images of the target vertex. Moreover, as opposed to the degree property, in order to perform such computations in the perturbation model, the adversary would need to know the structure of the entire graph, since even two far-apart vertices may become neighbors in that randomization model.

8. Experiments

In this section we report our experimental assessment of the effects of random sparsification and perturbation on the structure of the perturbed graph, as well as of the level of anonymity achieved, according to both k-obfuscation and k-preimage obfuscation notions. In all of the experiments we assume an adversary that uses the degree as the re-identification property, and that knows the randomization method and the value of the randomization parameter p.

8.1. Datasets

We use three large real-world datasets – `dblp`, `flickr`, and `Y360`.

- `dblp`: we extract a co-authorship graph from a recent snapshot of the DBLP database that considers only the journal publications. There is an undirected edge between two authors if they have coauthored a journal paper.
- Flickr is a popular online community for sharing photos, with millions of users. In addition to many photo-sharing facilities, users are creating a social network by explicitly marking other users as their *contacts*. In our `flickr` dataset, vertices represent users and edges represent the contact relationship.

Table 2: Dataset characteristics

Dataset	n	m	d	Δ	α
`dblp`	226 413	716 460	6.32	238	2.8
`flickr`	588 166	5 801 442	19.72	6 660	1.9
`Y360`	1 226 311	2 618 645	4.27	258	2.3

- Yahoo! 360 was a social networking and personal communication portal. That dataset models the friendship relationship among users. Among the three datasets, `Y360` is the sparsest one.

The main characteristics of the datasets are provided in Table 2, where n is the number of vertices, m is the number of edges, d is the average degree, Δ is the maximum degree, and α is the coefficient of fitting a power law in the degree distribution.

8.2. Graph statistics

The first objective of our experimental evaluation is to show that the method of randomized anonymization leaves to a large extent the structure of the original graph intact. Our strategy is to measure certain graph statistics on the original graph, on the anonymized graphs, and on random graphs. We then expect that for the anonymized graphs, the statistics under consideration are closer to those statistics on the original graph than to those on the random graph. The statistics that we measure are the following:

- *Clustering coefficient*: The fraction of closed triplets of vertices among all connected triplets;
- *Average distance*: The average distance among all pairs of vertices that are path-connected;
- *Diameter*: The maximum distance among all path-connected pairs of vertices;
- *Effective diameter*: The 90-*th* percentile distance among all path-connected pairs of vertices (i.e., the minimal value for which 90% of the finite pairwise distances in the graph are no larger than); and
- *Epidemic threshold*: To be defined in detail later.

We also report experiments based on graph clustering. We ran the METIS graph-clustering algorithm [27] with a prefixed number of clusters k on the original graph and on the perturbed one and we report their *Rand Index*, which is a measure of clustering similarity.[3] In Figure 2 we report all of the above statistics, for different values of p on the `dblp` and `Y360` datasets. We compare the graph obtained by sparsification and perturbation with a random Erdõs-Rényi graph containing the same number of edges and averaged over 50 random runs, and with the original graph. For the perturbation approach, the value of q is defined as a function of p according to Equation (1) in Section 3.

In all plots we report six values per curve, corresponding to $p = 2^i/100$ for $i = 0, \ldots, 5$.

We next describe the results following the order of plots in Figure 2 from left to right and from top to bottom.

Clustering coefficient. The first two plots of Figure 2 show that in both datasets, sparsification and perturbation do not lose much in terms of clustering coefficient for small values of p. When p grows to 0.16 the loss starts to be more substantial. In both datasets, sparsification preserves better the clustering coefficient.

Average distance, diameter and effective diameter. The next six plots of Figure 2 show that sparsification obviously increases distances among pairs of vertices. Perturbation, on the other hand, drastically reduces the diameter of the network, since the addition of random long-haul edges brings everyone closer.

[3] `http://en.wikipedia.org/wiki/Rand_index`

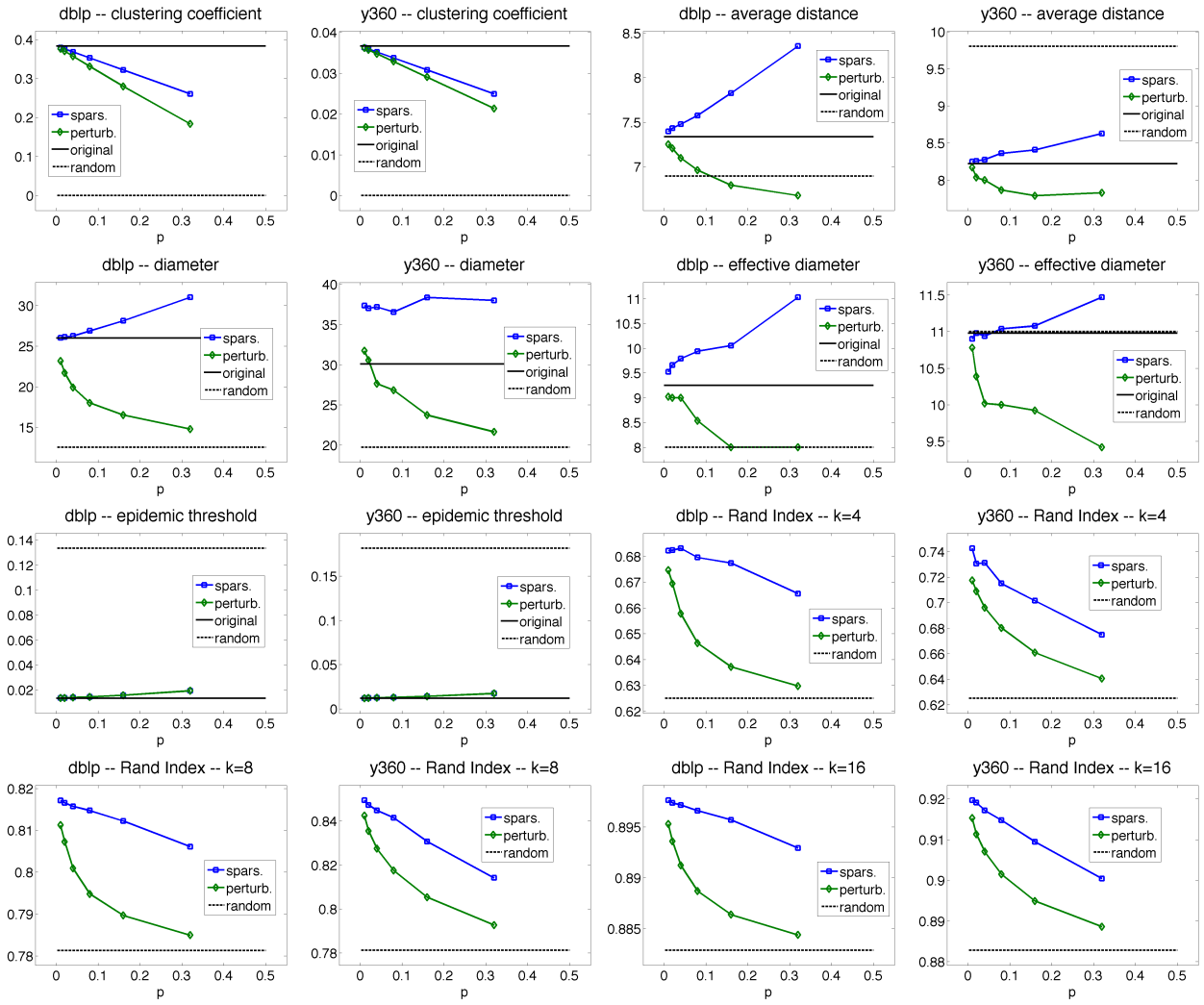

Figure 2: Effect of randomization on the `dblp` and `Y360` datasets.

Overall, sparsification preserves the graph distances better than perturbation does, with the exception of the diameter in `Y360`. However, it should be noted that in the effective diameter (which is a measure that is smoother and more robust to noise) sparsification performs very well especially for reasonably low values of p.

Epidemic threshold. The study of epidemic properties is an interesting way to characterize complex networks. In this scenario we assume that a virus propagates in the graph, according to some virus-propagation model, and infects vertices in the graphs. It turns out that for many virus-propagation models of interest, the graph can be characterized by a quantity called *epidemic threshold*, and the epidemic exhibits a threshold phenomena: if a certain parameter of the virus propagation model exceeds the epidemic threshold, then the virus spreads to all vertices of the graph, otherwise it dies out. In certain virus-propagation models [46], it can be shown that the epidemic threshold is λ_1^{-1}, where λ_1 is the largest eigenvalue of the adjacency matrix of the graph. This is the definition of epidemic threshold that we use here.

