



Graph Perturbation as Noise Graph Addition: A New Perspective for Graph Anonymization

Vicenç Torra^{1,2} and Julián Salas³(✉)

¹ Hamilton Institute, Maynooth University, Maynooth, Ireland
vtorra@ieee.org

² University of Skövde, Skövde, Sweden

³ CYBERCAT-Center for Cybersecurity Research of Catalonia, Internet Interdisciplinary Institute (IN3), Universitat Oberta de Catalunya, Barcelona, Spain
jsalaspi@uoc.edu

Abstract. Different types of data privacy techniques have been applied to graphs and social networks. They have been used under different assumptions on intruders' knowledge. i.e., different assumptions on what can lead to disclosure. The analysis of different methods is also led by how data protection techniques influence the analysis of the data. i.e., information loss or data utility.

One of the techniques proposed for graph is graph perturbation. Several algorithms have been proposed for this purpose. They proceed adding or removing edges, although some also consider adding and removing nodes.

In this paper we propose the study of these graph perturbation techniques from a different perspective. Following the model of standard database perturbation as noise addition, we propose to study graph perturbation as *noise graph addition*. We think that changing the perspective of graph sanitization in this direction will permit to study the properties of perturbed graphs in a more systematic way.

Keywords: Data privacy · Graphs · Social networks · Noise addition · Edge removal

1 Introduction

Data privacy has emerged as an important research area in the last years due to the digitalization of the society. Methods for data protection were initially developed for data from statistical offices, and methods focused on standard databases.

Currently, there is an increasing interest to deal with big data. This includes (see e.g. [10,38,40]) dealing with databases of large volumes, streaming data, and dynamic data.

Data from social networks is a typical example of big data, and we often need to take into account the three aspects just mentioned. Data from social networks is usually huge (as the number of users in most important networks are on the millions), posts can be modeled in terms of streaming, and is dynamic (we can consider multiple releases of the same data set).

As data from social networks is highly sensitive, a lot of research efforts have been devoted to develop data privacy mechanisms. Data privacy tools for social networks include disclosure risk models and measures [2, 35], information loss and data utility measures, and perturbation methods (masking methods) [21].

Tools have been developed for the different existing privacy models. We review the most relevant ones below.

1. Differential privacy. In this case, given a query, we need to avoid disclosure from the outcome of the query. Privacy mechanisms are tailored to queries.
2. k-Anonymity. Different competing definitions for k-anonymity for graphs exist, depending on the type of information available to the intruder. Then, for each of these definitions, several privacy mechanisms have been developed. One of the weakest condition is k-degree anonymity. Strongest conditions include nodes neighborhoods.
3. Reidentification. Methods introduce noise into a graph so that intruders are not able to find a node given some background information. Again, we can have different reidentification models according to the type of information available to the intruders (from the degree of a node to its neighborhood). Here noise is usually understood as adding and removing edges from a graph. Nevertheless, adding and removing nodes have also been considered. Methods for achieving k-anonymity are also available for avoiding reidentification.

Note that while differential privacy and k-anonymity are defined as Boolean constraints (i.e., a protection level is fixed and then we can check if this level is achieved or not), this is not the case for reidentification. The risk of reidentification is defined in terms of the proportion of records that can be reidentified.

In this work we focus on methods for graphs perturbation. More particularly, we consider algorithms for graph randomization. In our context, they are the methods that modify a graph to avoid re-identification (and also to achieve k-anonymity). The goal is to formalize data perturbation for graphs, as a model for perturbation of social networks.

Methods for graph randomization usually consist on adding and removing edges. Performance of these methods is evaluated with respect to how the modification modifies the graph (the information loss caused to the graph), and the disclosure risk after perturbation. Often, the execution time is also considered.

The structure of the paper is as follows. In Sect. 2 we review noise addition and random graphs, as the formalization introduced in this paper is based on the former and uses the latter. In Sect. 3 we formalize noise addition for graphs. In Sect. 4, we discuss existing methods for graph randomization from this perspective. In Sect. 5, we present some results related to our proposal. The paper finishes with some conclusions.

2 Preliminaries

In this section we review some concepts and definitions that are needed later on in this paper. We discuss noise addition for standard numerical databases and random graphs.

2.1 Noise Addition

Noise addition has been applied for statistical disclosure control (SDC) for a long time. An overview of different algorithms for noise addition for microdata can be found in [7]. Simple application of noise addition corresponds to replace value x for variable V with mean \bar{V} and variance σ^2 by $x + \epsilon$ with $\epsilon \sim N(0, \sigma^2)$. Correlated noise has also been considered so that noise do not affect correlations computed from perturbed data. See [39] (Chapter 6) for details.

Noise addition is known because it can be used to describe other perturbative masking techniques. In particular, rank swapping, post-randomization and microaggregation are special cases of matrix masking (see [12]). i.e., multiplying the original microdata set X by a record-transforming mask matrix A and by an attribute transforming mask matrix B and adding a displacing mask C to obtain the masked dataset $Z = AXB + C$.

Such operations are well defined for matrices and numbers, here we will define an equivalent operation for graphs.

2.2 Random Graphs

Probability distributions over graphs are usually referred by random graphs. There are different models in the literature. We review some of them here.

The Gilbert Model. This model is denoted by $\mathcal{G}(n, p)$. There are n nodes and each edge is chosen with probability p . With mean degree $d_m = p(n - 1)$ for a graph of n nodes, the probability of a node to have degree k is approximated by $d_m^k e^{-d_m} / k!$. I.e., this probability follows a Poisson distribution.

The Erdős-Rényi Model. This model is denoted by $G(n, e)$, and represents a uniform probability of all graphs with n nodes and e edges. This model is similar to the previous one (basically asymptotically equivalent as [1] writes it).

Most usual networks we find in real-life situations (including all kind of social networks) do not fit well with these two models. For example, it is known that most networks have degree distributions that follow a power-law. Other properties usually present in real-life networks is transitivity (also known as clustering) which means that if nodes v_1 and v_2 are connected, and nodes v_2 and v_3 are also connected, the probability of having v_1 and v_3 connected is high.

