Data Privacy in Online Social Networks

Evaggelia Pitoura[1] Ling Liu[2]

Abstract

The ever increasing volume of available social network datasets creates the need to derive anonymization methods that permit useful analysis of such datasets without disclosing any sensitive information regarding their users. In this article, we review the problem of privacy in online social networks focusing on related models and algorithms.

1 Introduction

Recent years have witnessed a tremendous increase of the popularity of online social networking (OSN) sites. More and more people join various social networks on the web, (such as Facebook, Linkedin, or Twitter) to communicate and share information with their friends.

Social networks are instances of a general type of datasets, termed *network* or *graph* datasets. Network datasets appear in a variety of domains. For example, network datasets include *information networks* of articles connected by citations or co-authorship, *communication networks* of Internet hosts related by traffic flows, mobile-phone and email users connected based on the messages they have interchanged, and *epidemics networks* describing for example, the transmission of infectious diseases among individuals. The actors and the social relationships among them in social networks (and graph datasets in general) can be modeled using graph models whose vertices correspond to actors and edges to relationships.

[1]Computer Science Department, University of Ioannina, GR45110, Ioannina, Greece, pitoura@cs.uoi.gr

[2]College of Computing, Georgia Institute of Technology, Atlanta, GA 30332, lingliu@cc.gatech.edu

The increasing volume of available network datasets creates both the need to protect the privacy of the individuals involved in them, but also the potential to extend their analysis towards improving our understanding of their structure and the complex patterns involved in them. Social networks are analyzed for example to study disease transmission, to detect spam, to measure the influence of a publication, to detect trends in user behavior, or to evaluate the resiliency of a network to attacks.

In this article, we consider the problem of publishing social network data to allow useful analysis without disclosing sensitive information. The problem can be formalized as follows.

In a nutshell, given a network G construct an anonymized network G^* in which private information is hidden. There are three different entities involved:

1. the *users* of the social network whose private data needs to be protected,
2. the *adversary* or *attacker* that attempts to combine G^* with any external information that she owns or can attain to deduce private information,
3. the benign *analyst* who wants to use G^* to extract useful information.

Then, the problem is how to construct G^* so that both (i) the private data of each user are protected under all possible forms of external knowledge of the attacker and (ii) the utility of G^* for the analyst remains high. The problem and its solutions take various forms depending on what type of information is considered private, the extent and form of the background knowledge of the attacker, and the privacy and utility criteria that the released network needs to satisfy.

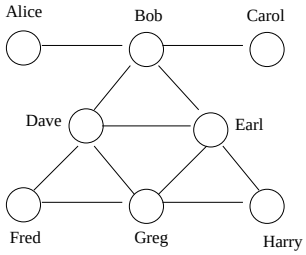
In the remainder of this article, we focus in this privacy problem and its variations. In Section 2, we present the formal model of the problem and in Section 3 related algorithms. In Section 4, we outline other issues besides data anonymization. Section 5 concludes the paper.

2 The OSN Privacy Model

In this section, we introduce the basic abstractions used to formulate the privacy problem in OSNs. First, we present the graph model that is typically used to represent OSNs. Then, we focus on the model of privacy that involves three important

ID	Age	Sex	HIV
Alice	25	Female	Pos
Bob	30	Male	Neg
Carol	30	Female	Neg
Dave	19	Male	Pos
Earl	45	Male	Neg
Fred	23	Male	Neg
Gerg	45	Male	Pos
Harry	19	Male	Neg

(a)



(b)

Figure 1: (a) Example of tabular data, (b) example of social network data.

components (i) a concrete specification of what is considered private and needs to be anonymized, (ii) the type of external information that an attacker or adversary may posses, and (iii) a set of criteria to assess the degree of privacy attained (i.e, "how much" private are the data released) and the associated cost or loss in their utility for analysis.

Most existing work on privacy in data publishing has focused on relational data, where the data to be released are in tabular format with each record representing a separate entity. An example of tabular data is shown in Figure 1(a), while an example of OSN data is shown in Figure 1(b). However, anonymization techniques for tabular data do not apply to social network data, since these techniques do not take into account the interconnectedness of the entities (for example, the fact that Alice is connected with Bob). For a survey on privacy preservation for relational data, see for example [4].

