International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems
© World Scientific Publishing Company

DATA PRIVACY FOR SIMPLY ANONYMIZED SOCIAL NETWORK LOGS REPRESENTED AS GRAPHS - CONSIDERATIONS FOR GRAPH ALTERATION OPERATIONS

DAVID F. NETTLETON[1,2]
[1]*Artificial Intelligence Research Institute, IIIA*
Spanish National Research Council, CSIC
Campus Universitat Autònoma de Barcelona 08193 Bellaterra,
Catalonia (Spain).
dnettleton@iiia.csic.es
[2]*Web Research Group, Department of Technology*
Pompeu Fabra University, 08018 Barcelona, Catalonia (Spain).

VICENÇ TORRA
Artificial Intelligence Research Institute, IIIA
Spanish National Research Council, CSIC
Campus Universitat Autònoma de Barcelona 08193 Bellaterra,
Catalonia (Spain).
vtorra@iiia.csic.es

Received (received date)
Revised (revised date)
Accepted (accepted date)

In this paper we review the state of the art on graph privacy with special emphasis on applications to online social networks, and we consider some novel aspects which have not been greatly covered in the specialized literature on graph privacy. The following key considerations are covered: (i) choice of different operators to modify the graph; (ii) information loss based on the cost of graph operations in terms of statistical characteristics (degree, clustering coefficient and path length) in the original graph; (iii) computational cost of the operations; (iv) in the case of the aggregation of two nodes, the choice of similar adjacent nodes rather than isomorphic topologies, in order to maintain the overall structure of the graph; (v) a statistically knowledgeable attacker who is able to search for regions of a simply anonymized graph based on statistical characteristics and map those onto a given node and its immediate neighborhood.

Keywords: data privacy, graphs, operators, heuristics.

1. Introduction and Motivation

Data Privacy in Social Network logs has currently become a key issue. Hundreds of millions of users all over the world are generating high volume data logs of their online

social network activity and relations. On the one hand, this data represents a threat to an individuals' data privacy, and on the other hand the data offers a great analysis opportunity to data miners. If we can sufficiently protect the data by anonymization techniques, then we can make the social network log data publicly available for commercial and academic use.

As an example of a real dataset, consider the Enron emails dataset[1,2,3]. The Enron email dataset consists of a collection of 150 folders corresponding to the emails to and from senior management and others at Enron, collected over a period between 1998 to 2002. The total number of emails sent/received between users is approx. 1.5 million. Each email sender/recipient represents a node in the graph and the activity is represented by the number of emails sent-received along the edges which connect the users. From this dataset we derive the network structure (an edge is created between two nodes 'a' and 'b' when at least one email is sent between them). We note that this is a directed graph: 'a' may send one or more emails to 'b', but 'b' may not send any emails to 'a'. Apart from the structure of the graph itself, the activity data is also derived from the same dataset.

As another example of a real dataset, consider the Facebook New Orleans dataset. This dataset was generated by Viswanath et al[4] by crawling the Facebook New Orleans regional network, and consists of approx. 60,000 users, 1.5 million links between users, and 800,000 logged interactions over a two year period. In contrast to the Enron dataset, for which a link between users is established when an email is sent/received between them, in the case of the Facebook users, a link is established by the explicit solicitation and acceptance of friendship. Also, in the Facebook dataset, 'writes to wall' is the activity indicator, however it can be argued that users also communicate by other methods, such as the 'chat' option of Facebook, or another email system (messenger, ...), social network (Twitter, ...) or by some other medium (telephone, physical contact, ...). Each user in Facebook has a 'wall' which is a public 'message board' on which they themselves or other users who are 'friends' can publish and share their messages, links, photos, and general online content.

We observe that in the case of other social network datasets where the links/edges are derived from user interaction, such as those based on emails sent/received, the derived graph is a truer representation of 'live' links. However, for these datasets, the potential problem of spammers and other types of emails (broadcasting, ...) has to be taken into consideration when measuring activity.

The activity logs we consider have already been simply anonymized, that is, the user ids/names have been substituted by randomly generated user ids. The only other information in the dataset is the timestamp when a link was created or when a write to wall was made. However, a key problem with graph anonymization, which does not occur for 'tabular data', is that even though the user ids consist of anonymous numbers, the characteristics of the links between them can make individual users and subgroups identifiable. Therefore, graph anonymization has to try to retain the statistical characteristics of the graph which serve for bona fide analysis, while making it as

difficult as possible for a potential attacker to identify individuals and/or subgroups of individuals.

In the recent literature on graph privacy, different attack and defense strategies have been proposed, based on classic graph theory and on the specific nature of online social networks. Proposed anonymization methods in the literature, include: adding and deleting edges, k-partitioning which aggregates nodes in k partitions, creating a list of possible labels for each node, and so on. In the case of adding edges, an effective predictive model exists which serves as a heuristic for choosing which edges to add while minimizing the information loss. On the other hand, attack methods include: crawling and querying the graph using different strengths of query and auxiliary knowledge; passive and active attacks; the infiltration of the graph with 'sybil' nodes; use of malicious websites and auxiliary information, and so on.

In Section 2 of the paper, we review the state of the art on graph privacy with special emphasis on applications to online social networks. In this review we identify some areas which can be improved or which have not been considered fully in the specialized literature on graph privacy, such as using a complete set of operators to modify the graph, basing the choice of elements (nodes/edges) for modification on the statistical properties of the graph, basing the re-linking of local neighborhoods on the statistical properties of the neighborhood, and the choice of adjacent nodes based on similarity instead of looking for isomorphisms globally, as candidates for node aggregation.

In the light of this, in Section 3 we will consider the following key aspects:

(i) Choice of six different operators to modify the graph;
(ii) Calculation of the information loss based on the cost of graph operations in terms of statistical characteristics (degree, clustering coefficient and path length) in the original graph;
(iii) Consideration of the computational cost of the operations;
(iv) Choice of similar adjacent nodes rather than isomorphic topologies (which could be in any part of the graph) for node aggregation, in order to maintain the overall structure of the graph;
(v) Assumption of a statistically knowledgeable attacker who is able to search for regions of a simply anonymized graph based on statistical characteristics and map those onto a given node and its immediate neighborhood.