Some models extend the previous ones considering degree sequences that follow other distributions than the Poisson. One of such models is based on having a given degree sequence.

Models Based on a Given Degree Sequence. We denote this model by $\mathcal{D}(n, d^n)$, where n is the number of nodes and d^n is a degree sequence. That is,

$d^n \in \mathbb{N}^n$. $\mathcal{D}(n, d^n)$ represents a uniform probability of all graphs with n nodes and degree sequence d^n . Not all degree sequences are allowed (e.g., there is no graph for $\mathcal{D}(5, d^5 = (1, 0, 0, 0, 0))$). A graphical degree sequence is one for which there is at least one graph that satisfies it.

The Havel-Hakimi algorithm [18, 19] is an example of algorithm that builds a graph for a graphical degree sequence. In order to have a graph *drawn* from a uniform distribution $\mathcal{D}(n, d^n)$ we can use [23]. The authors discuss a polynomial algorithm for generating a graph from a distribution arbitrarily close to a uniform distribution, once the degree sequence is known. The algorithm is efficient (see [22]) for p-stable graphs. This type of approach is known as the Markov chain approach, as algorithms are based on switching edges according to given probabilities. See e.g. [26, 27].

Another way to draw a graph according to a uniform distribution from the degree sequence, is to assign to each node in the graph d_i stubs (i.e., edges still to be determined or out-going edges not yet assigned to the in-going ones). Then, choose at random pairs of stubs to settle a proper edge. Nevertheless this does not always work as we may finish with multiple edges and loops (see e.g. [3]).

The Configuration Model. In this case it is considered that the degree sequence of a graph follows a given distribution. To build such a graph we can first randomly choose a graphical degree sequence and then apply the machinery for building a graph from the degree sequence as explained above.

In this configuration model, the degree sequence adds a constraint to the possible graphs. All previous models can be combined with the fact that the degree sequence is given. In fact, other types of constraints can be considered in a model. For example, temporal and spatial.

For example, Spatial graphs may be generated in which the probability of having an edge is proportional to the distance between the nodes [39]. Random dot product graph models, a model proposed by Nickel et al. [29], is another example. In a random dot product graph, each node has associated a vector in an \mathbb{R}^d space, and then the probability of having an edge between two nodes is proportional to the random dot product of the vectors associated to the two nodes.

Given a probability distribution over graphs \mathcal{G} we will denote that we draw a graph G from \mathcal{G} using $G \sim \mathcal{G}$. E.g., $G \sim \mathcal{G}(100, 0.2)$ is a graph G drawn from a Gilbert model with 100 nodes and probability 0.2 of having an edge between two nodes.

3 Noise Addition for Graphs

The main contribution of this paper is to formalize graph perturbation in terms of noise addition. For this, we introduce two different definitions for this type of perturbation. Both of them are based on selecting a graph according to a probability distribution over graphs and *adding* the selected graph to the original one.

We first define what we mean with graph *addition*.

In order to add two graphs, we must align the set of nodes in which we are going to add them. For example if G_1 and G_2 do not have any node in common, then the addition $G_1 \oplus G_2$ is equivalent to the disjoint union $G_1 \cup G_2$.

The next simplest case is when one of the graphs is a subset of the other. Then, the resulting graph will have the largest set of nodes (i.e., the union of both sets of nodes), and the final set of edges is an *exclusive-or* of the edges of both graphs.

Using an *exclusive-or* we can model both addition and removal of edges. Whether we remove or add an edge will depend on whether this edge is in only one graph or in both. Note also that this process is somehow analogous to noise addition where $X' = X + \epsilon$ with ϵ following a normal distribution. In particular, adding noise may correspond to increasing or decreasing the values in X . We formalize below graph addition for subgraphs.

Definition 1. Let $G_1(V, E_1)$ and $G_2(V', E_2)$ be two graphs with $V \subset V'$; then, we define the addition of G_1 and G_2 as the graph $G = (V', E)$ where E is defined

$$E = \{e | e \in V \wedge e \notin V'\} \cup \{e | e \notin V \wedge e \in V'\}$$

We will denote that G is the addition of G_1 and G_2 by

$$G = G_1 \oplus G_2.$$

Note that for any graph G_1 and G_2 , G is a valid graph (i.e., G has no cycles and no multiedges).

We discuss now the definitions of noise graph addition. As one of the graphs, say G , is drawn from a distribution of graphs, the set of vertices created in G are not necessarily connected with the ones in the original graph. Nevertheless, even in this case, we can align (or, better, we still need to align) the two set of nodes. The difference between alternative definitions is about how the two sets of nodes are aligned. So, the general definition will be based on an alignment between the two set of nodes. This alignment corresponds to a homomorphism $A : V' \rightarrow V \cup \{\emptyset\}$. Here, \emptyset corresponds to a null node, or dummy node. This notation comes from graph matching terminology (see e.g., [4]).

Definition 2. Let $G_1(V, E_1)$ and $G_2(V', E_2)$ be two graphs with $|V| \leq |V'|$. An alignment is a homomorphism $A : V' \rightarrow V \cup \{\emptyset\}$.

A matching algorithm builds such homomorphism for a given pair of graphs.

Definition 3. Let $G_1(V, E_1)$ and $G_2(V', E_2)$ be two graphs with $|V| \leq |V'|$. A matching algorithm M is a function that given G_1 and G_2 returns an alignment.

We denote it by $M(G_1, G_2)$. Then, given $A = M(G_1, G_2)$, $A(v)$ for any $v \in V'$ is either a node in V or \emptyset .