The values of epidemic threshold for `dblp` and `Y360` datasets are reported in the first half of the third row of Figure 2. The results are very intuitive: real graphs have very low epidemic thresholds; essentially

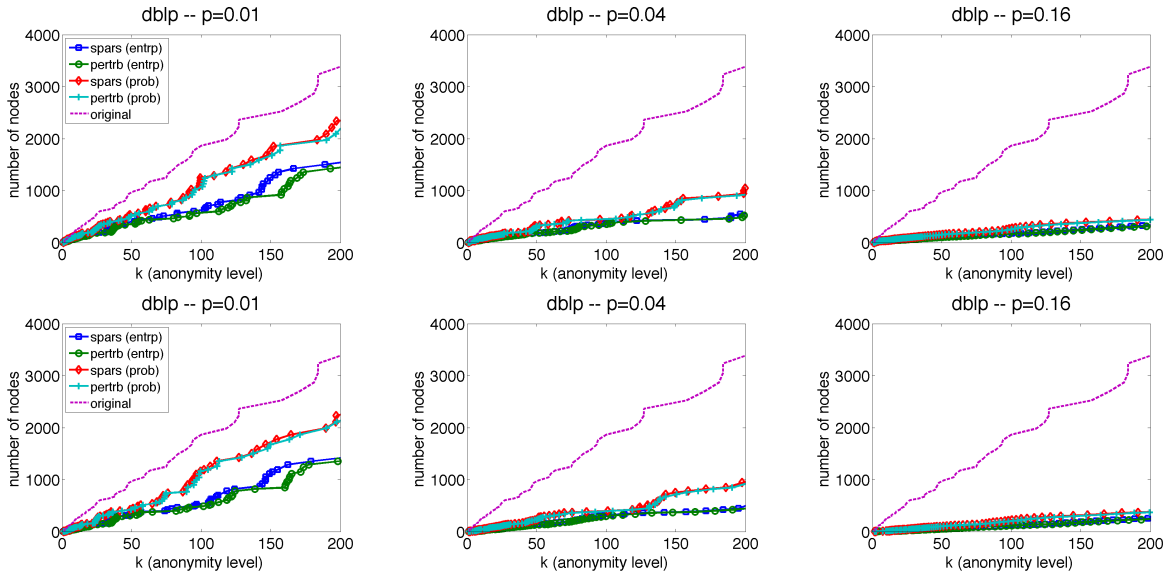

Figure 3: Anonymity levels of k-obfuscation (top) and k-preimage obfuscation (bottom) on the `dblp` dataset. The x-axis shows the anonymity level and the y-axis shows the number of vertices that *do not* reach the corresponding anonymity level. The plots report those values for the original graph, and for the sparsified and perturbed graphs.

epidemic outbreaks are likely because of the existence of hubs. On the other hand, random graphs have high tolerance to epidemic outbreaks. Perturbation and sparsification both produce graphs that have epidemic thresholds which are very close to that of the original graph.

Clustering similarity. We also asses the extent to which the results of a data mining analysis are sensitive to the perturbation process. In the last two rows of Figure 2 we report the similarity measure between a clustering obtained on the original graph and a clustering of the perturbed one. We can observe that both perturbation and sparsification perform well for small values of p, being always above 0.8 similarity, but as p grows, perturbation has a larger effect on clustering similarity than sparsification does.

8.3. Anonymity level

So far we have discussed the extent to which the structure of the graph is preserved for some given values of p. The next question is the level of anonymity which is achieved for the same values of p.

Figures 3 and 4 report this information for `dblp` and `Y360` datasets respectively. The two rows in the plots refer to the two adversarial models that we discussed. The top row reports the achieved levels of k-obfuscation and k-candidate anonymity, while the bottom row reports the achieved levels of k-preimage obfuscation and k-preimage candidate anonymity. Specifically, for a given value of p, we have on the x-axis the anonymity level and on the y-axis the number of vertices that *do not* reach such anonymity level. The plots report those values for the original graph, and for the sparsified and perturbed graphs. The curves that correspond to obfuscation levels are marked by `(entrp)` (since they are based on the entropy) while those that correspond to candidate anonymity are marked by `(prob)` (since they are based on the probability values).

We can draw three observations from these plots:

(1) Both perturbation and sparsification provide good privacy protection already at very low values of p.

(2) As proved in Proposition 4.2, the obfuscation level measured by entropy is always larger than the obfuscation level measured by probability.

(3) Although, in general, given a graph G and a randomization G_p, the corresponding level of image obfuscation is usually different from the corresponding level of preimage obfuscation, it turns out that in practice the plots for k-obfuscation and k-preimage obfuscation levels tend to be very similar.

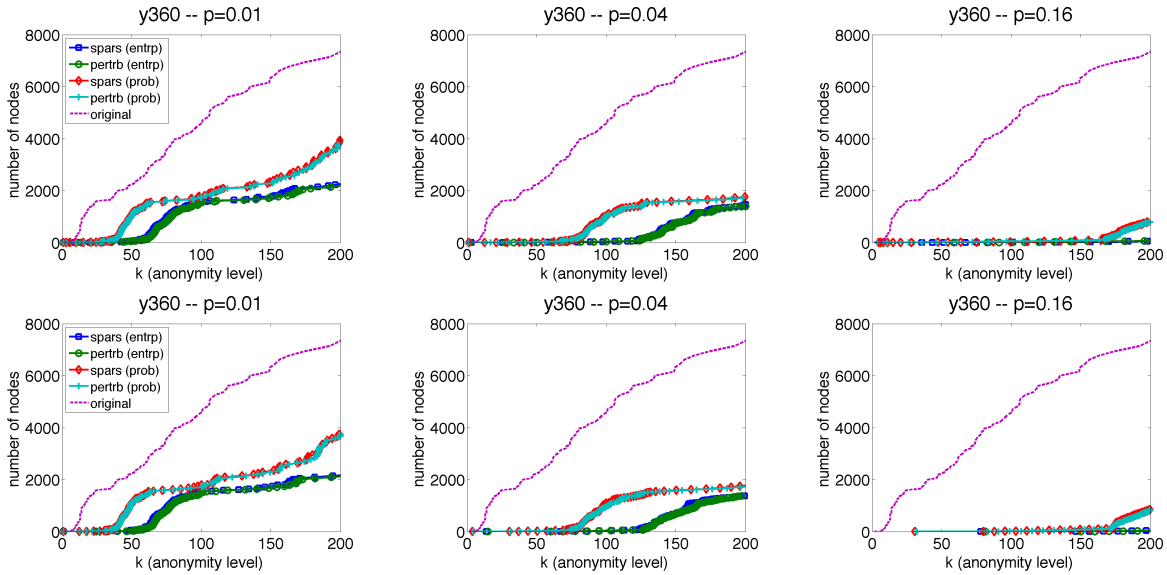

Figure 4: Anonymity levels of k-obfuscation (top) and k-preimage obfuscation (bottom) on the `Y360` dataset.

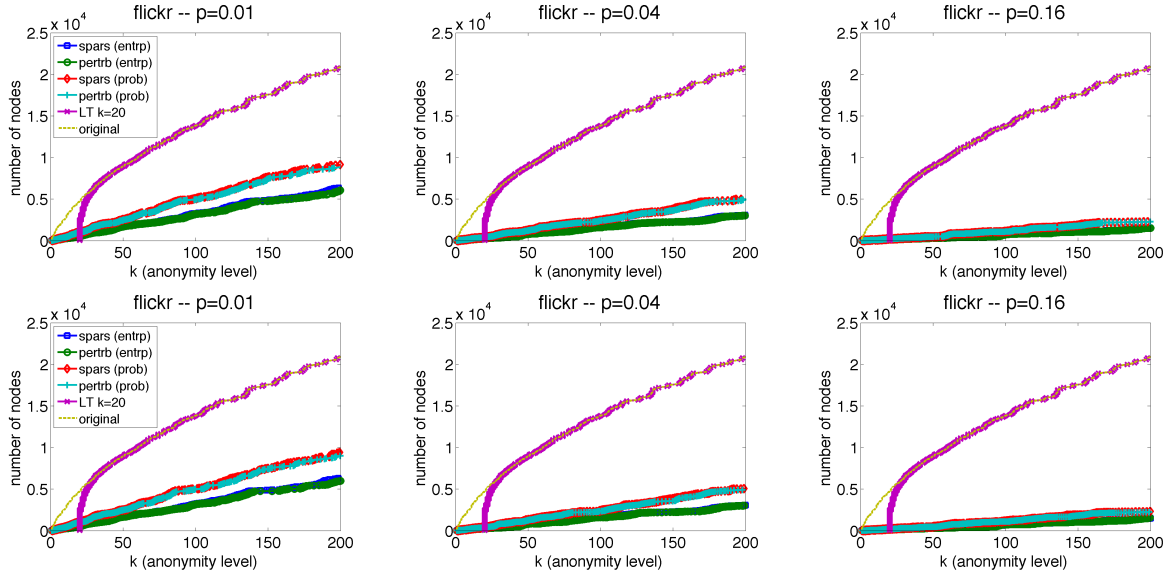

Figure 5: Anonymity levels of k-obfuscation (top) and k-preimage obfuscation (bottom) on the `flickr` dataset.

8.4. Comparison with methods of degree anonymity

Here we present a comparison with the method for k-degree anonymity by Liu and Terzi [33] (LT hereinafter). This set of experiments is conducted on the `flickr` dataset and the results are reported in Figures 5 and 6. In all of the experiments we assume an anonymity parameter $k = 20$ for the LT method.