The structure of the paper is as follows: in Section 2 we present the state of the art in graph privacy applied to social networks; in Section 3 we consider how the graph statistics are calculated in order to quantify information loss for six different graph modifier operators, using a simple topology as an example to calculate the key graph statistics of degree, clustering coefficient and path length; finally, in Section 4 we summarize the present study.

Note that in Section 3.2 we comment the incorporation of network activity as a weight on the topology. However, this is not incorporated into the statistical calculations for information loss, but it is left for a following stage.

2. Related Work and State of the Art

Privacy in on-line social networks is a relatively new area of research which however has a solid base in classic graph theory and data privacy concepts such a k-anonymity[5]. For the purposes of the present study, we will divide the authors into the following themes: attack/defense/both; techniques based on/not based on k-anonymization; techniques using auxiliary information (additional to the simply anonymized graph structure).

Table 1. Authors classified by different approaches/themes

Theme	Sub-Theme	Authors
Focus	Attack	Narayanan[6], Backstrom[7]
	Defense	Dwork[8], Liu[9]
	Both	Hay[10,11], Zhou[12], Bhagat[13]
Anonymization algorithm	k-anonymization	Hay[10,11], Liu[9], Zhou[12], Bhagat[13].
	Other	Dwork[8] (e-indistinguishability)
Adversary information	Only simply anonymized graph topology	Hay[10,11], Liu[9], Zhou[12]
	Additional information	Narayanan[6] (overlap between two social networks), Bhagat[13]
	Sybil nodes	Backstrom[7]

In Table 1 we can see a summary of the approaches and focuses of different authors based on these themes. In the present work, we are particularly interested in the attack and defense of a simply anonymized social network graph without additional information being available to the adversary and without the creation of sybil nodes.

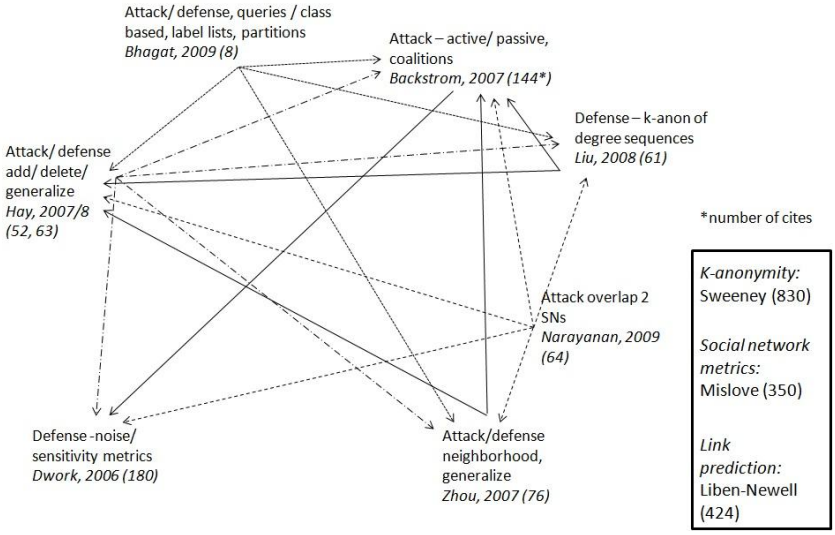


Fig. 1. Schematic graph of mutual citations of authors included in the state of the art

Fig. 1 shows a schematic representation as a directed graph, of the key publications of the state of the art authors for graph privacy, showing their inter-relation in terms of cross-references. For example, Hay references four other key authors: Backstrom[7], Liu[9], Zhou[12] and Dwork[8]. On the other hand, Hay[10,11] is referenced by Bhagat[13], Liu[9], Narayanan[6] and Zhou[12]. In the bottom right of Fig. 1 we also show three highly cited references, but which do not reciprocally cite the other authors, because the publications were previous to the current state of the art, or because they deal with more theoretical concepts of graphs, data privacy[5] or metrics[14].

2.1 *Attack methods*

On the one hand, attack methods can contemplate the analysis of simply anonymized data such as in the studies presented by Hay[10,11] and Zhou[12]. On the other hand, attack methods can make use of auxiliary information which may be available (or obtainable), in order to de-anonymize users and sub-graphs. The studies presented by Narayanan[6], Backstrom[7] and Bhagat[13] exemplify this latter approach. In the following section, we will firstly consider the use of auxiliary information in the attack, and then we will look at attacks for which auxiliary information is absent.

One type of auxiliary information is that obtained by interrelating different information sources. Narayanan[6] presents an attack method, based on background information derived from the overlap between the Twitter and Flickr online social networks. The Twitter, Flickr and LiveJournal social networks were also considered individually. The node mapping re-identification algorithm has two main steps: (i) select seeds (nodes in the network); (ii) extend seeds (propagate outwards mapping the network) using an iterative method. It makes use of the following properties: eccentricity, edge directionality and node degrees. The method revisits nodes to remap when necessary and considers reverse matching between the original graph and the new graph (that is, the target graph and the original graph are interchangeable). The risk measure is the percentage of nodes affected, weighted using a node centrality measure. Each node has a weight which is based on its centrality (the number of connections a given node has, divided by the total number of nodes in the graph). The results showed a 31% correct identification of nodes; 41% of the incorrectly identified were at a distance of 1 from the target node; and 55% of the incorrect nodes were in the same geographical location as the target node.