2.1 Social Networks as Graphs

A social network and in general any network dataset is commonly modeled as a graph where vertices correspond to users or other entities participating in the network, such as for example, groups of users, or material shared among users (such as photos or documents). Links between users represent relationship between them (such as users being "friends" or "followers" of each other), interactions (such

as information exchange in the form of emails or messages), various dependencies, or commonalities among users (such as similar behavior or common interests).

Definition 1 (Basic OSN Model) *An OSN is modeled as an undirected graph* $G = (V, E)$ *where* V *is a set of vertices and* $E = V \times V$ *is a set of edges. The vertices* $v \in V$ *model individual social actors in the network, while the edges* $(v_1, v_2) \in E$ *represent a relationship between social actors* v_1 *and* v_2*.*

An example of social network data representing by a graph is shown in Figure 1(b). Vertices corresponds to individuals actors and edges among them represent relationships.

Extended graph models have been used to capture the cases in which the released data include additional information besides individuals and their connections. Such models include:

- edge weights or edge labels used to represent, for example, the importance or strength of connection between the two individuals, (e.g., the number of messages that they have exchanged, or the affinity of their relationship), the type of the connection (e.g., friends), or other attributes of the corresponding relationship that the edge models.

- directed graph models to express the fact that the relationship between two individuals is not symmetric (i.e, that fact that v_1 is connected with v_2 does not necessarily means that v_2 is also connected with v_1) as is the case in some social networks, e.g., "follow" in Twitter,

- attributes or labels associated with vertices, for example, in Figure 1, each vertex instead of being labeled with just the ID of the corresponding actor, it could have been labeled with additional attributes of the actor (such as her genre or age).

- the domain of the labels associated with vertices and edges may take values from hierarchical domains; in such cases, the value of the labels may differ in the level of the provided detail and thus in the level of privacy disclosed (e.g., age may be described using general categories such as young, or middle-aged).

Besides general graphs, specific types of graph, such as k-partite graphs, have also been used to represent OSNs. A *k-partite graph* consists of k disjoint sets of vertices, with no edges between vertices of the same set. Such graphs are used to model social networks that involve entities of different types. For example, users at a site such as deli.cio.us, place tags on Web pages, thus creating social networks that involve three distinct types of entities: users, tags, and pages. This naturally leads to a 3-partite graph with the corresponding three set of entities. In this graph, the only edges are among entities of the same type. Users are related to each other, if they use the same tags frequently, or if they tag the same pages. Similarly, tags are related to each other, if they appear on the same pages or are used by the same users and pages are related to each other, if they have many of the same tags or are tagged by many of the same users.

2.2 Privacy Model

But what do participants in an OSN consider as private information that needs to be protected?

This appears to be highly subjective. Some interesting facts regarding the patterns of personal information revelation and the privacy implications associated with OSNs were reported in a study involving 4000 users in the early days of Facebook [9]. The authors observed an apparent openness of the users to reveal personal information to a vast network of loosely defined acquaintances or complete strangers. Further, the relation between privacy and the social network of each user was multifaceted. In certain occasions, users wanted personal information to be known only by a small circle of close friends, and not by strangers, whereas in other instances, they were willing to reveal personal information to anonymous strangers but not to those that know them better. It needs also to be stressed that as opposed to real life, where friendships have degrees of closeness, in online networks, friendships tend to be binary. The authors also highlight potential privacy breaches and other dangers, such as identifying anonymized users using similar photos that the users have released in other networks, or combining released information such as age, address or occupation for stalking users, for identity theft, or even for deducing their social security numbers.

Another issue that complicates privacy management in OSNs is the multitude of the available information. In relational data, the information made available is

in the form of attribute-value pairs, thus privacy preservation is performed at the level of attributes. In particular, the goal is to protect the disclosure of the values of specific attributes associated with a user, called *sensitive* attributes. For example, some users may consider as sensitive attributes their political beliefs, others their age or the type of disease that they may suffer from. For example, in Figure 1(a), users may want to protect the value of their age or HIV attribute.