We recall that the objective of the LT method is to find the minimum number of edge additions and deletions so that the resulting graph is k-degree anonymous. To understand better the behavior of the LT method, we consider a typical real-world graph whose degree distribution follows a power law. We observe that for such a typical power law graph, almost all vertices satisfy already the k-degree anonymity require-

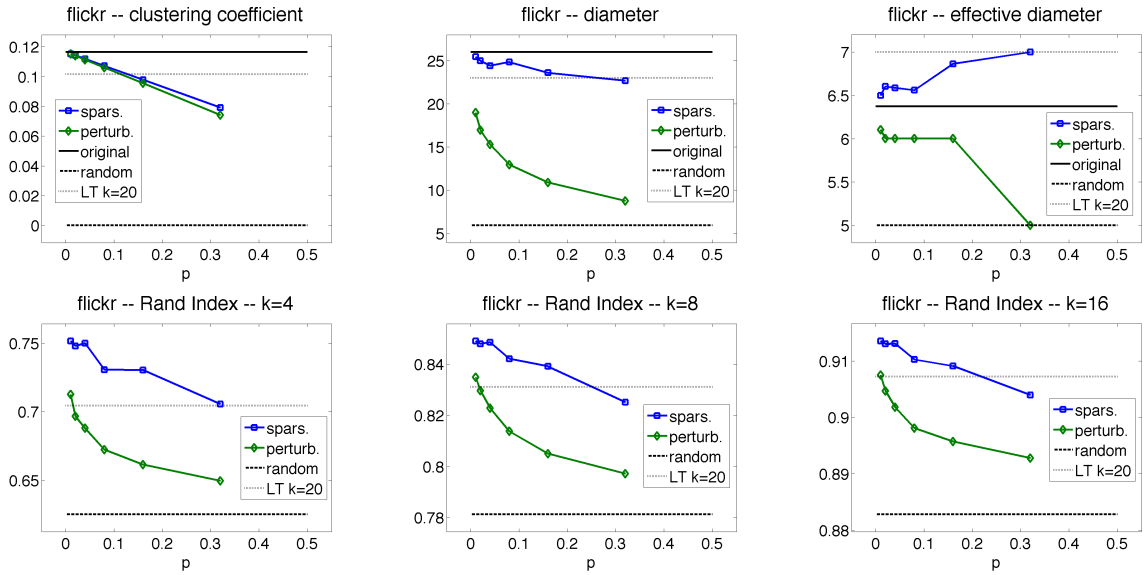

Figure 6: Effect of randomization on the `flickr` dataset.

ment, with the possible exception of a small number of hubs. Thus, the LT algorithm has only to "adjust" the degrees of those hubs (possibly by adding or deleting edges) while leaving the majority of the other vertices unaffected. Therefore, LT is able of achieving k-degree anonymity with a very small number of edge additions and deletions. Even though the resulting graph satisfies the k-degree anonymization requirement, one may argue that leaving the majority of the vertices unaffected has serious privacy implications. For example, the resulting graph is still vulnerable to the attack of Backstrom et al. [2].

In contrast, the randomized algorithm affects all vertices in the graph and thus it offers stronger anonymization properties. In a sense, the randomized anonymization destroys not only degree structure, but also higher neighborhood structures. Unfortunately, as we pointed out in Section 7, quantifying the anonymity level for higher neighborhood structures seems a computationally infeasible problem (for the data owner as well as for the adversary).

Our intuition is made clearer by studying the plots in Figure 5. We observe that all vertices in the output graph of the LT method satisfy k-degree anonymity with $k = 20$; however, when taking higher values of k, the curve of anonymity levels in the LT-anonymized graph quickly converges to the curve of anonymity levels in the original graph. On the other hand, the randomization methods provide significantly better levels of anonymization for all vertices in the graph, with the exception of the very small minority of vertices whose original anonymization level is smaller than 20.

Nevertheless, as both LT and our randomization methods adopt the k-degree adversarial assumption, it is interesting to compare to which extent they maintain the graph structure. In Figure 6 we can observe that for reasonably small values of p, sparsification always maintains the structure of the graph better than LT with $k = 20$. For instance, for $p = 0.04$ (third point in the plots) sparsification maintains very well the graph properties, while providing enough obfuscation as reported in the second column of Figure 5.

In summary, randomization methods and the LT method have different objectives, and they operate on different ends of the anonymization spectrum, thus comparing the two methods in a fair way seems very tricky. It would be interesting to combine the two techniques; i.e., to apply first the randomization method in order to obfuscate all detectable structures in the entire graph (and not only the degrees), and then apply the LT method only for the hubs, in order to provide also for them a minimal level of degree-anonymity.

9. Resilience of obfuscation by sparsification to link-prediction attacks

As the methods that we study add and remove edges, it is natural to ask whether an adversary might use the disclosed perturbed graph structure to reconstruct, even partially, the original graph. Estimating the likelihood of the existence of an edge between two vertices based on observed links is a well studied data mining task, called *link prediction* [32, 35]. In this section, we study the resilience of perturbation methods to such attacks. In particular, for the sake of simplicity, we focus on random sparsification; the case of random perturbation is discussed at the end of the section. To the best of our knowledge, this is the first study that formally analyzes the resilience of the randomization method to link-prediction attacks. A main feature of our analysis is in its generality, as it applies to any link-prediction method, independently of its specific operation.

We assume that the original graph $G = (V, E)$ is anonymized according to the sparsification model with edge removal probability of p. The resulting sparsified graph, $G_p = (U, E_p)$, is published. The adversary may apply on G_p a link-prediction model M_{LP}. The model predicts that some of the currently non-existing edges, $e \in (U \times U) \setminus E_p$, were removed during the randomized sparsification process; namely, the link-prediction model yields a new subset $E' \subseteq U \times U$, where $E' \supseteq E_p$. Hence, the attacker obtains a new graph $G' = (U, E')$, as a prediction of the original graph $G = (V = U, E)$.

In the sequence $G \mapsto G_p \mapsto G'$, the first graph, G, is the original graph, which remains unseen for the adversary; the adversary observes only the latter two graphs, G_p and G'. The set of vertices is the same in all three graphs, but, as before, we denote it by V in the context of the original graph and by U in the context of the observed graphs. The three graphs differ in their edge sets. While E_p is a subset of E (as a result of sparsification), E' (being the edge set of G') is a superset of E_p; specifically, $E' = E_p \cup E_{\mathrm{LP}}$, where E_{LP} is the set of edges that were predicted by the link-prediction model M_{LP}.

Our approach is general in the sense that it treats the model M_{LP} as a black box, for which the only assumptions made are regarding its predictive accuracy. Hence, our analysis is valid in general and any link-prediction algorithm from the literature could potentially be used.

In concrete, we make the following assumptions regarding the model M_{LP}:

(1) M_{LP} makes binary predictions: for each pair of vertices, $(y, z) \in (U \times U) \setminus E_p$, the algorithm makes a prediction $M_{\mathrm{LP}}(y, z) \in \{0, 1\}$ whether (y, z) is a missing edge or not. The pair (y, z) is included in the set E_{LP} if and only if $M_{\mathrm{LP}}(y, z) = 1$.

(2) We say that the prediction of M_{LP} for a pair $(y, z) \in (U \times U) \setminus E_p$ is *correct* if either (i) $M_{\mathrm{LP}}(y, z) = 1$ and $(y, z) \in E$, or (ii) $M_{\mathrm{LP}}(y, z) = 0$ and $(y, z) \notin E$.

(3) The quality of M_{LP} is quantified by two parameters — η and θ. The parameter η expresses the *false positive rate*, namely, the probability that M_{LP} makes a wrong prediction ($M_{\mathrm{LP}}(y, z) = 1$) for a non-edge of the graph G ($(y, z) \notin E$). Similarly, θ expresses the *false negative rate*, namely, the probability that M_{LP} makes a wrong prediction ($M_{\mathrm{LP}}(y, z) = 0$) for an actual edge of the graph G ($(y, z) \in E$).[4]

The table below summarizes all possible cases and their corresponding probabilities:

	$M_{\mathrm{LP}}(y, z) = 1$	$M_{\mathrm{LP}}(y, z) = 0$
$(y, z) \in E$	$1 - \theta$	θ
$(y, z) \notin E$	η	$1 - \eta$

9.1. Quantifying the level of obfuscation under the link-prediction attack

Consider a vertex v in the original graph G with degree $d(v) = a$, and assume that v is mapped to a vertex u in the sparsified graph G_p with degree $d_p(u) = b$. Subsequently, through the link-prediction process, the graph G_p is transformed to the graph G', where G' results from G_p by adding edges. Thus, the degree of the vertex u in G' becomes $d'(u) = c \geq b = d_p(u)$.

[4] It is worth noting that in the *statistical hypothesis testing* jargon, our η and θ are better known as α and β respectively. Moreover, $1 - \eta$ is also known as *specificity*, while $1 - \theta$ corresponds to *sensitivity* or *recall*.

The conditional probability

$$f(v;u) = \Pr(d_p(u) = b, d'(u) = c \mid d(v) = a, v \mapsto u) \tag{12}$$

is the probability that a vertex v in the original graph G that has degree $d(v) = a$ is mapped to a vertex u in the two observed graphs G_p and G' with degrees b and c, respectively. On the other hand, the inverse conditional probability

$$f'(v;u) = \Pr(d(v) = a \mid d_p(u) = b, d'(u) = c, v \mapsto u) \tag{13}$$

is the probability that a vertex u of degrees b and c in the two observed graphs, G_p and G', respectively, is the image of a vertex v in the original graph G that had degree $d(v) = a$. Following the paradigm that underlies our analysis in the previous sections, we proceed to derive formulas for computing those two probabilities.

We begin with $f'(v;u)$. By Bayes Theorem, we have

$$f'(v;u) = \Pr(d(v) = a \mid d_p(u) = b, d'(u) = c) = \frac{\Pr(d(v) = a, d_p(u) = b, d'(u) = c)}{\Pr(d_p(u) = b, d'(u) = c)}\,. \tag{14}$$

We show how to evaluate $f'(v;u)$ by providing derivations for the numerator and the denominator in Equation (14). For the numerator we have:

$$\begin{aligned}\Pr(d(v) = a, d_p(u) = b, d'(u) = c) = \\ \Pr(d'(u) = c \mid d_p(u) = b, d(v) = a) \cdot \Pr(d_p(u) = b \mid d(v) = a) \cdot \Pr(d(v) = a)\,.\end{aligned} \tag{15}$$

The three factors in Equation (15) may be estimated as follows.