Another way of obtaining auxiliary information is by 'infiltrating' the network we wish to expose. Backstrom[7] presents an 'active' and a 'passive' attack on a dataset extracted from the LiveJournal application. 'Active' implies the creation of k new 'sybil' nodes to infiltrate the network, whereas 'passive' implies the use of existing user accounts which 'find themselves' in the network. The active method has the following steps: choose victims (nodes), create sybil nodes, create pattern of links; then it tries to find the defined structure in the complete graph using two different search methods. The passive

method requires a collusion between *k* existing users, and makes use of the observation that most nodes in a real social network belong to a small uniquely identifiable sub-graph. Thus, if a user is in coalition with *k-1* other users after the release of the network, this user will be able to identify additional nodes which are connected to the coalition, and therefore learn the edge relations between them. The risk measure (number of nodes identified) for the active attack gave that a 90% probability of identification was achieved using a coalition of k=7. In the case of the passive attack, a smaller coalition (k=5) gave the same probability of identification (90%). However, in the active attack, a limit was assigned to the degrees of the users created (in order to remain inconspicuous) whereas for the passive attack, there was no such limit.

In Bhagat[13], the attack is based on querying the anonymized data. Example queries select subpopulations and try to identify patterns of interactions between them, such as: (i) how many users are in a subpopulation; (ii) patterns of interaction over a given period of time (day, month, ..); (iii) establish if a graph of interactions can be partitioned with small 'cuts'. The risk measure is calculated as a 'privacy level' for different values of *k* and *m*, where is *k* equal to the number of labels and the frequency of appearance in a list, and *m* is equal to the number of partitions (size of the classes). Also, *age groupings* are used for identification purposes. The risk is then measured as the percentage error on classifying based on different values of *k*, *m*, and the age groupings.

In the context of attacks without auxiliary information, that is, based only on a simply anonymized graph dataset, we will now consider the studies of Hay[10], and Zhou[12].

Hay[10,11] considers an attack method which attempts re-identification using two types of queries: vertex refinement and sub-graph queries. The risk measure is considered as the percentage of nodes whose equivalent candidate set falls into one of a given set of buckets (1 node, 2-4 nodes, 5-10 nodes, ...). In Ref. 11, the attack methods (vertex refinement and sub-graph queries) are the same as in Ref. 10, as are the risk measures (using candidate buckets for nodes). The information loss is also considered in the same way as in Ref.10. The datasets used for testing by Hay in Refs. 10 and 11 are 'Enron', 'Hep-th' (also referred to as 'co-authors' in other papers) and 'Net' (IP address traffic log).

Finally, Zhou[12] considers that an attack is on a "neighborhood", in which the adversary uses only information about neighbors and node degrees of neighbors in order to re-identify a given vertex.

In conclusion to this section on attack methods, we can say that for the authors we have surveyed, attack approaches which use auxiliary information are more common than attack approaches which try to 'crack' simply anonymized graph datasets. However, from a statisticians point of view, we are interested in making simply anonymized graph datasets publicly available for the 'bona-fide' analysis community, and therefore we need to protect from the non 'bona-fide' analysts (attackers) whose intention is to de-anonymize.

2.2 *Anonymization methods*

Almost all the anonymization methods we have found in the literature use k-anonymity (or something similar) as a basis. Examples of k-anonymity based methods include those of Liu[9], Bhagat[13], Zhou[12], Hay[10,11], Feder[15] and Wang[16]. One approach we found that could be considered somewhat different from k-anonymity is that of Dwork[8], which defines a privacy measure which is denominated as 'e-indistinguishability'.

The objective of defense methods which use k-anonymity is to produce k-degree anonymized degree sequences. That is, there must exist k-1 other nodes in the graph with the same degree as the target node. Liu[9] achieves this by using an algorithm which has two steps: (a) degree anonymization; (b) graph construction, which builds a new synthetic graph using greedy swap and priority. It makes use of the "small world" concept, that is, most nodes are not neighbors of one another but can be reached from every other node by a small number of hops. A key aspect is the use of degree distribution / sequence information. The potential attacker is not detailed with the exception of the assumption that he has a priori knowledge of the degrees of certain nodes. The information loss measure is calculated using typical graph statistics: (i) the norm of vector of differences between the degree sequences of new graph and the original graph; (ii) the average length of the shortest path between all pairs of reachable nodes; (iii) edge intersection, the percentage of edges in a degree anonymous graph which are also in the original graph; (iv) clustering coefficient, the average fraction of pairs of neighbors of a node that are also connected to each other. Liu[9] uses synthetic data and real data. The synthetic data consists of random graphs, small, graphs and scale free (power law degree distribution) graphs. The real data consists of four datasets: 'prefuse', 'enron', 'powergrid' and 'co-authors'.

Bhagat[13] presents a defense method called *class-based anonymization*, which consists of grouping the users into classes, and masking the mapping between the users and the nodes that represent them in the anonymized graph. This method is then made more robust to attack (but less useful for analysis) by incorporating a partitioning scheme similar to Hay's generalization method[11]. The class-based anonymization approach, creates a *'label list'* of candidate node ids for each vertex in the graph, one of which is the real id of the corresponding node. The attacker does not know which of the candidate node ids is the real one. Bhagat's method[13] differs from other approaches in the literature in that the graph includes 'rich data', that is, the users' personal details such as location, age, gender, game subscriptions, hobbies, ...). The information loss (utility) is calculated using the following statistics: pair queries (single hops in the graph); trio queries (two hops); triangle queries (clustering coefficient).

Zhou, in Ref. 12, presents a sophisticated anonymization algorithm which firstly generalizes vertex labels and secondly adds edges. The generalization of vertex labels is understood as the grouping together of labels based on some taxonomy. One of the precepts of the approach is to create local topologies which are isomorphic with other local topologies, achieved by adding edges to them. K-anonymization is used for

generalization, which makes use of two assumptions: (i) vertex degree follows a power law distribution and (ii) a "small world phenomenon" exists (described in terms of diameter and number of hops). Each vertex $v \in V(G)$ is grouped with at least (k-1) other vertices such that their anonymized neighborhoods are isomorphic (that is, of the same shape, or can be made to correspond). The method is edge preserving and uses a greedy search method to calculate the cost of anonymization and selects the lowest cost. The algorithm selects 'seed vertices' which have the largest neighborhoods.