Definition 4. Let $G_1(V, E_1)$ be a graph, let \mathcal{G} be a probability distribution on graphs, and let M be a matching algorithm; then, noise graph addition for G_1

following the distribution \mathcal{G} and based on M consists of (i) drawing a graph $G_2(V', E_2)$ from \mathcal{G} such that $|V| \leq |V'|$ (i.e., $G_2 \sim \mathcal{G}$), (ii) aligning it with M obtaining $A = M(G_1, G_2)$, (iii) defining G'_2 as G_2 after renaming the nodes V' into V'' as follows

$$V'' = V \cup \{v' | v' \in V' \text{ and } A(v') = \emptyset\}$$

where $B_V : V' \rightarrow V''$ is, as expected, $B_V(v') = v'$ if $A(v') = \emptyset$ and $B_V(v') = A(v')$ otherwise, (iv) defining $B_E(E_2) = \{(B_V(v_1), B_V(v_2)) | (v_1, v_2) \in E_2\}$ and (v) defining the protected graph $G'(V', E')$ as

$$G' = G_1(V, E_1) \oplus G_2(V'', B_E(E_2)).$$

This definition allows for adding noise graphs that are not necessarily subgraphs of each other.

Another equivalent way, is to think of graphs as labeled graphs, that is, a graph $G = (V, E, \mu)$, where $\mu : V \rightarrow L_V$ is a function assigning labels to the vertices of the graph G . For simplicity we will assume that L_V is a set of integers. Then, given two labeled graphs G_1 and G_2 , their edges would be defined by pairs of integers corresponding to the labels of their corresponding nodes. In such way, an alignment will be given naturally by the nodes that have the same label in both graphs.

Otherwise, if the nodes from two graphs G_1 and G_2 are labeled respectively as $[n_1] = 1, \dots, n_1$ and $[n_2] = 1, \dots, n_2$. To define a different alignment, we may relabel their nodes in the following way.

First, choose two subsets $S_1 \subset [n_1]$ and $S_2 \subset [n_2]$ of the same size, assume that $n_1 \leq n_2$, define a bijection $f : S_2 \mapsto S_1$, then relabel the nodes in $V(G_2)$ as $f(j)$ if $j \in S_2$ and as $n_1 + 1, \dots, n_2$ for the remaining nodes, whose new labels $f(j)$ are not in S_2 . In this way we obtain an alignment. Now we can identify a node with its label and then use Definition 1 for graph addition.

3.1 Graph Matching and Edit Distance

On the other side of noise addition, we find the concept record linkage. In the context of graphs this concept is similar to graph matching. Graph matching is a key task in several pattern recognition applications [41]. However, it has a high computational cost. Indeed, finding isomorphic graphs is NP-complete. Thus, several methods have been devised for efficient approximate graph matching, some use tree structures and strings such as [28, 30, 42], other are able to process very large scale graphs such as [24, 37] or [36].

A commonly used distance for graph matching and finding isomorphic subgraphs is graph edit distance (GED). It was formalized mathematically in [34].

For defining GED, a set of edit operations is introduced, for example, deletion, insertion and substitution of nodes and edges. Then, the similarity of two graphs is defined in terms of the shortest or least cost sequence of edit operations that transforms one graph into the other.

Of course when $GED(G_1, G_2) = 0$ this means that such graphs are isomorphic. Therefore for computing the similarity (or testing an isomorphism) between graphs an exhaustive solution is to define all possible labelings (alignments) and testing for similarity, this is quite inefficient.

However, we may use our definition of addition for defining a distance (that we call edge distance) between two labeled graphs as:

$$ed(G_1, G_2) = |E(G_1 \oplus G_2)|$$

We will prove that ed is a metric in the following theorem.

Theorem 1. *Let \mathcal{G} be the space of labeled graphs without isolated nodes, $G_1, G_2 \in \mathcal{G}$. Let $ed(G_1, G_2) = |E(G_1 \oplus G_2)|$, then the following conditions hold:*

1. $ed(G_1, G_2) = 0 \Leftrightarrow G_1 = G_2$
2. $ed(G_1, G_2) = ed(G_2, G_1)$
3. $ed(G_1, G_3) \leq ed(G_1, G_2) + ed(G_2, G_3)$

Proof. First, $ed(G_1, G_2) = 0$ implies that every $uv \in G_1$ is also in G_2 and the other way around. Hence $E(G_1) = E(G_2)$. Since they do not have isolated nodes, also $V(G_1) = V(G_2)$ and $G_1 = G_2$. Second, $ed(G_1, G_2) = ed(G_2, G_1)$ comes from $G_1 \oplus G_2 = G_2 \oplus G_1$.

Third, for the triangle inequality we calculate two cases for an edge $uv \in E(G_1) \oplus E(G_3)$.

- (i) $uv \in E(G_1)$ and $uv \notin E(G_3)$ In this case, if $uv \in E(G_2)$ then $uv \in E(G_2 \oplus G_3)$. If $uv \notin E(G_2)$ then $uv \in E(G_1 \oplus G_2)$.
- (ii) $uv \in G_3$ and $uv \notin G_1$ This case is equivalent to case *i*) only substituting G_1 by G_3 .

Therefore we conclude that ed is a metric.

In fact, our metric $ed(G_1, G_2)$ actually counts the number of edges we have to modify (erase/add) to transform a graph G_1 to obtain G_2 . Moreover, if we sum $G_1 \oplus (G_1 \oplus G_2)$ we obtain G_2 and $G_2 \oplus (G_1 \oplus G_2) = G_1$.

Therefore we may use our definition for graph addition not only for adding noise, but also for measuring it. This is useful for comparing any published graph with other graphs published under a different privacy protection algorithm (such as a k -anonymous graph).

Moreover, we can use the edge distance to find the median graph G_{median} similar to the median graph obtained in [13] for the graph edit distance.

For a family of graphs \mathcal{F} , we define:

$$G_{median}(\mathcal{F}) = arg \min_{\tilde{G}} \sum_{G_i \in \mathcal{F}} ed(\tilde{G}, G_i)$$

Note that G_{median} is not necessarily unique for a given family of graphs \mathcal{F} , so G_{median} is also a set of graphs.