- The probability $\Pr(d(v) = a)$ expresses the degree distribution in the original graph. The degree distribution is known to the data owner. We strengthen our security model by assuming that it is known also to the adversary. (For example, we may assume that the data owner publishes that distribution.)
- The conditional probability $\Pr(d_p(u) = b \mid d(v) = a)$ has already been discussed, and for the sparsification model it is given by Equation (6).
- To compute the conditional probability $\Pr(d'(u) = c \mid d_p(u) = b, d(v) = a)$, we use the accuracy parameters, η and θ, of the link-prediction model M_{LP}. M_{LP} provides predictions for each of the $N - 1 - b$ non-edges in the set $\{(u, z) : z \in U \setminus \{u\}, (u, z) \notin E_p\}$. Of those $N - 1 - b$ non-edges in G_p, $a - b$ were true edges in G that were removed during sparsification; the remaining $N - 1 - a$ non-edges were also non-edges in G. The link-prediction algorithm M_{LP} will predict that the degree of a vertex u in G' is $d'(u) = c$ if and only if there exist integers $s, t \geq 0$ such that $s + t = c - b$ and the algorithm predicts s edges correctly out of the $a - b$ true edges that were removed by sparsification, and t edges incorrectly out of the $N - 1 - a$ non-edges in G. Therefore,

$$\begin{aligned}\Pr(d'(u) = c \mid d(v) = a, d_p(u) = b) = \\ \sum_{s,t \geq 0 \mid s+t=c-b} \binom{a-b}{s} (1-\theta)^s \theta^{a-b-s} \binom{N-1-a}{t} \eta^t (1-\eta)^{N-1-a-t}.\end{aligned} \tag{16}$$

The denominator of Equation (14) is estimated in a similar way. We first apply the chain rule to obtain

$$\Pr(d_p(u) = b, d'(u) = c) = \Pr(d'(u) = c \mid d_p(u) = b) \cdot \Pr(d_p(u) = b)\,.$$

The probability $\Pr(d_p(u) = b)$ is then given by Equation (11). As for $\Pr(d'(u) = c \mid d_p(u) = b)$, we compute it by using the law of total probability and conditioning on the degree of the preimage v of u in G. Given that $d_p(u) = b$, the degree in G of the preimage v can be any value between b and $N - 1$. Therefore,

$$\begin{aligned}\Pr(d'(u) = c \mid d_p(u) = b) = \\ \sum_{a=b}^{N-1} \Pr(d'(u) = c \mid d_p(u) = b, d(v) = a) \cdot \Pr(d(v) = a \mid d_p(u) = b)\,.\end{aligned} \tag{17}$$

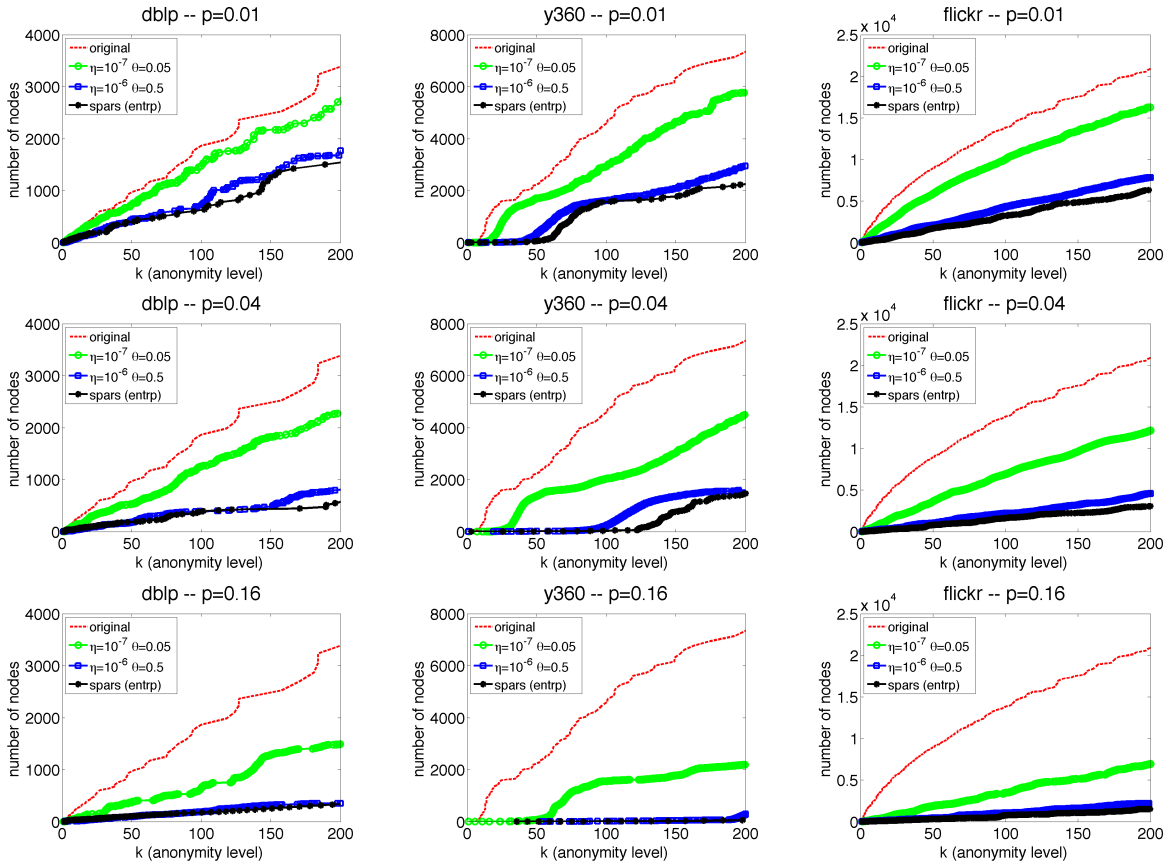

Figure 7: Resilience to link-prediction attacks: Anonymity levels of k-obfuscation on the three datasets for different values of p.

The first conditional probability on the right hand side of Equation (17) can be computed using Equation (16); the second conditional probability on the right hand side of Equation (17) can be computed, as discussed in Section 6, using Equations (10), (11), and (6).

The computation of $f(v;u)$, Equation (12), is similar to that of $f'(v;u)$, Equation (13). Using Bayes Theorem, we get that

$$f(v;u) = \Pr(d_p(u) = b, d'(u) = c \mid d(v) = a) = \frac{\Pr(d(v) = a, d_p(u) = b, d'(u) = c)}{\Pr(d(v) = a)}\,. \tag{18}$$

The numerator in Equation (18) is the same as in Equation (14). The denominator is $\Pr(d(v) = a)$, which, as explained earlier, can be extracted from the original graph G.

Next, we proceed along the same lines as in Section 5. In order to quantify the levels of degree obfuscation that are offered by the sparsified graph G_p, assuming the link-prediction model M_{LP}, we construct the $n \times n$ matrix F, where $F_{i,j} = f(v_i; u_j)$, $1 \leq i, j \leq n$. The (i,j)th entry in F is computed as described above, with $a = d(v_i)$, $b = d_p(u_j)$, and $c = d'(u_j)$. The matrix F enables us to compute the probability distributions $X_v(\cdot)$, for all $v \in V$, by normalizing its rows, see Equation (3). To quantify the levels of degree preimage obfuscation that are offered by the sparsified graph G_p, assuming the link-prediction model M_{LP}, we construct the matrix F', where $F'_{i,j} = f'(v_i; u_j)$, $1 \leq i, j \leq n$, and then derive the probability distributions $X_u(\cdot)$, for all $u \in U$, through Equation (4). Obfuscation by sparsification is immune to an attack that utilizes the link-prediction model M_{LP} if the levels of degree obfuscation and preimage obfuscation are not affected significantly by M_{LP}. This is empirically assessed next.

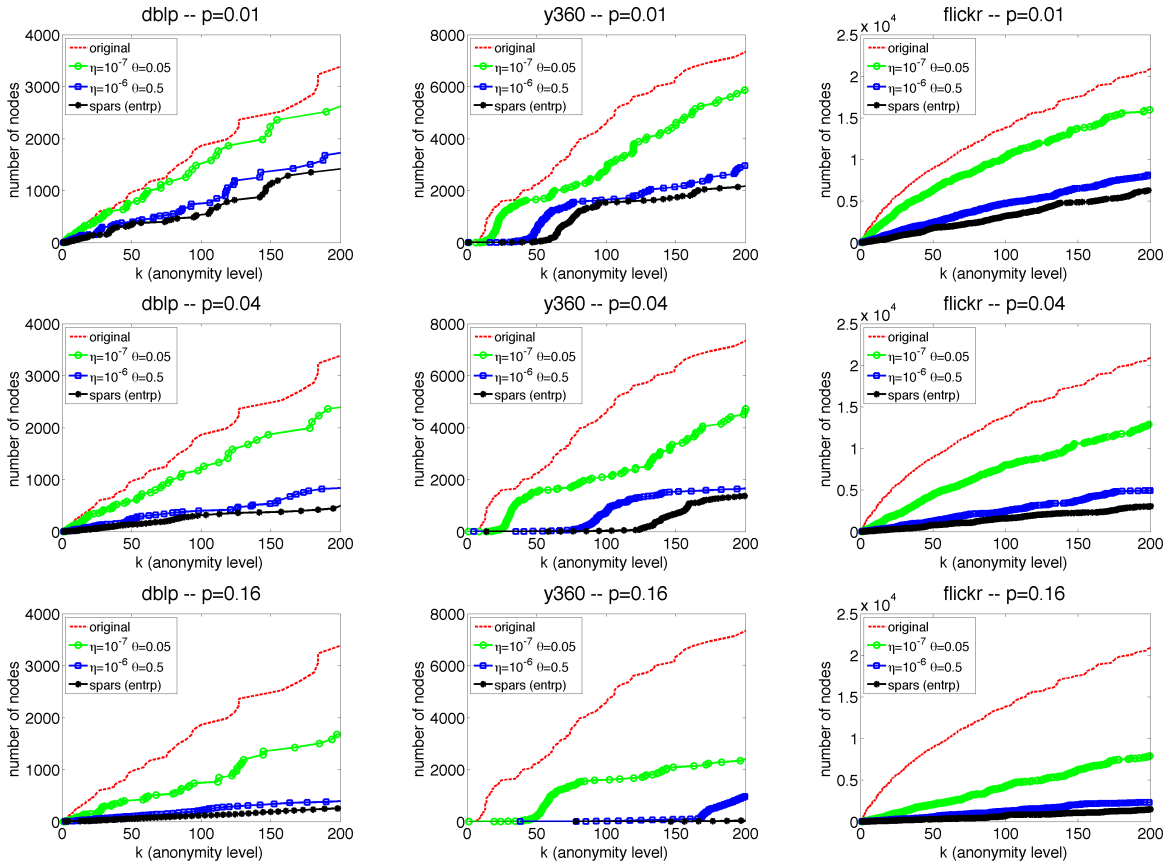

Figure 8: Resilience to link-prediction attacks: Anonymity levels of k-preimage obfuscation on the three datasets for different values of p.