The overall information loss (quality) measure of Zhou[12] is calculated by a formula which unifies different 'sub-measures', based on (i) vertex generalization and (ii) adding edges. For vertex generalization, it is calculated as the percentage of leaf nodes in the original graph which are converted to non-leaf nodes in the new graph. For edge insertion, it is calculated as the total number of edges added. The final component of information loss considers the number of vertices in a new neighborhood which were not linked in the original neighborhood. Zhou[12] uses real datasets ('KDD Cup 2003', 'arxiv' (.org) and the 'co-authors' (high energy physics) dataset).

A method which simply adds and deletes links (edges), is that of Hay[10], which presents a simple graph anonymization technique which selects edges randomly for modification. The information loss measure calculates some common graph metrics (clustering coefficient, path length distribution, degree distribution, ...) in the graph before and after anonymization. The information loss is considered from the point of view of an analyst who consults these statistical properties.

In Ref. 11, Hay extends the work of Ref. 10 to define a more sophisticated generalization method based on k-partitioning. The anonymization method generalizes the graph into k-partitions (groups nodes and edges into partitions) and publishes the anonymized (generalized) graph. The anonymized graph consists of a set of nodes (one for each partition) and a set of super-edges, which indicate the density of edges (in the original graph) between the partitions they connect. The new graph is optimized to fit the original graph with respect to a set of graph properties, by a maximum likelihood method using simulated annealing to search the state space

The problem of anonymizing the underlying graph of interactions in a social network is studied in Feder[15].The authors consider different combinatorial problems which can arise from the notion of (k,l)-anonymity in graphs. That is, for every node in the graph there exist at least 'k' other nodes that share at least 'l' of its neighbors. Given this definition, one example of a combinatorial problem would be: given an input graph how can the minimum number of edges to be added be calculated in order to achieve (k,l)-anonymity of the graph. The authors consider different solutions for this minimization problem, and find that for certain values of 'k' and 'l' the problems are solvable in polynomial-time, whereas for others they are NP-hard.

Another approach is that of 'granular computing', which is really referring to something similar to aggregation or generalization. Wang[16] considers privacy protection in social network data based on 'granular computing', which consists of grouping individuals with the same combination of attribute values into a common pool, or

'information granule'. The authors generalize the techniques for protecting personal privacy in tabulated data, and propose some metrics for anonymity. A description logic is used as a knowledge representation formalism, and the anonymity metrics are considered in open world and closed world contexts. In the former case, the graph may be incomplete, whereas in the latter, the graph is assumed to be complete.

The only method we found which is somewhat different from k-anonymity, is that of Dwork[8]. This approach considers perturbation as a sort of 'noise' introduced into the dataset, giving rise to an 'e-distinguishability' measure. In Ref. 8, Dwork considers the calibration of noise (as an anonymization method) using a 'sensitivity' function which depends on the input data. The paper focuses on the processing of tabular databases, however a section is devoted to consider the introduction of noise in graphs, by the adding and removing of links. It is considered theoretically, how much noise has to be added to a graph for the graph to become disconnected, non-expansive and/or poorly clustered. Different categories of 'sensitivity' are considered, for example, '1-sensitive'. In a graph context, an example of a '1-sensitive' function would be the distance of a given graph from the nearest disconnected graph.

In conclusion to this section on anonymization approaches, we observe that generalization is a common approach (concept/class/label generalization). In terms of the nodes and edges, this implies that an aggregation operator must be applied to the graph. Adding edges (Hay[10], Zhou[12]) is also an approach which is employed to obtain anonymity. However, some key aspects are still only approximations: (i) how do we optimally choose nodes for modification in an Online Social Network context in order to minimize information loss? (ii) which operators (which act on nodes and edges) are best to minimize information loss and achieve anonymity?

2.3 *Other observations and results*

Other interesting works in the social network privacy field are: Klienberg[17,18] presents an exposition of graph mining and privacy issues and state of art; Kossinets[19] and Kumar[20] consider how the evolution (growth) of social net works can be modeled and empirically measured; Xu[21] considers a utility based anonymization method based on local recoding and with an attribute-based utility measure; Felt[22] considers privacy protection for social network software from a software engineering point of view; Danezis[23] considers how to detect sybil nodes in *online social networks* with a practical approach and empirical evaluation; finally, Bonneau[24], considers different methods/data sources an attacker can make use of inside Facebook: (i) public listings; (ii) false profiles (sybil nodes); (iii) profile compromise and phishing; (iv) malicious applications; (v) use of the Facebook query language (similar to SQL) to collect information about URL ids.

As well as the definition of k-anonymity method by Sweeney[5]; the graph privacy community often cites some works not specifically on graph privacy, which are worth mentioning here: Mislove[14] which defines the key metrics of a social network; Liben-Newell[25], which defines a method for predicting link creation in social networks.

3. Proposed graph operators and how they modify graph elements and local topology

In this section we present the main considerations of the current paper: (a) offering a comprehensive set of operators for graph modification; and (b) modifying a local node-edge topology and evaluating the result, based on graph statistics.

When modifying a graph for anonymization purposes, all of the references we have encountered in the state of the art make one of three modifications to a graph: (i) add edge; (ii) delete edge; (iii) aggregate nodes (such as in k-anonymity). Thus, there is no method which explicitly considers adding and/or deleting individual nodes. Also we find a lack of consideration of node disaggregation (dividing a node n_1 in two or more nodes n_2, n_3 , ... n_n), understood as the inverse process of node aggregation. In general terms, we could consider a set of 6 operations, which are the following:

(i) Node addition
(ii) Node deletion
(iii) Edge addition
(iv) Edge deletion
(v) Node aggregation
(vi) Node disaggregation

It can been seen that we have not considered 'edge aggregation' or 'edge disaggregation'. These operators could be defined as 'higher level' operators which involve several of the six operators we have already defined. However, it is our opinion that the complexity of applying aggregation and disaggregation of edges, and the possible loss of context of the nodes themselves, made them less viable for our present study.