In the following result we will show that the original graph G is such that is at minimum distance from m protected graphs in $G \oplus \mathcal{G}$ if there is not any edge that belongs to more than $m/2$ of the graphs in \mathcal{G} .

Theorem 2. *Let $\mathcal{F} = G \oplus \mathcal{G}$ where $\mathcal{G} = \{G'_1, \dots, G'_m\}$. If $|\{G'_i \in \mathcal{G} : e \in G'_i\}| \leq \frac{|\mathcal{G}|}{2}$ for all $e \in E(\mathcal{G})$, then $G \in G_{median}(\mathcal{F})$.*

Proof. For each $G'_i \in \mathcal{G}$ denote $G_i = G \oplus G'_i$. Let \tilde{G} any graph different from G . Then, $E(\tilde{G} \oplus G) = A \cup B$, where A denotes the edges in $\tilde{G} \setminus G$ and B the edges in $G \setminus \tilde{G}$. Therefore $\tilde{G} \oplus G_i = \tilde{G} \oplus G \oplus G'_i = (A \cup B) \oplus G'_i$.

For $e \notin A \cup B$, then its either in both $E(G \oplus G_i)$ and $E(\tilde{G} \oplus G_i)$ or in none of both.

For $e \in A \cup B$ and $e \notin E(G'_i)$ then $e \in E((A \cup B) \oplus G'_i) = E(\tilde{G} \oplus G_i)$, in this case, $e \notin E(G'_i) = E(G \oplus G \oplus G'_i) = E(G \oplus G_i)$, no matter whether e was in G or not. So e is not counted in the sum of $|E(G \oplus G_i)|$ but it is counted in $|E(\tilde{G} \oplus G_i)|$

For $e \in A \cup B$ and $e \in E(G'_i)$ then $e \notin E((A \cup B) \oplus G'_i)$. So, $e \notin E(\tilde{G} \oplus G_i)$ but $e \in E(G'_i) = E(G \oplus G_i)$.

We conclude for each edge $e \in A \cup B$ that the cases when $e \in E(G'_i)$, the edge e is counted in $ed(G, G_i)$ but not in $ed(\tilde{G}, G_i)$. While, in the cases when $e \notin E(G'_i)$, the edge e is not counted in $ed(G, G_i)$ but it is counted in $ed(\tilde{G}, G_i)$.

Therefore, since all edges e belong to at most half of the graphs in \mathcal{G} , then $\sum_{G_i \in \mathcal{F}} ed(G, G_i) \leq \sum_{G_i \in \mathcal{F}} ed(\tilde{G}, G_i)$, or equivalently, $G \in G_{median}(\mathcal{F})$.

This provides us with a similar result as in noise addition for SDC in which the expected value $E(x + \epsilon) = x$ for $\epsilon \in N(0, \sigma^2)$.

4 Edge Noise Addition, Local Randomization and Degree Preserving Randomization

In this section we explore the relation of graph noise addition with previous randomization algorithms.

4.1 Edge Noise Addition

The method for graph matching from [28], uses the structure of the neighbors at increasing distance from each node in the graph to grow a tree that is later used to codify the entire graph. Such queries are mentioned in [21] and can be iteratively refined to re-identify nodes in the graph. Hay et al. [20] propose as a solution a graph generalization algorithm similar to [8] and recently improved in [31]. To measure privacy, they sample a graph from all the possible graphs that are consistent with such generalization.

In [20], they have suggested a random perturbation method that consisted on adding m edges and removing m edges. It was later studied from an information-theoretic perspective in [6]. They also suggested a random sparsification method. Other random perturbation methods may be found in [9].

All these perturbation approaches are included in our definition. Observe that they correspond to constraints on the definition of the set of noise graphs (\mathcal{G}) that we are using.

For example, the obfuscation by random sparsification is defined in [6] as follows. The data owner selects a probability $p \in [0, 1]$ and for each edge $e \in E(G)$ the data owner performs an independent Bernoulli trial, $B_e \sim B(1, p)$. And leaves the edge in the graph in case of success (i.e., $B_e = 1$) and remove it otherwise ($B_e = 0$). Letting $E_p = \{e \in E | B_e = 1\}$ be the subset of edges that passed this selection process, the data owner releases the subgraph $G_p = (U = V, E_p)$. Then, they argue that such graph will offer 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 .

Note that random sparsification is obtained with our method by using $\mathcal{G} = \mathcal{G}(n; 1 - p) \cap G$, then adding $G \oplus G'$ for some $G' \in \mathcal{G}$.

Another relevant example is when we define $\mathcal{G} = \{G' : |E(G')| = m\}$, then $G \oplus G'$ where $G' \sim \mathcal{G}$ is equivalent to adding x and deleting $m - x$ edges, in total changing m edges from the original graph. If we restrict \mathcal{G} to be the family of graphs G' such that $|E(G')| = 2m$ and $|E(G') \cap E(G)| = m$, then we are adding m edges and deleting m other edges, as in [20].

Note that, in this case all the m edges may be incident to the same node or they may be all independent, however such restrictions could be added with our method simply by specifying that the graphs G' in \mathcal{G} do not have degree greater than $m - 1$, or that their maximum degree is at most 1.

We will show an example of such a local restriction in the following subsection.

4.2 Local Randomization

In [43] a comparison between edge randomization and k -anonymity is carried out, considering that the adversary knowledge is the degree of the nodes in the original graph. There, they calculate the prior and posterior risk measures of the existence of a link between node i and j (after adding k and removing k edges) as:

$$\begin{aligned}
 P(a_{ij} = 1) &= \frac{m}{N} \\
 P(a_{ij} = 1 | \tilde{a}_{ij} = 1) &= \frac{m - k}{m} \\
 P(a_{ij} = 1 | \tilde{a}_{ij} = 0) &= \frac{k}{N - m}
 \end{aligned}
 \tag{1}$$

We define $G \oplus G_u^t$ to be the local t -randomization which adds the graph G_u^t with vertex set $V(G_u^t) = u, u_1, \dots, u_t$ and edge set $E(G_u^t) = uu_1, \dots, uu_t$ in which u is a fixed vertex from G and $u_1, \dots, u_t \subset V(G \setminus u)$.