In OSNs, sensitive or private data privacy protection goes beyond protecting the values of single attributes. In general, sensitive information may be classified as referring to:

- *vertex existence:* whether a target individual appears in the network or not (for example, we may want to hide the appearance of individuals in a disease infection network),
- *identity disclosure:* besides just the existence of an individual, her correspondence to a specific vertex in the social network is revealed (for example, the fact that Alice is associated with the leftmost vertex in the graph of Figure 1(b))
- *vertex attribute disclosure:* besides the identity of an individual, other attributes that she considers private are also disclosed (for example, the fact that Alice is HIV positive)
- *link or edge disclosure:* the sensitive relationship between two individuals is disclosed, (for example, the fact that Alice and Bob are friends in the graph of Figure 1(b))
- *edge attribute disclosure:* sensitive attributes related with an edge are disclosed
- *content disclosure:* the sensitive data associated with each vertex or edge is compromised (for example, the actual content of the email messages exchanged between two individuals in an email communication network is disclosed)
- *property disclosure:* properties regarding the network structure around an individual are revealed, such as vertex degrees clustering coefficient or properties of the neighbors of a vertex (for example, implying that the specific individual is a community leader).

2.3 The Attacker

Central to protecting the privacy of an OSN is knowing the type of external knowledge that a malicious user, called attacker or adversary, has or can attain regarding the OSN.

In relational data, a common assumption is that the attacker knows the values of specific attributes, called *quasi identifiers*. Using the values of these attributes, such as say the age or the address of a user, an adversary may potentially identify an individual in a released table, even when the identity of the individual is masked. For example, it suffices for an attacker to know that Alice is 25 years old, to deduce from the tabular data in Figure 1(a) that Alice is HIV positive, even if her identity is hidden.

For OSNs, external knowledge may take many different forms. Specifically, besides values of specific attributes, the attacker may have knowledge of the structural properties of the network. A commonly made reasonable assumption is that the structural knowledge of the attacker is local, limited to a small radius or neighborhood around the targeted individual.

In general, external information may either

- consist of background knowledge of the adversary, or
- be acquired through specific malicious actions of the adversary, called *attacks*.

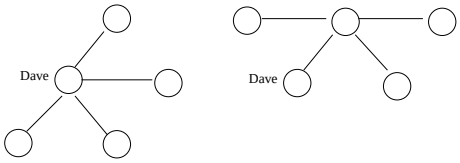
Background knowledge may exist due to the fact that the adversary is a participant in the network which gives her innate knowledge of the other participating entities and their relationships. Background knowledge may be also attained through public information sources.

External information sources may be accurate or not. Even when accurate, the provided information may not be necessarily complete. Thus, a distinction is made between:

- *a closed world adversary*, in which case, external information is complete and absent facts are considered false,
- *an open world adversary*, in which case, external information may be incomplete and absent facts are simply unknown

ID	$\mathcal{H}_0$	$\mathcal{H}_1$	$\mathcal{H}_2$
Alice	Female	1	{4}
Bob	Male	4	{1, 1, 4, 4 }
Carol	Female	1	{4}
Dave	Male	4	{2, 4, 4, 4, }
Earl	Male	4	{2, 4, 4, 4 }
Fred	Male	2	{4, 4}
Gerg	Male	4	{2, 2, 4, 4}
Harry	Male	2	{4, 4}

(a)



(b)

Figure 2: (a) The results of the vertex refinement queries on the graph of Figure 1(b), and (b) the results of subgraph queries with 4 edge facts on the graph of Figure 1(b) using breadth first search search (left) and depth first search (right)

The authors of [12] use a query-based model of attacks. An adversary is assumed to have access to a source that provides answers to a structural query Q evaluated for a single target vertex v of the original graph $G(V, E)$. Let $Q(v)$ be the provided answer. The candidate set for v with regards to Q is a set of vertices in the anonymized graph, $G_a(V_a, E_a)$ defined as: $cand_Q(v) = \{w \in V_a \mid Q(v) = Q(w)\}$.