9.2. Empirical assessment

We apply the procedure just described to quantify the obfuscation achieved by random sparsification (for various values of p) under a link-prediction attack, parameterized over the predictive effectiveness measures η and θ. In particular, we use two different settings for η and θ: $(\eta = 10^{-7}, \theta = 0.05)$ and $(\eta = 10^{-6}, \theta = 0.5)$.

In order to better understand the η and θ settings, we derive their corresponding interpretation in terms of *precision* and *recall*. In any binary prediction task, precision and recall are defined as

$$\text{precision} = \frac{TP}{TP + FP}, \quad \text{recall} = \frac{TP}{P},$$

where TP is the number of true positives, which in our setting are candidate edges in $(U \times U) \setminus E_p$ that are correctly classified as edges of E, FP is the number of False Positives, i.e., edges in $(U \times U) \setminus E_p$ that are incorrectly classified as edges of E, and P is the number of actual positives, i.e., $|E \setminus E_p|$. It is easy to see that the recall equals $1 - \theta$. To derive the precision, we note that $TP = (1 - \theta) \cdot |E \setminus E_p|$, and $FP = \eta \cdot |(V \times V) \setminus E|$.

Assuming a network with $5 \cdot 10^5$ vertices and $5 \cdot 10^6$ edges (similar to our `flickr` dataset), and assuming a sparsification parameter $p = 0.04$, the expected number of edges in the sparsified graph is $4.8 \cdot 10^6$. That gives a precision of $1/(1 + 2.5) \simeq 0.286$ for the setting $(\eta = 10^{-6}, \theta = 0.5)$, and a precision of $0.19/(0.19 + 0.25) \simeq 0.432$ for the setting $(\eta = 10^{-7}, \theta = 0.05)$. As reported in the literature [35], typical precision values for link prediction are in the range [0.1,0.3], and very rarely it is above 0.5. Those relatively small precision values are due to the fact that in the link-prediction problem, due to the quadratic number of possible edges, the

distribution of class labels is highly unbalanced. In the setting we just described, the link-prediction task would have to predict existence or non-existence for approximately $|U|^2 - |E_p| = 25 \cdot 10^{10} - 4.8 \cdot 10^6 \simeq 25 \cdot 10^{10}$ candidate edges, among which only $0.2 \cdot 10^6$ have been truly removed. The binary-prediction task is indeed very unbalanced, and thus hard; the fraction of positive instances is only $8 \cdot 10^{-7}$.

In Figures 7 and 8 we report the levels of anonymity maintained under the link-prediction attack for k-obfuscation and k-preimage obfuscation respectively, and for the three datasets that we used in the previous section. As in the plots shown there, for a given value of p we have on the x-axis the anonymity level and on the y-axis the number of vertices that *do not* reach such anonymity level. The plots report those values for the original graph G, for the sparsified graph G_p, and for the graph G' reconstructed by the adversary by means of link-prediction attack under the two settings: $(\eta = 10^{-7}, \theta = 0.05)$ and $(\eta = 10^{-6}, \theta = 0.5)$.

We observe that the attack with parameters $(\eta = 10^{-6}, \theta = 0.5)$ is rather ineffective as, in all the datasets and for all values of p, the level of anonymity does not change much from the sparsified graph. Under the much stronger (and less realistic) attack with parameters $(\eta = 10^{-7}, \theta = 0.05)$, which better reconstructs the original graph, the achieved anonymity level is, as expected, closer to that of the original graph. But, as we have already discussed, the first setting represents a very good, yet realistic, link-prediction model, while the second setting is too optimistic for the attacker.

As the levels of degree obfuscation and preimage obfuscation are not affected significantly by an attack based on reconstructing the original graph by means of a link-prediction method M_{LP}, we conclude that obfuscation by random sparsification is immune to such attacks.

9.3. The case of random perturbation

In principle, an adversary might implement a link-prediction attack even to a graph which has been anonymized by means of random perturbation, i.e., by both removing and adding edges. In such cases, the link-prediction method M_{LP} would need not only to guess which edges have been removed among the ones that do not exist in G_p, but also to guess which edges are artificial among the ones that do exist in G_p. Obviously, this case is more complex for the attacker than the case of random sparsification. We avoid replicating the theoretical analysis and the empirical assessment for such case. However, we can safely claim that it is unlikely that such an attack would succeed, given that we already showed that link-prediction fails in threatening anonymity under the easier case of random sparsification.

10. The distributed setting

Assume that the network data is split between s players. Each player controls a subset of the vertices and knows only the edges that are adjacent to one of the vertices under his control. Specifically, if V^i is the set of vertices under the control of the ith player, $1 \leq i \leq s$, then the overall set of vertices is $V = \bigcup_{i=1}^{s} V^i$; player i knows only edges of the form (v, v') where either v or v', or both, belong to V^i. The goal is to design a protocol that allows the players to modify the set of edges in the unified graph so that the graph becomes sanitized in some sense, without revealing to other players sensitive information like the identity of vertices or existence of edges between vertices.

To illustrate such a setting, assume that the players are banks, the vertices are accounts in those banks, and the edges denote financial transactions between the accounts. In such a setting, each vertex is identified by an account number, but the bank is trusted to protect the identity of the clients that hold those accounts. A bank that transfers money from one of his clients to a client of another bank, knows that those two vertices are connected by an edge, but it does not know the identity of the client of the other bank. As another example, assume that the network is a correspondence network between email addresses. Here it is natural to assume that the identity of the vertices is not confidential, since typical email addresses disclose the name of the individual that holds them. In this case, it is needed only to protect from each player the existence of edges between vertices that are controlled by other players. Namely, even though the identity of the individuals behind the email addresses `MeTarzan@Site1.com` and `YouJane@Site2.com`, that are controlled by players 1 and 2 respectively, is expected to be known to player 3, players 1 and 2 are trusted to withhold from player 3 the fact that those two individuals exchanged emails.

Algorithm 1 Secure computation of the sum

Input: Each player i has an input vector $\mathbf{a}_i \in \mathbb{Z}_r^n$, $1 \leq i \leq s$.
Output: $\mathbf{a} := \sum_{i=1}^{s} \mathbf{a}_i \bmod r$.

1: Each player i selects s random share vectors $\mathbf{a}_{i,j} \in \mathbb{Z}_r^n$, $1 \leq j \leq s$, such that $\sum_{j=1}^{s} \mathbf{a}_{i,j} = \mathbf{a}_i \bmod r$.
2: Each player i sends $\mathbf{a}_{i,j}$ to player j for all $1 \leq j \neq i \leq s$.
3: Each player j computes $\mathbf{b}_j = \sum_{i=1}^{s} \mathbf{a}_{i,j} \bmod r$.
4: Each player j, $2 \leq j \leq s$, sends $\mathbf{b}_j$ to player 1.
5: Player 1 computes $\mathbf{a} = \sum_{j=1}^{s} \mathbf{b}_j \bmod r$ and broadcasts it.

In the context of our work, the goal is to arrive at a k-obfuscation of the combined social network. In Section 10.1 we describe two basic problems of secure multi-party computation on which we rely later on: secure summation of private integers and secure multiplication of private bits. In Section 10.2 we explain how the players can jointly compute an obfuscation of the unified network by either perturbation or sparsification, using some probability parameter p. In Section 10.3 we describe a distributed protocol that allows the players to verify whether the obfuscated network that they jointly computed satisfies k-degree obfuscation. Finally, in Section 10.4 we comment on the implementation of other methods for identity obfuscation in networks in settings where the network data is distributed.

10.1. Two basic problems of secure multi-party computation

Computing the sum of private integers has well known simple secure multi-party protocols [6, 26]. In that problem, each player holds a private nonnegative integer and the players wish to compute the sum of those integers without disclosing to any of the players any information on the input integers of other players, beyond what is implied by his own input and the final output. For the sake of completeness we include here a version of the protocol that was presented by Benaloh [6] for adding private vectors — Algorithm 1. In that algorithm, r is any sufficiently large integer that bounds all components of the sum vector.

Theorem 1. *Algorithm 1 computes the required sum* $\mathbf{a} = \sum_{i=1}^{s} \mathbf{a}_i$. *It offers perfect privacy with respect to semi-honest players in the following sense: any coalition of players cannot learn from their view of the protocol about the input vectors of other players more information than what is implied by their own input vectors and the final output.*

The proof of Theorem 1 is standard and hence omitted.

Another problem of secure multi-party computation is that of computing the product of private bits; i.e., computing $b = \prod_{i=1}^{s} b_i$, where b_i is a private bit held by player i, $1 \leq i \leq s$. Due to its simplicity, it may be solved in a practical manner by one of the known generic solutions [4, 5, 16]. A much simpler protocol that employs less cryptographic primitives was presented in [43] for the case $s > 2$. The protocol, which we include here as Algorithm 2, is based on the fact that the product of all bits equals 1 if and only if their sum equals s. Hence, the protocol basically computes the sum of the private bits modulo r, where r is any integer greater than or equal to $s + 1$. But instead of completing the summation protocol, it stops at the point in which players 1 and s have two additive shares of the sum; i.e., player 1 has a random share a and player s has a share a_s and $a + a_s = \sum_{i=1}^{s} b_i \bmod r$ (Lines 1-5). It is now needed to verify whether $a + a_s = s \bmod r$, or, alternatively, whether $a_s = a'$ where $a' = s - a \bmod r$ (Line 6). This equality verification is carried out by means of random shifting and hashing those two values and sending the hashed values to player 2 so that he may perform the equality verification (Lines 7-10).