In the following sections, for the sake of brevity, we will only show diagrammatic representations for (v) node aggregation and (vi) node disaggregation.

Another relevant issue is that node aggregation methods found in the state of the art[11,12] apply a k-partitioning in which nodes are chosen for their ability to become isomorphic to other nodes, in terms of the topologies of their respective immediate neighborhoods. Apart from the high computational cost of comparing nodes to find the most similar isomorphically, we also propose that this method is really a 'tabular data' approach to anonymization, applied to non-tabular data. As a consequence, this method loses the vision of the whole graph topology structure. For example, consider two neighborhoods: (i) a lecturer in a Chinese university connected locally to his students and some colleague teachers (immediate neighborhood); (ii) a lecturer in a Spanish university connected locally to his students and some colleague teachers (immediate neighborhood). Applying the isomorphism similarity criteria, these two neighborhoods may be fairly similar and therefore the algorithm chooses them to be modified. However, as a consequence these two nodes lose some of their relation with respect to the whole graph. As they are geographically remote and from different cultures, we would expect the

surrounding wider neighborhood (for example, hops between 2 and 4) to be distinct. A local pairing method would only aggregate physically adjacent node pairs, where the node pair would be chosen with a probability weighted by average graph statistics. For example, in the case of the Chinese lecturers, it would be quite probable that we may aggregate two students of the Chinese lecturer, if they were also directly linked together, given that the statistical properties of two students (of the Chinese lecturer) would probably be similar. However, it would not be so probable (although possible) that we modify the local topology of the Chinese lecturer with respect to the local topology of the Spanish lecturer, something which may happen in Zhou's approach[12]. Once an aggregation operation is realized, we also have to re-link the respective neighbors, taking into account the average statistical properties of the nodes in the graph.

In Fig. 2 we illustrate what we have just described, where the selected nodes and their immediate neighborhoods represent the Chinese and the Spanish lecturers. Zhou's method[12] would modify the two selected topologies to obtain isomorphisms, whereas local aggregation would carry out separate local modifications to reduce risk of re-identification. With respect to local topology modifications, it could be said that a local aggregation method is focused on minimizing information loss whereas Zhou's method is focused on minimizing risk of disclosure. However, a general algorithm could be devised (for example, using simulated annealing) which is driven equally in the fitness function by both information loss and risk of disclosure.

Thus, in the remainder of this Section we will consider how to modify the graph structure:

(a) Operators used to modify the graph and their functionality.

(b) Statistical measures for representing the graph and for calculating the information loss in terms of the average degree, the clustering coefficient and the shortest path length.

The aspect of incorporating network activity into the calculations is discussed in Section 3.2, although we do not enter into more detail, given that our current focus are topologically based statistics. The incorporation of network activity as a weighting factor on the topological statistics will be considered in detail in future work.

In the following sections, we will use simple examples to illustrate the consequences of applying the operators to a graph topology. This will give us insight into how to quantify the information loss (by typical graph metrics such as degree, clustering coefficient and shortest paths).

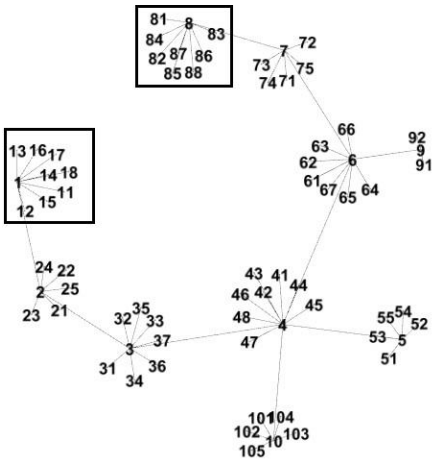


Fig. 2. Topology showing two distant nodes (in terms of path length) chosen for aggregation based on isomorphism

3.1 *Evaluating the cost (information loss) of graph modification.*

The following is a simple example which serves to clarify the procedure of calculating the information loss. However it does not generalize how graph properties are affected by the different modification operators. This would require empirical evaluation with real datasets, which is proposed in the conclusions as a next step.

In this section, for each of the six graph modifier operators (mentioned at the beginning of Section 3), we will calculate three standard metrics from graph theory to represent the characteristics of the graph:

(a) *Average degree of the nodes in the network* . The number of links emanating from a given node, to other nodes (initially we will consider all links as non-directional, as in undirected graphs).

(b) *Average clustering coefficient of the graph* . The average for all users (nodes), of the number of friends of a user who are also mutual friends with respect to the total possible number of links between those friends:

CC = Number of links between friends of user i /
Number of possible links between friends of user i

For example, if user 1 has N friends, and of those N friends, there are M mutual links between them, independent of the link with user 1, then the CC for this 'cluster' will be M / N (N-1). Thus, if N=3 and M=4, then CC = 4 / (3 * (3-1)) = 4 / 6 = 0.66. For the New Orleans Facebook dataset[4] a global CC value of 0.15 was reported.

(c) *Average shortest path of the graph.* For each node we find the shortest number of hops to reach every other node, then we calculate the average for the whole graph. The average normalized shortest path for the complete original graph is:

$$sp = \sum_{1}^{n} \sum_{1}^{m} sp(u_m, u_n) \quad / \quad (n \times m\text{-}1) \tag{1}$$

where $sp(u_m, u_n)$ is the shortest path from node u_m to node u_n, n being the number of nodes in the graph and m the average number of mutually related nodes.

For statistics (a) to (c), first we calculate the metrics for the original graph, then we recalculate for the graph resulting from the modification by each of the six operators. This enables us to make an initial comparison of the effect of each operator on a local topology and make some comparisons between them.

3.1.1 Computational cost of calculating the statistics that characterize the graph

In this Section we will consider three aspects: **(i)** the cost of calculating the statistics for a complete graph, which may be the original graph, or a proposed new topology; **(ii)** the cost of calculating the statistics for a neighborhood; **(iii)** mechanisms for reducing the computation cost.