Then $G \oplus G_u^t$ changes t random edges incident to u in G . So we can apply local t -randomization for all u in $V(G)$ to obtain the graph

$$G^t = G \bigoplus_{u \in V(G)} G_u^t$$

Following the notation from [43], we compare the privacy guarantees of local t -randomization to the equivalent that would be $k = tn/2$ in their case. This value comes from the fact that each node has t randomized edges and therefore are counted twice.

We consider an adversary that knows about a given node and its degree. Then tries to infer if the adjacent edges in the published graph come from the original.

For a given node i we denote its degree as d_i and as \bar{d}_i the value $n - 1 - d_i$, in general for any value t , we denote as \bar{t} the value $n - 1 - t$, this is the complement of the degree d_i and the complement of the edges in $E(G_u^t)$ respectively.

Theorem 3. *The adversary's prior and posterior probabilities to predict whether there is a sensitive link between two target individuals $i, j \in V(G)$ by exploiting the degree d_i and accessing to the randomized graph G^t are the following:*

- (i) the probability $P(a_{ij} = 1) = \frac{d_i}{n - 1}$
- (ii) the probability $P(a_{ij} = 1 | a_{ij}^t = 1)$ is equal to:

$$\frac{d_i(\bar{t}^2 + t^2)}{d_i(\bar{t}^2 + t^2) + 2\bar{d}_i t t}$$

- (iii) the probability $P(a_{ij} = 1 | a_{ij}^t = 0)$ is equal to:

$$\frac{2d_i \bar{t} t}{2d_i t t + \bar{d}_i(\bar{t}^2 + t^2)}$$

Proof. (i) First of all, $P(a_{ij} = 1) = \frac{d_i}{n - 1}$ because the adversary knows d_i and there are only $n - 1$ possible neighbors for i

(ii) Now we calculate $P(a_{ij} = 1 | a_{ij}^t = 1)$ in our graph G^t .

There are only two possibilities for an edge to belong to $E(G^t)$ (i.e., $a_{ij}^t = 1$) and to G (i.e., $a_{ij} = 1$). The edge was already in $E(G)$ and it does not belong to $E(G_i^t)$ neither to $E(G_j^t)$ or it is in $E(G)$ and at the same time belongs to both $E(G_i^t)$ and $E(G_j^t)$.

For an edge to belong to $E(G^t)$ and not to G (i.e., $a_{ij} = 0$). There are also two cases, $a_{ij} = 0$ and the edge belongs to exactly one of $E(G_i^t)$ or $E(G_j^t)$.

Considering that the probability that $ij \in E(G^t) = \frac{t}{n-1}$, the probability that $ij \notin E(G^t) = \frac{\bar{t}}{n-1}$, $P(a_{ij} = 1) = \frac{d_i}{n-1}$ and $P(a_{ij} = 0) = \frac{\bar{d}_i}{n-1}$ we obtain the following:

$$\frac{d_i \bar{t}^2 + d_i t^2}{d_i \bar{t}^2 + d_i t^2 + \bar{d}_i t t + \bar{d}_i t t}$$

which yields (ii).

For (iii) we follow a similar argument as in (ii). For $P(a_{ij} = 1 | a_{ij}^t = 0)$ there are also two possibilities for an edge that was in $E(G)$ to be removed from $E(G^t)$. The edge ij is in $E(G)$ and either $ij \in E(G_i^t) \setminus E(G_j^t)$ or $ij \in E(G_j^t) \setminus E(G_i^t)$. For an edge to belong to G and not to $E(G^t)$. There are also two more cases, $a_{ij} = 1$ and the edge belongs to both $E(G_i^t)$ and $E(G_j^t)$, or to none of them. This implies that:

$$P(a_{ij} = 1 | a_{ij}^t = 0) = \frac{d_i \bar{t}t + d_i t \bar{t}}{d_i \bar{t}t + d_i t \bar{t} + \bar{d}_i \bar{t}^2 + \bar{d}_i t^2}$$

Which simplifies to (iii) and finishes the proof.

4.3 Degree Preserving Randomization

As we have discussed in the introduction, graphs with a given degree sequence can be uniformly generated by starting from a graph $G \in \mathcal{D}(n, d^n)$ and swapping some of its edges. The method of swap randomization is also used for generating matrices with given margins for representing tables and assessing data mining results in [14].

Recall that a swap in a graph consists of choosing two edges $u_1 u_2, u_3 u_4 \in E(G)$, such that $u_2 u_3, u_4 u_1 \notin E(G)$ to obtain the graph \tilde{G} such that $V(\tilde{G}) = V(G)$ and $E(\tilde{G}) = E(G) \setminus \{u_1 u_2, u_3 u_4\} \cup \{u_2 u_3, u_4 u_1\}$. It is also called alternating circuit in [39] were it is used for proving that all multigraphs may be transformed into graphs via alternating circuits, and used for generating synthetic spatial graphs.

In [32] and [33] this is used to prove that scale-free sequences with parameter $\gamma > 2$ are P-stable and graphic. Note that the P-stability property guarantees that a graph with a given degree sequence may be uniformly generated by choosing a graph $G \in \mathcal{D}(n, d^n)$ by applying sufficient random swaps.

Then for a given graph G with n nodes labeled as $[n] = 1, \dots, n$ We define the family $\mathcal{S}_G = G'$ such that $G' = \{i, j, k, l\} \subset [n]$, where $ij, kl \in E(G)$ and $jk, li \notin E(G)$. In other words \mathcal{S}_G is the set of alternating 4-circuits of G .

We may choose an arbitrary number m of such noise graphs G'_1, \dots, G'_m and add them to G , that is $G = G \oplus G'_1 \oplus \dots \oplus G'_m$.