10.2. Distributed randomization

Given a distributed graph $G = (V, E)$ as described above, and a probability parameter $p \in [0, 1]$, we describe how the players may execute randomization by sparsification and randomization by perturbation.

Randomization by sparsification may be carried out easily. Consider the edge $e = (u, v)$ and assume that $u \in V^i$ and $v \in V^j$. Assume, without loss of generality, that $i \leq j$. Then player i tosses the coin for that edge and he informs player j in case the edge was removed.

Algorithm 2 Secure computation of the product of private bits

Input: Each player i has an input bit $b_i \in \mathbb{Z}_2$, $1 \leq i \leq s$.
Output: $b = \prod_{i=1}^{s} b_i$.

1: Each player i selects s random shares $b_{i,j} \in \mathbb{Z}_r$, $1 \leq j \leq s$, such that $\sum_{j=1}^{s} b_{i,j} = b_i \bmod r$.
2: Each player i sends $b_{i,j}$ to player j for all $1 \leq j \neq i \leq s$.
3: Each player j computes $a_j = \sum_{i=1}^{s} b_{i,j} \bmod r$.
4: Each player j, $2 \leq j \leq s-1$, sends a_j to player 1.
5: Player 1 computes $a = \left(\sum_{j=1}^{s-1} a_j\right) \bmod r$.
6: Player 1 computes $a' = (s - a) \bmod r$.
7: Players 1 and s agree on a large random number k and on a secure hash function h.
8: Player 1 sends to player 2 the value $u := h(k + a')$.
9: Player s sends to player 2 the value $v := h(k + a_s)$.
10: Player 2 outputs 1 if $u = v$ and 0 otherwise.

Randomization by perturbation can be carried out similarly. First, the players need to compute the total number of edges in the graph, m, in order to derive the probability $q(p)$ for turning a non-edge into an edge, see Equation (1). The computation of n and m (the total number of vertices and edges, respectively, in G) can be carried out by invoking Algorithm 1. When invoking that algorithm in order to compute m, the input of the ith player is the number of edges that connect a vertex in V^i to a vertex in V^j for $j \geq i$ (so that each edge is counted exactly once). After computing $q(p)$, the players decide about the removal of an existing edge in the same way as described above, and about the addition of a non-existing edge similarly, using the probability q. Here too, the coin toss is made by the player with the smaller index; specifically, if $e = (u, v)$ is a non-edge, and $u \in V^i$ and $v \in V^j$, where $i \leq j$, it is player i who tosses the q-biased coin in order to decide whether e should be added or not, and then he informs player j in case e was selected to be added.

The operation of the distributed protocol, for any partition of the network among any number of players, coincides with that of the non-distributed algorithm. Namely, whether the whole network is held by one player or distributed among s players, for any $s > 1$, the resulting obfuscation will be the same, in the probabilistic sense, as formalized in the next theorem.

Theorem 2. *Let:*

- $G = (V, E)$ *be a graph;*
- $s \geq 1$ *be an integer;*
- Π *be a partition of the vertices in* V *among* s *players;*
- $\Gamma(V)$ *be the family of all graphs over the set of vertices* V*.*
- $\phi_G : \Gamma(V) \to [0, 1]$ *be a probability distribution over* $\Gamma(V)$ *that specifies for each* $G' \in \Gamma(V)$ *the probability that the distributed protocol will obfuscate* G *into* G'*.*

Then ϕ_G *is independent of* s *and* Π*.*

The proof of Theorem 2 is straightforward. Indeed, the distributed protocol perfectly simulates the non-distributed algorithm since it does not matter whether the coin tosses are made by a single player or by many players. Hence, all experiments in Section 8 are relevant also for the distributed setting.

10.3. Distributed verification of k-degree obfuscation

Once the players obtained the randomized graph G_p, they need to compute the level of degree obfuscation that it offers, along the lines of Section 6. To that end, each of the players computes the following vector:

$$\mathbf{y}^i = (y_0^i, \ldots, y_{n-1}^i), \quad 1 \leq i \leq s,$$

where y_j^i is the number of vertices in V^i that have degree j in G_p. Then, by invoking Algorithm 1, they may add those vectors securely and get the vector

$$\mathbf{y} = (y_0, \ldots, y_{n-1}),$$

where y_j is the number of vertices in G_p that have degree j, $0 \leq j \leq n-1$ (the maximal degree is $n-1$ since $|V| = n$).

Given the degree sequence $\mathbf{y}$, each player can construct the rows of the matrix $F_{i,j} = f(v_i; u_j)$, $1 \leq i, j \leq n$, that correspond to each of the vertices under his control, using Equations (5)+(6) or (5)+(7). Then, the probability distribution that corresponds to each of his vertices is given by Equation (3). Consequently, player i may compute the minimal entropy among the probability distributions $X_v(\cdot)$ where $v \in V^i$ and see whether it satisfies the required level of obfuscation.

Let b_i be a bit that equals 1 if k-obfuscation was achieved for all vertices under the control of player i, $1 \leq i \leq s$, and 0 otherwise. If $b = \prod_{i=1}^{s} b_i = 1$, then the entire graph G_p satisfies k-obfuscation and may therefore be released. If, however, $b = 0$, then the players need to increase p and repeat this procedure.

The bits b_i expose some information on the vertices in V^i. For example, if for a given value of p all players $i \neq j$ got $b_i = 1$ but player j got $b_j = 0$, it reveals the fact that V^j has a vertex with degree which is an outlier in the general degree distribution. To avoid such information leakage, the players may invoke a secure multi-party protocol for computing b, as discussed in Section 10.1.

10.4. Concluding comments

As mentioned in the introduction, distributed privacy-preserving social-network analysis has not been well reported in the literature so far [52]. To the best of our knowledge, the only study that considers anonymization of distributed networks is [42]. That study proposes an algorithm for network anonymization by means of clustering vertices, and then devises a privacy-preserving distributed version of that algorithm for settings in which the network data is split among several data owners.

So far, there has been no attempt to transform the algorithms of [10, 33, 51, 61, 63], that achieve k-anonymity with respect to some property of the vertices, into distributed protocols. Those algorithms perform local changes in the graph based on some global information. For example, in order to achieve k-neighborhood anonymity [61] for some specific vertex, it is necessary first to know how many vertices in the graph share the same type of neighborhood. The need for global information would require the players to engage in a secure multi-party protocol in order to compute the required information without disclosing sensitive information.

The great advantage of the method of randomization is that it may be implemented locally with minimal computation of global information, as shown in Sections 10.2 and 10.3. Due to the simplicity of the randomization method, the corresponding distributed protocols require no cryptographic primitives, such as homomorphic encryption, secret sharing or oblivious transfer, as many distributed protocols for achieving anonymity require (e.g. [23, 25, 60]), but only basic protocols as described in Section 10.1.

11. Related work

We have already discussed a number of papers that are most related to our work. In this section we summarize our discussion and provide a wider, yet brief, overview of the research in the area of anonymizing social graphs. We do not aim at providing a comprehensive survey of the research area as this is beyond the scope of the current paper; interested readers may consult the various available surveys [19, 52, 62] or tutorials [18].

Probably the first work to highlight the privacy risks associated to publishing naïvely anonymized social graphs is the work of Backstrom et al. [2]. In their paper, Backstrom et al. assume that a social graph is published, with all user identifiers removed. They then show that it is possible for an adversary to learn whether certain edges of the original graph exist. In particular, they propose two kinds of attacks: *active attacks* and *passive attacks*. In attacks of the first kind, the adversary actively creates a fake subgraph H and attaches it to the set of target vertices for which she wishes to find out whether they have edges among

them. After the anonymized graph is released, the adversary can find the subgraph H in the released graph G, and follow edges from H to locate the target vertices in G, and determine all edges among them. Attacks of the second kind are based on the observation that most vertices in social networks usually belong to a small uniquely identifiable subgraph. Thus, an adversary can collude with other vertices in the network to identify additional vertices connected to the distinct subset of the coalition. The experiments on a real social network with 4.4 million vertices and 77 million edges show that the creation of 7 vertices by an attacker can reveal on average 70 target vertices and compromise the privacy of approximately 2 400 edges between them. Following the work of Backstrom et al., there has been a growing effort in trying to provide various forms of k-anonymity for graphs.

Another work presenting negative results is by Narayanan and Shmatikov [37], who assume that the anonymized social graph is released along with vertex-attribute information. Moreover, they assume that a second non-anonymized social network, on the same set of vertices, is available. They then develop a de-anonymization algorithm. They experimentally show that one third of the users who can be verified to have accounts on both Twitter and Flickr can be re-identified in the "anonymous" Twitter graph with only 12% error rate.

Relational data anonymity. The traditional k-anonymity framework [39] was developed for relational tables in a quite simple setting. The basic assumptions are that the table to be anonymized contains entity-specific information, that each tuple in the table corresponds uniquely to an individual, and that attributes are divided into *quasi-identifiers* (i.e., attributes that appear in publicly available datasets, whose combination may yield a unique identification by means of linking attacks) and *sensitive attributes* (attributes that are publicly unknown and need to be protected). The data is transformed in such a way that, for every combination of values of the quasi-identifier in the sanitized data, there are at least k records that share those values.