(i) Cost of calculating the statistics for the whole graph. We define m_{CS} as the cost of calculating the statistics that characterize the graph (see Tables 2 and 3): average degree, average clustering coefficient and average shortest path for the graph.

- Average degree: (for n nodes × m average number of related nodes)
- Clustering coefficient: (for n nodes × m average number of related nodes × q average number of links between nodes of m)
- Average shortest path: (for number of possible node pairs np × average number of hops between node pairs *anh* × *average number of possible routes between each pair anpr*)

This gives:

Average degree: ($n \times m$) +
Clustering coefficient: ($n \times m \times q$) +
Average shortest path: ($np \times anh \times anpr$)

$$\begin{aligned} m_{CS} &= (n \times adn) + (n \times m \times q) + (np \times anh \times anpr) \\ &= (n \times adn) + (n \times m \times q) + (n \times n \times anh\ x\ anpr) \end{aligned} \tag{2}$$

(ii) Cost of calculating the statistics that characterize a neighborhood. A neighborhood refers to the nodes which are directly linked to a given node (hops = 1), which is to be the objective of some modification by one of the operators. The statistics are calculated a maximum of $\sqrt{(nn)}$ times, where *nn* is the number of nodes in the neighborhood, which represents the number of possible linking permutations tested in order to find a semi-

optimum configuration for a neighborhood, when we add, delete or modify a given graph element or elements in that neighborhood. The same statistics (degree, clustering coefficient and path length) are calculated as for the whole graph, the only difference being that in the computational cost calculation we substitute n by $\sqrt{nn}$, where n is the number of nodes in the whole graph. However, we only calculate the statistics average degree and path length for the nodes in the immediate neighborhood (hops = 1), and the clustering coefficient necessarily for neighbors of neighbors (hops = 2).

(iii) Mechanisms to reduce computation cost. The computation cost is highest if we visit the graph in a 'linked list' crawling manner. However, we can clearly make very significant savings if we pre-calculate the statistics (degree, clustering coefficient and average path length) for each node and place them in a table which allows direct access using the node id as a hash-key, for example. The tables would be updated after any modifications to the graph, by efficiently propagation. Other methods for efficiently representing and processing large graphs, such as compact bitmap representations of the topology (nodes and edges), can also be made use of.

Examples: for illustration, we consider the graph in Fig. 3. We observe that the graph is undirected, given that even if it is user u_1 who establishes (invites) the establishment of a link with users u_2, u_4 and u_7 (who then accept or not the invitation), once established the link is bidirectional in terms of information flow, and there is no defined hierarchy in the relation.

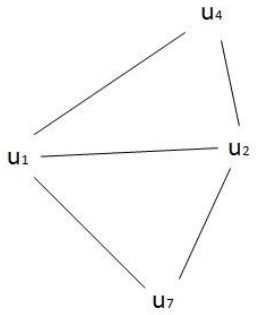


Fig. 3. Graph representation of the relation between four users of an online social network

Now we will calculate the metrics for this graph: average degree, average clustering coefficient and average shortest path. Average degree (*ad*) and average clustering coefficient (*cc*) are given in Table 2a, whereas average shortest path (*sp*) is given in Table 2b.

Table 2a. Degree and clustering coefficient metrics for original graph.

Node	***Degree*** *(normalized value in parenthesis)*	***Clustering coefficient***
u1	3 *(1.0)*	4 / 6 = 0.66rec
u2	3 *(1.0)*	4 / 6 = 0.66rec
u4	2 *(0.66rec)*	2 / 2 = 1.0
u7	2 *(0.66rec)*	2 / 2 = 1.0
Average	*ad* = 3.33rec / 4 = *0.833rec*	*cc* = 3.33rec / 4 = *0.833rec*

Table 2b. Shortest path metrics for original graph (normalized values in parenthesis).

	u1	***u2***	***u4***	***u7***	*Average normalized value for node*
u1	0	1 *(0.5)*	1 (0.5)	1 (0.5)	*0.5*
u2	1 (0.5)	0	1 (0.5)	1 (0.5)	*0.5*
u4	1 (0.5)	1 (0.5)	0	2 (1.0)	*0.66rec*
u7	1 (0.5)	1 (0.5)	2 (1.0)	0	*0.66rec*

The average shortest path for the original graph, computed from the shortest path given in Table 2b, is:

sp = ((0.5+0.5+0.5) + (0.5+0.5+0.5)+(0.5+0.5+1.0)+(0.5+0.5+1.0)) / 12 = (7 / 12) = 0.5833rec.

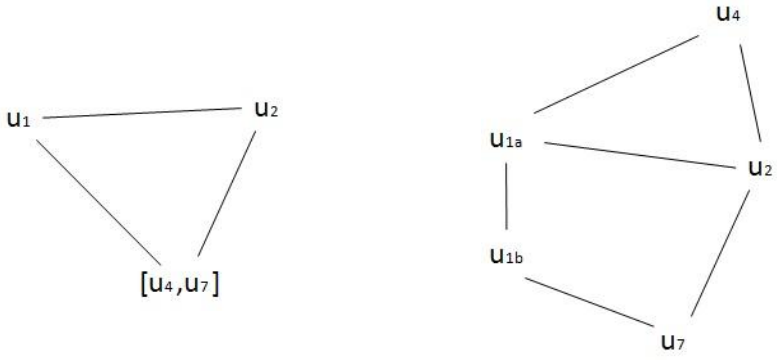
Therefore we now have the three normalized descriptive metrics for the original graph: ad = 0.833rec , cc = 0.833rec and sp = 0.5833rec.

We now consider the application of the six possible modifications to the graph in Fig. 3, and the re-calculation of the corresponding metrics following each modification.