Three types of queries are proposed. *Vertex refinement* queries is a class of queries of increasing power that report on the local structure of the graph around a vertex. The weakest knowledge query, query $\mathcal{H}_0(v)$, simply returns the label of v. Query $\mathcal{H}_1(v)$ returns the degree of v, while $\mathcal{H}_2(v)$ returns the multi-set of the degrees of each of the neighbors of v. Generalizing, $\mathcal{H}_i(v)$ returns the multi-set of values which are the result of evaluating $\mathcal{H}_{i-1}$ on the vertices adjacent to v, whereas $\mathcal{H}^*$ stands for the iterative computation of $\mathcal{H}$, until no new vertices are distinguished. An example is shown in Figure 2 (a) using the graph of Figure 1(b) and assuming that vertices are labeled with the sex attribute.

Subgraph queries are a class of queries about the existence of a subgraph around the target vertex. Their descriptive power is measured by the number of edges in the subgraph, termed *edge facts*. Different subgraphs correspond to different strategies of knowledge acquisition by the adversary, including breadth-first and depth-first exploration. For a given number of edge facts, some queries are

more effective at distinguishing individuals. Figure 2(b) depicts the result of two subgraph queries with three edge facts around the individual $v = Dave$. As opposed to vertex refinement queries that are closed world, subgraph queries are open world. Finally, *hub fingerprint queries* assume the existence of hubs. A hub is a vertex with a high degree and high betweenness centrality (i.e., the proportion of shortest paths in the network that include the vertex). A hub fingerprint query, $F_i(v)$ for a target vertex v, is a description of the connections of v to a set of designated hubs in the network, where i is a limit on the maximum distance. Hub fingerprint queries may be open-world or closed world. For example consider that $Dave$ and $Earl$ in Figure 1(b) are hubs, then $F_1(Fred)$ = (1; 0) - meaning that *Fred* is connected with $Dave$ but not with $Earl$ within distance 1 - and similarly, $F_2(Fred)$ = (1;1).

Since users often participate in more than one social network, the authors of [17] consider a form of attack that is based on knowing the structure of another *auxiliary social network* with overlapping membership with the targer network. Given a set of seeds vertices with known mappings between the auxiliary and the target OSNs, the attacker attempts to learn private information about other members of the target network. The authors applied their de-anonymization algorithm using Twitter as the target network and Flickr as the auxiliary network. They show that a third of the users who are verifiable members of both Flickr and Twitter can be recognized in the completely anonymous Twitter graph with only 12% error rate, even though the overlap in the relationships for these members is less than 15%.

Particular to social networks is the fact that the attacker may be a user of the network herself, thus she may interfere and alter the network before its anonymized version is released. Consequently, targeted attacks are characterized as [1]:

- *active*, when the attacker attempts to compromise privacy by strategically creating new users and edges before the anonymized network is released, so that these new vertices and edges will be present in the released network, and

- *passive*, when the attacker tries to deduce private information only after the anonymized network has been released.

A specific active attack for attaining structural information of targeted individuals is described in [1]. Before the anonymized network is released, the attacker creates links with other cooperating users (or, creates new users and links with them),

so that a highly connected subgraph having a distinctive structure is formed. Then, she links this subgraph to the targeted individuals. After the anonymized network is released, the attacker searches to locate the injected subgraph in the network. From this subgraph, she can identify the targeted individuals and attain structural information about them. The authors show that given a network G with n vertices, it is possible to construct a pattern subgraph with $k = O(log(n))$ vertices that will be unique in G with high probability. Further, this subgraph can be efficiently found in the released network G^* and can be linked to as many as $O(log^2(n))$ target vertices.

In general, the success of an attack depends both on the descriptive power of the external information and on the structural diversity of the OSN itself.

2.4 Evaluation

An anonymized network is evaluated with regards to two criteria, namely on whether: (i) the private data of each user are protected and (ii) the utility of the graph is preserved.