Following this procedure for m large enough is equivalent to randomizing G to obtain all the graphs $\mathcal{D}(n, d^n)$. Actually, for any two graphs with the same degree sequence $G_1, G_2 \in \mathcal{D}(n, d^n)$, $G' = G_1 \oplus G_2$ must have at each node i the same number of edges ij that belong to $E(G)$ as the edges il that do not belong to $E(G)$, since the node i has the same degree in G_1 and in G_2 . Thus, any graph in $\mathcal{D}(n, d^n)$ may be generated by starting at some graph G and adding G' from the set \mathcal{G} of all the graphs that are a union of alternating circuits of G .

5 The Most General Approach for Noise Addition for Graphs

In all the previous sections we were restrictive with respect to the noise graphs that we were adding, in this last section we apply the Gilbert model which was discussed in Sect. 2.2, without any restriction.

This has as a consequence that we may not be able to obtain strong conclusions on the structure of the graphs obtained after noise addition, in contrast to the results in previous sections. However, by reducing the restrictions, we are increasing the possible noise graphs that we add, hence we increase uncertainty and therefore protection. In Table 1 we present the noise addition methods discussed in this paper, together with some of their properties and requirements.

Table 1. Graph perturbation methods in terms of graph noise addition.

Noise addition method	Definition of \mathcal{G}	Additional requirements for $G' \in \mathcal{G}$	Properties of $G \oplus \mathcal{G}$
Random perturbation [20]	$ E(G') = 2m$	$ E(G') \cap E(G) = m$ $ E(G') \cap E(\overline{G}) = m$	G' adds m edges and removes m edges
Random sparsification [6]	$G' \in \mathcal{G}(n; 1 - p) \cap G$	None	The edges of G remain with probability p , no added edges
Local t -randomization	$G' = G_u^t$	Applied to every node in G	Every node has t modified incident edges
Degree preserving randomization [5]	$G' \in \mathcal{S}_G$	\mathcal{S}_G is the set of swaps of G	$G, G \oplus G' \in \mathcal{D}(n, d^n)$
Gilbert model	$G' \in \mathcal{G}(n; 1 - p)$	None	Every edge is added or removed with probability p

Proposition 1. *Let $G_1(V, E_1)$ be an arbitrary undirected graph, and let $G_2(V, E_2)$ be an undirected graph generated from a Gilbert model $\mathcal{G}(|V|, p)$. Then, the expected number of edges of G_2 is $p \cdot |V| \cdot (|V| - 1)/2$.*

Proposition 2. *Let $G_1(V, E_1)$ be an arbitrary undirected graph with $n_1 = |E_1|$ edges, and let $G_2(V, E_2)$ be an undirected graph generated from a Gilbert model with n_2 edges. Then, the noise graph addition of G_1 and G_2*

$$G' = G_1(V, E_1) \oplus G_2(V, E_2)$$

will have on average $|E'| = n_2 \cdot (t - n_1)/t + (t - n_2) \cdot n_1/t$ edges, where $t = |V| \cdot (|V| - 1)/2$.

Proof. Let the graph G_1 have $n_1 = |E_1|$ edges, and let the graph G_2 have n_2 edges. Being the graph undirected, the number of edges for a complete graph is $t = v * (v - 1)/2$ where $v = |V|$ is the number of nodes. Then, we will have on average that

- $n_2 \cdot n_1/t$ edges of G_2 link two nodes also linked by G_1 ,
- $n_2 \cdot (t - n_1)/t$ edges of G_2 link two nodes not linked by G_1 ,
- $(t - n_2) \cdot n_1/t$ edges of G_1 link two nodes not linked by G_2 , and
- $(t - n_2)(t - n_1)/t$ corresponds to pairs of nodes neither linked in G_1 nor in G_2 .

So, addition of graphs G_1 and G_2 result into a graph with $n_2 \cdot (t - n_1)/t + (t - n_2) \cdot n_1/t$ edges on average.

From this, it follows that only when $n_2 = 0$, the number of edges of G' is the same as the ones in G_1 ; or when $n_1 = t/2$. This is expressed as a lemma below.

Lemma 1. *Let $G_1(V, E_1)$ and $G_2(V, E_2)$ be as above with n_1 and n_2 the corresponding number of edges. Then, the noise graph addition of G_1 and G_2 , $G' = G_1(V, E_1) \oplus G_2(V, E_2)$ has n_1 nodes when $n_2 = 0$ or when $n_1 = t/2$.*

Proof. The solutions of $n_1 = n_2 \cdot (t - n_1)/t + (t - n_2) \cdot n_1/t$ are such that $tn_1 = n_2t - n_2n_1 + tn_1 - n_2n_1$ or, equivalently, $0 = n_2t - n_2n_1 + -n_2n_1$. So, one solution is that $n_2 = 0$, and if this is not the case, $n_1 = t/2$.

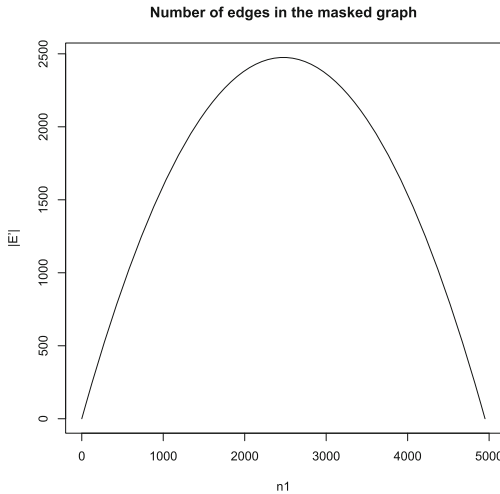


Fig. 1. Average number of edges in G' when the graph G_1 has 100 nodes and, thus, $t = 4950$, and n_1 is in the range $[0,4950]$. G_2 generated using the Gilbert model $\mathcal{G}(|V|, p)$ with p as in the graph $G(V, E_1)$ (i.e., the expected number of edges satisfying $|E_2| = |E_1|$).