- *Add node u_{10}. We assume that a new node u_{10} is added. We now recalculate the metrics for the new graph, which gives: ad = 4 ÷ 5 = 0.8 , cc = 2.66rec ÷ 5 = 0.53rec and sp = 10 ÷ 20 = 0.5.*
- *Delete node u_4. We assume that node u_4 is chosen for deletion. Once deleted, we recalculate the metrics, which gives: ad = 3 ÷ 3 = 1.0 , cc = 3 ÷ 3 = 1.0 and sp = 6 ÷ 6 = 1.0.*
- *Add edge {u_7 , u_4}. We assume that a new edge {u_7 , u_4}. is added. Once added, we recalculate the metrics, which gives: ad = 4 ÷ 4 = 1.0 , cc = 4 ÷ 4 = 1.0 and sp = 12 ÷ 12= 1.0.*
- *Delete edge {u_1 , u_4}. We assume that edge {u_1 , u_4} is chosen for deletion. Once deleted, we recalculate the metrics, which gives: ad = 2.66rec ÷ 4 = 0.66rec , cc = 2.33 ÷ 4 = 0.58 and sp = 8 ÷ 12 = 0.66rec.*
- *Aggregate nodes u_7 , u_4 → [u_7 , u_4]. We assume that nodes u_7 and u_4 are*

aggregated to form a new node [u_7 , u_4] (see Fig.4a). We now recalculate the metrics, which gives: ad = 3 ÷ 3 = 1.0 , cc = 3 ÷ 3 = 1.0 and sp = 6 ÷ 6= 1.0.

- *Disaggregate node u1 → u1a , u1b . We assume that node u1 is disaggregated into two new nodes u1a and u1b (see Fig. 4b). We now recalculate the metrics, which gives: ad = 4 ÷ 5 = 0.8 , cc = 2.4 ÷ 5 = 0.48 and sp = 14 ÷ 20= 0.7.*



(a) (b)

Fig. 4. Original graph (Fig. 3) altered by (a) aggregation of nodes u_4 and u_7 and (b) disaggregation of nodes u_1 into nodes u_{1a} and u_{1b}

We now present the summary of metrics for the six graph operations (Table 3a) and the distances from original graph (Table 3b).

With reference to Table 3b, we observe that the operations with the smallest normalized distance from the original graph (interpretable as 'cost') for metric 'ad' are 'add node' and 'disaggregate node', both with 0.03. In the case of metric 'cc', the operations 'delete node', 'add edge' and 'aggregate nodes' have the smallest cost, all with 0.17. For metric 'sp', the operations 'delete edge' and 'add node' have the smallest cost, with 0.077 and 0.083, respectively. Finally, if we add and average the normalized distances of the three metrics (last column of Table 3b), we get an overall distance for each operation. We see that 'add node' gives the lowest distance of 0.413, followed by 'delete edge' and 'disaggregate node', both with 0.497.

Thus, an initial conclusion in terms of the graph topology alone, would be that the 'cheapest' action is 'add node'. However, we have tried only one possible permutation for each operation, and we would need to try all possible permutations in order to choose the 'cheapest' for each operation, before comparing operations for real data.

Table 3a. Summary of metrics for six graph modifications.

modification	***ad - average degree***	***cc - average clustering coefficient****	***sp = average shortest path****
original graph	0.83	0.83	0.58
add node	0.80	0.53	0.50
delete node	1.00	1.00	1.00
delete edge	0.66	0.58	0.67
add edge	1.00	1.00	1.00
aggregate nodes	1.00	1.00	1.00
disaggregate node	0.80	0.48	0.70

**rounded to 2 decimal points for the sake of clarity*

Table 3b. Distance between new graph and original graph based on the metrics and for each type of modification.

operation	***ad - average degree***	***cc - average clustering coefficient***	***sp = average shortest path***	***Overall (ad + cc + sp)***
add node	***0.03****	0.30	***0.083***	***0.413***
delete node	0.17	***0.17***	0.417	0.757
delete edge	0.17	0.25	***0.077***	***0.497***
add edge	0.17	***0.17***	0.417	0.757
aggregate nodes	0.17	***0.17***	0.417	0.757
disaggregate node	***0.03***	0.35	0.117	***0.497***

**minimum distances are indicated in bold italic*

3.2 Evaluating the cost (information loss) of graph modification, incorporating inter-node activity as a weight.

In order to incorporate information about activity into the information loss calculation, the calculations of Section 3.1 would be repeated, but incorporating inter-node activity as an edge/node weight into the cost value, together with the distance metrics. As a consequence, edges and nodes with lower activity (below a given limit) would have a greater probability of being candidates to be altered by one of the graph operators.

In the case of the *Enron email corpus*[1,2,3], the activity measure is simply the number of emails sent/received. This measure is quite a strong indicator of contact between individuals, although of course people could also communicate via other means. For example, two persons A and B could send just a few key emails, and then communicate in more detail by telephone. The Enron email corpus also only records email in which the sender or receivers' email is a corporate Enron email (enron.com). Thus if two employees of Enron also communicate via their personal 'gmail' account, for example, this activity would not be registered in the Enron email corpus.

In the case of the *Facebook* dataset[4], we would use as the activity measure the 'writes to wall' (posts received and sent) over a given time period, between users: *activity* = (PR + PS) ÷ (max_{PR} + max_{PS}), where PR are posts received and PS are posts sent. As commented in the introduction, this measure has its aspects in favor and its inconveniences.

The activity measure between two users (nodes), $w(u_1, u_2)$ considers both 'posts received' and 'posts sent', and this value is assigned to edge $\{u_1, u_2\}$. To calculate the weight w_u for a node u, we simply sum the weights of all the edges $\sum w\{u, u_n\}$ connected to node u.