Some limitations of the k-anonymity model, with consequent proposals for improvement, have emerged in the literature. One first drawback is that the difference between quasi-identifiers and sensitive attributes may be sometimes vague, leading to a large number of quasi-identifiers. This problem has been studied by Aggarwal et al. [1], who analyzes the scalability of distortion with respect to the number of dimensions used in the k-anonymization process, showing that for sparse data the usability of the anonymized data could sensibly decrease. Xiao and Tao [53] propose a decomposition of quasi-identifiers and sensitive data into independently shared databases. Kifer and Gehrke [29] suggest to share anonymized marginals (statistical models such as density estimates of the original table) instead of the private table. Another drawback of simple k-anonymity is that it may not protect sensitive values when their entropy (diversity) is low: this is the case in which a sensitive value is too frequent in a set of tuples with the same quasi-identifier values after k-anonymization [31, 36, 48]. Regardless of these limitations, k-anonymity remains a widely accepted model both in scientific literature and in privacy related legislation,[5] and in recent years a large research effort has been devoted to develop algorithms for k-anonymity [15, 28, 30, 38, 44, 49]. Unfortunately, these methods can not be borrowed for social networks directly, due to the complex nature of graph data.

Network data anonymity. The methods for anonymity preservation in social networks can be broadly classified into three categories:

(1) Methods based on deterministic alteration of the graph by edge additions or deletions [10, 33, 51, 61, 63];
(2) Methods based on randomized alteration of the graph by addition, deletion or switching of edges [17, 22, 34, 50, 55, 56, 57];
(3) Methods based on generalization by means of clustering of vertices [9, 12, 13, 20, 21, 42, 58].

We have already discussed the work of Liu and Terzi [33], which is based on the degree property, the work of Zhou and Pei [61], where the adversary knows the distance-one neighborhood of the target vertex,

[5] A thorough discussion on the relevance of syntactic anonymity and a comparison to differential privacy can be found in the article of Clifton et al. [11].

and the works by Zou et al. [63] and Wu et al. [51] that anonymize a network by edge and vertex addition so that the resulting graph is k-anonymized with respect to vertex automorphism.

In another line of research in that category of works, Cheng et al. [10] identify two main types of sensitive information that a user may want to keep private and which may be under attack in a social network, namely *NodeInfo*, information associated to nodes (vertices), and *LinkInfo*, information associated to links (edges). Cheng et al. show that k-isomorphism, meaning anonymization by forming k pairwise isomorphic subgraphs, is both sufficient and necessary for protecting those two types of sensitive information.

In the second category of papers, regarding methods based on randomized alteration, we have already discussed the work of Hay et al. [22], which was one of the leading papers. Subsequently, Ying and Wu [55] introduced a randomization strategy that can preserve the spectral properties of the graph. In particular, they proposed two spectrum preserving randomization methods, i.e., randomly adding and deleting edges or randomly switching edges, which maintain the graph spectral characteristics, i.e., the largest eigenvalue of the adjacency matrix and the second smallest eigenvalue of the Laplacian matrix.

Liu et al. [34] analyze the case in which the social network edges are weighted. An edge weight can reflect the strength of the social tie, or the frequency of communication along that edge, which may be considered a sensitive information. In this context, Liu et al. study the problem of anonymizing a social graph where edge weights are considered sensitive. They consider two kinds of data characteristics to be preserved for data utility sake: how to keep the lengths of shortest paths between vertices within an error bound, and how to maintain the exact shortest paths between vertices in a selected subset.

Ying and Wu [56], study the problem of how to generate a synthetic graph that matches given features of a real social network, in addition to a given degree sequence. They propose a Markov Chain-based feature-preserving randomization.

Wu et al. [50] start from the claim that the topological features of a graph are significantly lost when a medium or high randomization is applied. Therefore, they focus on reconstructing a graph from the edge randomized graph so that accurate feature values can be recovered. To that end, they develop a method that is based on low-rank approximation. They show, on two modestly sized datasets, that accurate feature values can still be recovered from the randomized graphs even with large perturbation. Nevertheless, they also find that such reconstructions do not pose a meaningful threat for individual privacy (in line with our findings in Section 9 about resilience to link-prediction attacks).

In the third category of papers, regarding methods based on generalization, Hay et al. [20, 21] propose vertex clustering in order to achieve anonymity. The proposed method generalizes an input network by clustering vertices and publishing the number of vertices in each partition along with the densities of edges within and across partitions. Data analysts can still use the anonymized graphs to study macro-properties of the original graph. In Section 2.1, we already discussed the kind of adversarial knowledge that Hay et al. [20, 21] consider. Extending this approach, Campan and Truta [9] study the case in which vertices have additional attributes (e.g., demographic information). They propose clustering vertices and revealing only the number of intra- and inter-cluster edges. The vertex properties are generalized in such a way that all vertices in the same cluster have the same generalized representation. Tassa and Cohen [42] consider a similar setting and propose a sequential clustering algorithm that issues anonymized graphs with higher utility than those issued by the algorithm of Campan and Truta. In addition, they devise a privacy-preserving distributed version of their algorithm for settings in which the network data is split between several data owners.

Cormode et al. [13] consider a privacy issue in bipartite-graphs, where we have two sets of entities, the links among them, and also a set of attributes for each entity. For example, the two sets of entities might be patients and drugs, and the connection might represent which patient takes which drugs. The authors assume that the adversary will use knowledge of attributes rather than graph structure in a matching attack. As a result, it is not considered a privacy risk to publish the exact graph structure. To prevent matching attacks based on attributes, their technique masks the mapping between vertices in the graph and real-world entities (with their associated attribute vectors). This is done by clustering the vertices, and the corresponding entities, into groups. The framework is generalized to arbitrary interaction graphs with attributes on vertices and edges in the follow-up work [12].

Zheleva and Getoor [58] consider the case where there are multiple types of edges, one of which is assumed

to be sensitive and should be protected. It is assumed that the network is published without the sensitive edges and an adversary predicts sensitive edges based on the other observed non-sensitive edges. Various anonymization techniques are proposed, based on removing some types of edges and grouping vertices so that only the number of edges between groups is revealed. Similarly to our paper, Zheleva and Getoor bring the idea of using link prediction as a tool to compromise anonymization. However, their paper considers a very different model than ours.

Recently, Boldi et al. [7] suggested a new approach for obfuscation graph data by means of randomization. Their approach is based on injecting uncertainty to the edges of the social graph and publishing the resulting uncertain graph. As opposed to the methods discussed in this paper, which obfuscate graph data by adding or removing edges entirely, the method of Boldi et al. [7] uses finer-grained perturbations that add or remove edges partially. It is shown there that by such finer-grained perturbations it is possible to achieve comparable levels of obfuscation with smaller changes in the data.

12. Conclusion

Randomization techniques are very appealing as privacy protection mechanisms for various reasons. They are not geared towards protecting the graph against any particular adversarial attack. They are very simple and easy to implement. In addition, as their implementation does not require a global view of the entire graph, they can be easily implemented in distributed settings.

Unfortunately, recent studies have concluded that random perturbation can not achieve meaningful levels of anonymity without deteriorating the graph features. Those studies quantified the level of anonymity that is obtained by random perturbation by means of a-posteriori belief probabilities.

In this paper we offer a novel information-theoretic perspective on this issue, concluding that randomization techniques for identity obfuscation are back in the game, as they may achieve meaningful levels of obfuscation while still preserving characteristics of the original graph. We prove our claim by means of a principled theoretical analysis and a thorough experimental assessment.

We introduce an essential distinction between two different kinds of identity obfuscation, corresponding to an adversary wishing to re-identify a particular individual, or any individual. We propose to quantify the anonymity level that is provided by the perturbed network by means of entropy, and we explain why the entropy-based quantification is more adequate than the previously used quantification based on a-posteriori belief. Moreover, we prove that the obfuscation level quantified by means of the entropy is always no less than the one based on a-posteriori belief probabilities. We derive formulas to compute those entropies in the case where the background knowledge of the adversary is the degree of the target individual.

We conduct thorough experimentation on three very large datasets and measure multitude of features of the perturbed graph, showing that randomization techniques achieve meaningful levels of obfuscation while preserving most of the features of the original graph. We also study resilience of random perturbation methods to graph reconstruction attacks based on link prediction. Given a general link-prediction method, with a predefined prediction accuracy, we show how to quantify the level of anonymity guaranteed by the obfuscation. Then we empirically prove that even for very accurate link-prediction methods the level of anonymity guaranteed remains very close to the one before the attack.

As pointed out in other papers, the availability of additional information (for example interest groups) together with the published graph might create room for more informed attacks, based on link prediction [59].

Our analysis of resilience of random perturbation to link-prediction attacks does not assume any particular background knowledge; instead, it simply assumes a predefined prediction effectiveness of the link-prediction attack. In this perspective, we can conclude that randomization methods are resilient even against this form of more informed attacks.

In some settings, the network data is split between several data holders, or players. For example, the data in a network of email accounts, where two vertices are connected if they exchanged a minimal number of email messages, might be split between several email service providers. As another example, consider a transaction network where an edge denotes a financial transaction between two individuals; such a network

would be split between several banks. In such settings, each player controls some of the vertices (clients) and he knows only the edges that are adjacent to the vertices under his control. It is needed to devise distributed protocols that would allow the players to arrive at a sanitized version of the unified network, without revealing to them sensitive information on clients of other players.

The recent survey by Wu et al. [52] on privacy-preservation of graphs and social networks concludes by recommendations for future research in this emerging area. One of the proposed directions is distributed privacy-preserving social network analysis, which "has not been well reported in literature". The randomization method is a natural candidate method to be used for achieving anonymization in a distributed setting. It is far more suited to the distributed setting than the methods of the first and third categories that were discussed in the introduction since it hardly requires a global view of the entire graph.