Let us consider the case that we use our original graph to build a Gilbert model $\mathcal{G}(|V|, p)$ from the graph $G_1(V, E_1)$. That is, we generate a graph with

Table 2. Number of nodes and edges in some of small graphs in the literature using the Gilbert model $\mathcal{G}(|V|, p)$ with p as in the graph $G(V, E_1)$ (i.e., the expected number of edges satisfying $|E_2| = |E_1|$).

Name	Reference	Nodes	Edges	Edges added	% of edges added
Jazz	[16]	198	5484	2400	43.7
Karate	[44]	34	156	69	44.3
Football	[15]	115	1226	767	62.5
Erdos971	[45]	472	2628	2503	95.2
Urvemail	[17]	1133	10902	10531	96.5
CElegans	[11]	453	4050	3729	92.0
Caida	[25]	26475	106762	106697	99.9

$p = |E_1|/(|V| * (|V| - 1)/2)$. Then, the expected value for n_2 is $n_2 = n_1 = |E_1|$. Therefore, noise graph addition of G_1 and G_2 will have $n_1 \cdot (t - n_1)/t + (t - n_1) \cdot n_1/t = (2n_1t - 2n_1n_1)/t$ edges.

Figure 1 illustrates this case for the case of 100 nodes and edges between 0 and 4950 (i.e., the case of a complete graph). As proven in the lemma, only when $n_1 = 0$ or when $n_1 = t/2$ we have that the number of edges in the resulting graph is exactly the same as in the original graph (i.e., n_2 is 0 or $t/2$, respectively). We can also see that while $n_1 \leq 4950/2$, the number of edges in the resulting graph is not so different to the ones on the original graph (maximum difference is at $n_1 = t/4$ with a difference of $t/8$ edges), but when $n_1 > 4950/2$ the difference starts to diverge. Maximum difference is when the graph is complete that addition will result into the graph with no edges.

Observe that difference between the edges in the added graph and the original one will be $(2n_1t - 2n_1n_1)/t - n$ and that the maximum difference is at $n_1 = t/4$. For $n_1 = t/4$ the difference $(2n_1t - 2n_1n_1)/t - n = t/8$. For $n_1 = t/4$ this means a maximum of 50% increase in the number of edges.

Nevertheless, if we consider the proportion of edges added with respect to the total number of edges, this is $1 - 2 * n_1/t$, which implies that the proportion is decreasing with maximum equal to 100% when $n_1 = 1$. Naturally, when $n_1 \geq t/2$ we start to have deletions instead of additions and with $n_1 = t$ we have a maximum number of deletions (i.e., 100% because all edges are deleted).

Let us consider a few examples of graphs used in the literature. For each graph, Table 2 shows for a few examples of graphs, the number of nodes and edges, the expected number of edges added using a Gilbert model, and the expected increase in the number of edges. It can be seen that for large number of nodes, as the actual number of edges is rather low with respect to the ones in a complete graph, the % of increment of edges is around 100%.

6 Conclusions

In this paper we have introduced a formalization for graph perturbation based on noise addition. We have shown that some of the existing approaches for graph randomization can be understood from this perspective. We have also proven some properties for this approach, noted that it encompasses many of the previous randomization approaches while also defining a metric for graph modification. Additionally, we defined the local t -randomization approach which guarantees that every node has t -neighboring edges modified. It remains as future work to study new families of noise graphs and test their privacy and utility guarantees.

Acknowledgments. This work was partially supported by the Swedish Research Council (Vetenskapsrådet) project DRIAT (VR 2016-03346), the Spanish Government under grants RTI2018-095094-B-C22 “CONSENT” and TIN2014-57364-C2-2-R “SMARTGLACIS”, and the UOC postdoctoral fellowship program.

References

1. Aiello, W., Chung, F., Lu, L.: A random graph model for power law graphs. *J. Exp. Math.* **10**, 53–66 (2001)
2. Balsa, E., Troncoso, C., Díaz, C.: A metric to evaluate interaction obfuscation in online social networks. *Int. J. Unc. Fuzz. Knowl.-Based Syst.* **20**, 877–892 (2012)
3. Bannink, T., van der Hofstad, R., Stegehuis, C.: Switch chain mixing times through triangle counts. [arXiv:1711.06137](https://arxiv.org/abs/1711.06137) (2017)
4. Bengoetxea, E.: Inexact graph matching using estimation of distribution algorithms, PhD Dissertation, Ecole Nationale Supérieure des Télécommunications, Paris (2002)
5. Berge, C.: *Graphs and Hypergraphs*. North-Holland, Netherlands (1973)
6. Bonchi, F., Gionis, A., Tassa, T.: Identity obfuscation in graphs through the information theoretic lens. *Inf. Sci.* **275**, 232–256 (2014)
7. Brand, R.: Microdata protection through noise addition. In: Domingo-Ferrer, J. (ed.) *Inference Control in Statistical Databases*. LNCS, vol. 2316, pp. 97–116. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-47804-3_8
8. Campan, A., Truta, T.M.: Privacy, security, and trust in KDD, pp. 33–54 (2009)
9. Casas-Roma, J., Herrera-Joancomartí, J., Torra, V.: A survey of graph-modification techniques for privacy-preserving on networks. *Artif. Intell. Rev.* **47**(3), 341–366 (2017)
10. D’Acquisto, G., Domingo-Ferrer, J., Kikiras, P., Torra, V., de Montjoye, Y.-A., Bourka, A.: Privacy by design in big data: an overview of privacy enhancing technologies in the era of big data analytics, ENISA Report (2015)
11. Duch, J., Arenas, A.: Community identification using extremal optimization. *Phys. Rev. E* **72**, 027–104 (2005)
12. Duncan, G.T., Pearson, R.W.: Enhancing access to microdata while protecting confidentiality: prospects for the future. *Statist. Sci.* **6**(3), 219–232 (1991)
13. Ferrer, M., Valveny, E., Serratosa, F.: Median graph: a new exact algorithm using a distance based on the maximum common subgraph. *Pattern Recogn. Lett.* **30**(5), 579–588 (2009)