4. Summary

In this paper we have presented a vision of the state of the art for graph privacy and processing, and we have identified some research areas which have not been covered, such as the use of a comprehensive set of operators to modify the graph, and using local pairing for node aggregation. This has lead us to define six operators and how each of them would operate on a simple local topology of nodes. We have also considered key aspects of graph modification: node selection, computational cost and neighborhood re-linking. In terms of the attacker, we have assumed one who is statistically knowledgeable and able to map statistical properties (degree, clustering coefficient, path length) onto a given node and its immediate neighborhood topology. As next steps we propose to apply the method to the New Orleans Facebook dataset[4] and the Enron emails dataset[1,2,3], and benchmark against the k-partitioning method which uses isomorphic similarity for node aggregation. We will also incorporate activity information into the information loss calculation as a weighting factor on the topological statistics.

Acknowledgements

This research is partially supported by the Spanish MEC (projects ARES CONSOLIDER INGENIO 2010 CSD2007-00004 -- eAEGIS TSI2007-65406-C03-02). The authors would like to thank the anonymous reviewers for their suggestions and comments.

References

1. W. Cohen, Enron Email Dataset. "http://www.cs.cmu.edu/~enron/"
2. B. Klimt and Y. Yang, Introducing the Enron Corpus. *Proc. 1st Conf. on Email and Anti-Spam (CEAS)*, (Mountain View, CA, 2004).
3. J. Shetty and J. Adibi, Discovering Important Nodes through Graph Entropy - The Case of Enron Email Database. *KDD '2005* (Chicago, Illinois).
4. B. Viswanath, A. Mislove, M. Cha and K.P. Gummadi, On the Evolution of User Interaction in Facebook. In *Proc. 2nd ACM workshop on Online social networks (WOSN)*, (August 17,2009, Barcelona, Spain). "http://socialnetworks.mpi-sws.org/"
5. L. Sweeney, k-anonymity: a model for protecting privacy, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS).* Vol. 10, Issue: 5(2002) pp. 557-570.

6. A. Narayanan and V. Shmatikov, De-anonymizing Social Networks. *Proc. of the 2009 30th IEEE Symposium on Security and Privacy*, pages: 173-187 (2009).
7. L. Backstrom, C. Dwork and J. Kleinberg, Wherefore Art Thou R3579X? Anonymized Social Networks, Hidden Patterns, and Structural Steganography. *Proc. 16th Intl. World Wide Web Conference, 2007 (SESSION: Mining in social networks)*, pp. 181 - 190 .
8. C. Dwork, F. McSherry, K. Nissim and A. Smith, Calibrating Noise to Sensitivity in Private Data Analysis, *Proc. Theory of Cryptography Conf. (TCC)*, pp. 265-284, 2006.
9. K. Liu and E. Terzi, Towards Identity Anonymization on Graphs, *SIGMOD 2008*.
10. M. Hay, G. Miklau, D. Jensen, P. Weis and S. Srivastava, Anonymizing Social Networks. *SCIENCE Technical Report 07-19 (2007)* pp. 107--3, Vol. 245 (Computer Science Department, University of Massachusetts Amherst).
11. M. Hay, G. Miklau, D. Jensen, D. Towsley and P. Weis, Resisting structural re-identification in anonymized social networks. *Proc. of the VLDB Endowment (SESSION: Privacy and authentication)* Vol. 1 , Issue 1, pages 102-114, (August 2008).
12. B. Zhou and J. Pei, Preserving Privacy in Social Networks against Neighborhood Attacks. IEEE 24th International Conference on Data Engineering (ICDE), 2008.pp. 506 - 515.
13. S. Bhagat, G. Cormode, B. Krishnamurthy and D. Srivastava, Class-based graph anonymization for social network data. *Proc. of the VLDB Endowment (SESSION: Social networks and recommendations)* Vol. 2 , Issue 1, pages: 766-777, (August 2009).
14. A. Mislove, M. Marcon and K.P. Gummadi, Measurement and Analysis of Online Social Networks. *Proc. of the 7th ACM SIGCOMM Conf. on Internet Measurement. SESSION: Social networks*, pp. 29 - 42, 2007.
15. T. Feder T. Feder, S. U. Nabar and E. Terzi, Anonymizing Graphs, *CoRR abs/0810.5578*, October, 2008.
16. D.W. Wang, C.J. Liau and T. Sheng Hsu, Privacy Protection in Social Network Data Disclosure Based on Granular Computing, In *Proc. 2006 IEEE International Conference on Fuzzy Systems, IEEE Computer Society*, pp. 997-1003.
17. J. Kleinberg, Challenges in Mining Social Network Data. *Proc. of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '07)*, pages: 4 - 5 (2007).
18. J. Kleinberg, L. Backstrom, C. Dwork and D. Liben-Nowell. Algorithmic Perspectives on Large-Scale Social Network Data. *Data-Intensive Computing Symposium* (March 26, 2008 - Hosted by Yahoo! and the CCC). "research.yahoo.com/files/7KleinbergSocialNetwork.pdf"
19. G. Kossinets and D.J. Watts, Empirical analysis of an evolving social network. *Science 6* (January 2006), Vol. 311. no. 5757, pp. 88 - 90.
20. R. Kumar, J. Novak and A. Tomkins, Structure and Evolution of online social networks. *Proc. Link Mining: Models, Algorithms, and Applications 2010*, Part 4, 337-357.
21. J. Xu, W. Wang, J. Pei, X. Wang, B. Shi and Wai-Chee Fu. Utility based anonymization using local recoding. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 785 - 790, 2006.
22. A. Felt and D. Evans, Privacy protection for social networking APIs. *In Web 2.0 Security and Privacy (W2SP'08)*, 2008.
23. G. Danezis and P. Mittal. Sybilinfer: Detecting sybil nodes using social networks. *In NDSS. The Internet Society*, 2009.
24. J. Bonneau, J. Anderson and G. Danezis, Prying data out of a social network. *In ASONAM 2009: The 2009 International Conference on Social Network Analysis and Mining*, (Athens, Greece, July 20, 2009).
25. D. Liben-Nowell and J. Kleinberg, The Link Prediction problem for Social Networks. *Journal of the American Society for Information Science and Technology*, Volume 58, Issue 7, pages 1019–1031, May 2007.