An interesting direction for further research is the anonymization of social networks in which the links are of different types, as discussed by Zheleva and Getoor [58]. In such cases, it is possible that different clients will require to see different parts of the network that consist of link information which is relevant to them. For example, one client may be interested in seeing only links of collaboration, while another client may be interested in seeing only links of correspondence. Hence, such networks might be subjected to several partial data releases. (This is similar to the case of sequential release of tabular data, where each release of a given table contains only part of the table attributes. The anonymization of such sequential releases is an intricate task [40, 41, 47].) Even if each release is obfuscated in order to comply with k-obfuscation, it is possible that an adversary who examines several of those releases might extract inferences that are no longer consistent with k-obfuscation. It is therefore needed to devise methods to obfuscate each release so that even the combination of all of those releases maintains the desired privacy requirement.

Acknowledgements. The first two authors were partially supported by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037, "Social Media" (`www.cenitsocialmedia.es`). Part of this research was conducted when the third author was a guest of Yahoo! Research, Barcelona.

References

[1] C. C. Aggarwal. On k-anonymity and the curse of dimensionality. In *VLDB*, pages 901–909, 2005.
[2] L. Backstrom, C. Dwork, and J. M. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW*, pages 181–190, 2007.
[3] M. Barbaro and T. Zeller. A face is exposed for AOL searcher no. 4417749. *New York Times*, 2006.
[4] D. Beaver, S. Micali, and P. Rogaway. The round complexity of secure protocols. In *STOC*, pages 503–513, 1990.
[5] A. Ben-David, N. Nisan, and B. Pinkas. FairplayMP - A system for secure multi-party computation. In *CCS*, pages 257–266, 2008.
[6] J. Benaloh. Secret sharing homomorphisms: Keeping shares of a secret secret. In *Crypto*, pages 251–260, 1986.
[7] P. Boldi, F. Bonchi, A. Gionis, and T. Tassa. Injecting Uncertainty in Graphs for Identity Obfuscation. *PVLDB*, 5(11):1376–1387, 2012.
[8] F. Bonchi, A. Gionis, and T. Tassa. Identity obfuscation in graphs through the information theoretic lens. In *ICDE*, pages 924–935, 2011.
[9] A. Campan and T. Truta. A clustering approach for data and structural anonymity in social networks. In *PinKDD*, 2008.
[10] J. Cheng, A. W.-C. Fu, and J. Liu. K-isomorphism: privacy preserving network publication against structural attacks. In *SIGMOD*, pages 459–470, 2010.
[11] C. Clifton and T. Tassa. On syntactic anonymity and differential privacy. *Transactions on Data Privacy*, 6(2):161–183, 2013.
[12] G. Cormode, D. Srivastava, S. Bhagat, and B. Krishnamurthy. Class-based graph anonymization for social network data. *PVLDB*, 2(1):766–777, 2009.
[13] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. *PVLDB*, 1(1):833–844, 2008.
[14] A. Gionis, A. Mazza, and T. Tassa. k-Anonymization revisited. In *ICDE*, pages 744–753, 2008.
[15] J. Goldberger and T. Tassa. Efficient anonymizations with enhanced utility. *Transactions on Data Privacy*, 3(2):149–175, 2010.
[16] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.
[17] S. Hanhijärvi, G. Garriga, and K. Puolamaki. Randomization techniques for graphs. In *SDM*, pages 780–791, 2009.
[18] M. Hay, K. Liu, G. Miklau, J. Pei, and E. Terzi. Privacy-aware data management in information networks. In *SIGMOD*, pages 1201–1204, 2011.

[19] M. Hay, G. Miklau, and D. Jensen. Analyzing private network data. In F. Bonchi and E. Ferrari, editors, *Privacy-Aware Knowledge Discovery: Novel Applications and New Techniques*, pages 459 – 498. Chapman & Hall/CRC, 2010.
[20] M. Hay, G. Miklau, D. Jensen, D. F. Towsley, and C. Li. Resisting structural re-identification in anonymized social networks. *VLDB J.*, 19(6):797–823, 2010.
[21] M. Hay, G. Miklau, D. Jensen, D. F. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *PVLDB*, 1(1):102–114, 2008.
[22] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava. Anonymizing social networks. *University of Massachusetts Technical Report*, 07(19), 2007.
[23] W. Jiang and C. Clifton. A secure distributed framework for achieving k-anonymity. *The VLDB Journal*, 15(4):316–333, 2006.
[24] R. Jones, R. Kumar, B. Pang, and A. Tomkins. "I know what you did last summer": Query logs and user privacy. In *CIKM*, pages 909–914, 2007.
[25] P. Jurczyk and L. Xiong. Distributed anonymizations: Achieving privacy for both data subjects and data providers. In *DBSec*, pages 191–207, 2009.
[26] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *DMKD*, 2002.
[27] G. Karypis and V. Kumar. Analysis of multilevel graph partitioning. In *SC*, page 29, 1995.
[28] B. Kenig and T. Tassa. A practical approximation algorithm for optimal k-anonymity. In *DMKD*, 25(1):134-168, 2012
[29] D. Kifer and J. Gehrke. Injecting utility into anonymized datasets. In *SIGMOD*, pages 217–228, 2006.
[30] M. Last, T. Tassa, A. Zhmudyak, and E. Shmueli. Improving accuracy of classification models induced from anonymized datasets. *Information Sciences*, 256:138–161, 2014.
[31] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *ICDE*, pages 106–115, 2007.
[32] D. Liben-Nowell and J. M. Kleinberg. The link prediction problem for social networks. In *CIKM*, pages 556–559, 2003.
[33] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD*, pages 93–106, 2008.
[34] L. Liu, J. Wang, J. Liu, and J. Zhang. Privacy preservation in social networks with sensitive edge weights. In *SDM*, pages 954–965, 2009.
[35] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150 – 1170, 2011.
[36] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. ℓ-Diversity: privacy beyond k-anonymity. In *ICDE*, page 24, 2006.
[37] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, pages 173–187, 2009.
[38] H. Park and K. Shim. Approximate algorithms with generalizing attribute values for k-anonymity. In *Inf. Syst.*, 35(8):933–955, 2010.
[39] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *PODS*, page 188, 1998.
[40] E. Shmueli and T. Tassa. Privacy by diversity in sequential releases of databases. *Forthcoming*.
[41] E. Shmueli, T. Tassa, R. Wasserstein, B. Shapira, and L. Rokach. Limiting disclosure of sensitive data in sequential releases of databases. *Information Sciences*, 191:98–127, 2012.
[42] T. Tassa and D. Cohen. Anonymization of centralized and distributed social networks by sequential clustering. In *IEEE Trans. Knowl. Data Eng.*, 25(2):311–324, 2013.
[43] T. Tassa and E. Gudes. Secure distributed computation of anonymized views of shared databases. In *ACM Trans. Database Syst.*, 37(2):11, 2012.
[44] T. Tassa, A. Mazza, and A. Gionis. k-Concealment: An alternative model of k-type anonymity. In *Transactions on Data Privacy*, 5(1):189–222, 2012.
[45] B. Thompson and D. Yao. The union-split algorithm and cluster-based anonymization of social networks. In *ASIACCS*, pages 218–227, 2009.
[46] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos. Epidemic spreading in real networks: An eigenvalue viewpoint. In *SRDS*, pages 25–34, 2003.
[47] K. Wang and B. Fung, Anonymizing sequential release. In *KDD*, pages 414–423, 2006.
[48] R. Wong, J. Li, A. Fu, and K. Wang. (α, k)-anonymity: An enhanced k-anonymity model for privacy preserving data publishing. In *KDD*, pages 754–759, 2006.
[49] W.K. Wong, N. Mamoulis, and D.W.L. Cheung. Non-homogeneous generalization in privacy preserving data publishing. In *SIGMOD*, pages 747–758, 2010.
[50] L. Wu, X. Ying, and X. Wu. Reconstruction from randomized graph via low rank approximation. In *SDM*, pages 60–71, 2010.
[51] W. Wu, Y. Xiao, W. Wang, Z. He, and Z. Wang. k-Symmetry model for identity anonymization in social networks. In *EDBT*, pages 111–122, 2010.
[52] X. Wu, X. Ying, K. Liu, and L. Chen. A survey of privacy-preservation of graphs and social networks. In C. C. Aggarwal and H. Wang, editors, *Managing and Mining Graph Data*, pages 421–453. Springer, 2010.
[53] X. Xiao and Y. Tao. Anatomy: Simple and Effective Privacy Preservation. In *VLDB*, pages 139–150, 2006.
[54] X. Ying, K. Pan, X. Wu, and L. Guo. Comparisons of randomization and k-degree anonymization schemes for privacy preserving social network publishing. In *SNA-KDD*, pages 1–10, 2009.
[55] X. Ying and X. Wu. Randomizing social networks: A spectrum preserving approach. In *SDM*, pages 739–750, 2008.

[56] X. Ying and X. Wu. Graph generation with prescribed feature constraints. In *SDM*, pages 966–977, 2009.
[57] X. Ying and X. Wu. On link privacy in randomizing social networks. In *PAKDD*, pages 28–39, 2009.
[58] E. Zheleva and L. Getoor. Preserving the privacy of sensitive relationship in graph data. In *PinKDD*, pages 153–171, 2007.
[59] 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*, pages 531–540, 2009.
[60] S. Zhong, Z. Yang, and R. Wright. Privacy-enhancing k-anonymization of customer data. In *PODS*, pages 139–147, 2005.
[61] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *ICDE*, pages 506–515, 2008.
[62] B. Zhou, J. Pei, and W. Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *SIGKDD Explorations*, 10(2):12–22, 2008.
[63] L. Zou, L. Chen, and M. T. Özsu. K-automorphism: A general framework for privacy preserving network publication. *PVLDB*, 2(1):946–957, 2009.