14. Gionis, A., Mannila, H., Mielikäinen, T., Tsaparas, P.: Assessing data mining results via swap randomization. *ACM Trans. Knowl. Discov. Data* **1**(3), 14 (2007)
15. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. *Proc. Nat. Acad. Sci. US Am.* **99**(12), 7821–7826 (2002)
16. Gleiser, P., Danon, L.: Community structure in jazz. *Adv. Complex Syst.* **6**, 565 (2003)
17. Guimera, R., Danon, L., Diaz-Guilera, A., Giralt, F., Arenas, A.: Self-similar community structure in a network of human interaction. *Phys. Rev. E* **68**, 065103(R) (2003)
18. Hakimi, S.L.: On realizability of a set of integers as degrees of the vertices of a linear graph I. *J. Soc. Ind. Appl. Math.* **10**, 496–506 (1962)
19. Havel, V.: A remark on the existence of finite graphs. *Časopis Pro Pěstování Matematiky* (in Czech) **80**, 477–480 (1955)
20. Hay, M., Miklau, G., Jensen, D., Weis, P., Srivastava, S.: Anonymizing social networks, Technical report No. 07-19, Computer Science Department, University of Massachusetts Amherst, UMass Amherst (2007)
21. Hay, M., Miklau, G., Jensen, D., Towsley, D., Weis, P.: Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow.* **1**(1), 102–114 (2008)
22. Jerrum, M., McKay, B.D., Sinclair, A.: When is a graphical sequence stable? In: Frieze, A., Luczak, T. (eds.) *Random Graphs*, vol. 2, pp. 101–115. Wiley-Interscience, Hoboken (1992)
23. Jerrum, M., Sinclair, A.: Fast uniform generation of regular graphs. *Theoret. Comput. Sci.* **73**, 91–100 (1990)
24. Koutra, D., Shah, N., Vogelstein, J.T., Gallagher, B., Faloutsos, C.: Deltacon: principled massive-graph similarity function with attribution. *ACM Trans. Knowl. Discov. Data* **10**(3), 28:1–28:43 (2016)
25. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data* **1**(1), 1–40 (2007)
26. Miklós, I., Erdős, P.L., Soukup, L.: Towards random uniform sampling of bipartite graphs with given degree sequence. *Electr. J. Comb.* **20**(1), 16 (2013)
27. Milo, R., Kashtan, N., Itzkovitz, S., Newman, M.E.J., Alon, U.: On the uniform generation of random graphs with prescribed degree sequences. [arXiv:cond-mat/0312028v2](https://arxiv.org/abs/cond-mat/0312028v2) (2004)
28. Nettleton, D.F., Salas, J.: Approximate matching of neighborhood subgraphs - an ordered string graph levenshtein method. *Int. J. Unc. Fuzz. Knowl.-Based Syst.* **24**(03), 411–431 (2016)
29. Nickel, C.L.M.: Random dot product graphs: a model for social networks, PhD. dissertation, Maryland (2006)
30. Robles-Kelly, A., Hancock, E.R.: String edit distance, random walks and graph matching. *Int. J. Pattern Recogn. Artif. Intell.* **18**(03), 315–327 (2004)
31. Ros-Martín, M., Salas, J., Casas-Roma, J.: Scalable non-deterministic clustering-based k-anonymization for rich networks. *Int. J. Inf. Secur.* **18**(2), 219–238 (2019)
32. Salas, J., Torra, V.: Graphic sequences, distances and k-degree anonymity. *Discrete Appl. Math.* **188**, 25–31 (2015)
33. Salas, J., Torra, V.: Improving the characterization of p-stability for applications in network privacy. *Discrete Appl. Math.* **206**, 109–114 (2016)
34. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.* **SMC-13**(3), 353–362 (1983)
35. Stokes, K., Torra, V.: Reidentification and k-anonymity: a model for disclosure risk in graphs. *Soft Comput.* **16**(10), 1657–1670 (2012)

36. Sun, Z., Wang, H., Wang, H., Shao, B., Li, J.: Efficient subgraph matching on billion node graphs. *Proc. VLDB Endow.* **5**(9), 788–799 (2012)
37. Tian, Y., Patel, J.M.: Tale: a tool for approximate large graph matching. In: 2008 IEEE 24th International Conference on Data Engineering, pp. 963–972, April (2008)
38. Torra, V.: *Data Privacy: Foundations, New Developments and the Big Data Challenge*. SBD, vol. 28. Springer, Cham (2017). <https://doi.org/10.1007/978-3-319-57358-8>
39. Torra, V., Jonsson, A., Navarro-Arribas, G., Salas, J.: Synthetic generation of spatial graphs. *Int. J. Intell. Syst.* **33**(12), 2364–2378 (2018)
40. Torra, V., Navarro-Arribas, G.: Big data privacy and anonymization. In: Lehmann, A., Whitehouse, D., Fischer-Hübner, S., Fritsch, L., Raab, C. (eds.) *Privacy and Identity 2016*. IAICT, vol. 498, pp. 15–26. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-55783-0_2
41. Vento, M.: A long trip in the charming world of graphs for pattern recognition. *Pattern Recogn.* **48**(2), 291–301 (2015)
42. Yan, X., Han, J.: gspan: graph-based substructure pattern mining. In: 2002 IEEE International Conference on Data Mining, 2002. Proceedings., pp. 721–724, December (2002)
43. Ying, X., Pan, K., Wu, X., Guo, L.: Comparisons of randomization and k-degree anonymization schemes for privacy preserving social network publishing. In: *Proceedings of the 3rd Workshop on Social Network Mining and Analysis, ser. SNA-KDD 2009*, pp. 10:1–10:10. New York, NY, USA, ACM (2009)
44. Zachary, W.W.: An information flow model for conflict and fission in small groups. *J. Anthropol. Res.* **33**, 452–473 (1977)
45. <http://vlado.fmf.uni-lj.si/pub/networks/pajek/data/gphs.htm>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