2.4.1 Privacy Criterion

A privacy criterion characterizes how safe it is to release a specific instance of an OSN. In relational data, a criterion that is often used is *k-anonymity* [18, 20].

Definition 2 (k-anonymity) *Given a set of quasi-attributes $Q_1, \ldots, Q_d$, a released relation T^* is said to be k-anonymous with respect to $Q_1, \ldots, Q_d$, if each unique tuple in the projection of T^* on $Q_1, \ldots, Q_d$, occurs at least k times.*

This means that an individual is hidden among k others, in the following sense. An adversary that knows only the values of the d quasi attributes can only guess which of the k tuples correspond to a specific individual with probability $1/k$. For example, if we consider that the quasi attribute is Sex, and the sensitive attribute is HIV, then, the Table in Figure 1(a) is 2-anonymous.

Many approaches inspired by k-anonymity have been proposed for ONS. They differ on what is considered as "quasi" information. Examples of such definitions are given in Section 3.

2.4.2 Utility Criterion

Utility, also called information loss or anonymization quality, depends on the type of analysis that we want to perform on the anonymized graph.

In the case of relational data, in general, utility is measured using the sum of information loss in individual tuples. This may be quantified by the distance of the tuple in the original table from the anonymized tuple in the released table.

In the case of OSNs, utility may be measured by various metrics. Such metrics can be roughly put into two categories with regards of whether the goal is to:

- preserve general graph properties of the released graph, or
- maintain a high quality for the result of executing aggregate network queries on the released graph.

Some examples of graph properties that we want to maintain include betweenness (that measures the degree an individual lies between other individuals in the network, in their shortest path), closenes (that measures the degree an individual is near to all other individuals in the network directly or indirectly), the shortest distance between the vertex and all other vertices reachable from it, centrality (the counts the number of relationships to other individuals in the network), and path length (that is, the distances between pairs of vertices in the network).

In terms of graph properties, the best utility is achieved when the released graph G^* is isomorphic to the original graph G.

Definition 3 (Graph Isomorphism) *Given two graphs $Q = (V_Q, E_Q)$ and $G = (V_G, E_G)$, Q is isomorphic to G, if and only if there exists at least one bijective function $f : V_Q \rightarrow V_G$ such that for any edge $(u, v) \in E_Q$, there exists an edge $(f(u), f(v)) \in E_G$.*

Measuring utility in terms of aggregate queries is useful especially in the case in which the edges or the vertices of the graph are labeled. In such a case, we often want to compute the aggregate on some path or subgraph that satisfies a given condition. For example, assume that the vertices in the graph of Figure 2(b) are labeled with the age of the participants; an aggregate query would be to compute the average distance between people of similar ages as opposed to the average distance of people of different ages. Such queries are useful in many applications, for example in customer relationship management.

Often, utility is empirically evaluated through experiments on the anonymized graph.

3 Algorithms

There are two general approaches to constructing the anonymized graph G^*, namely releasing data vs releasing statistics. In this article, we focus on the former. The latter based on *differential privacy* [7] is beyond the scope of this work.

3.1 Naive Anonymization

With *naive* or *pseudo* anonymization, all identifiers in the initial graph G are replaced with random numbers in the released graph G*. An example is shown in Figure 3(a).

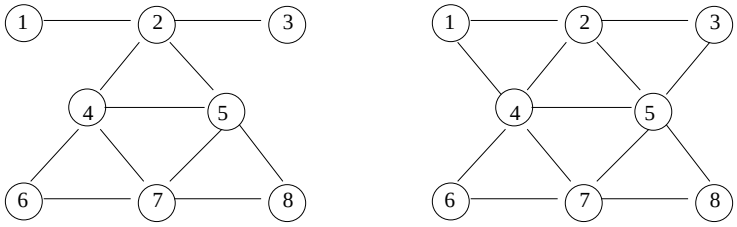
Let us consider vertex re-identification. Can an adversary deduce which vertex in the anonymized network in Figure 3(a) corresponds to Carol? With no external information, Carol could correspond with equal probability to any of the 8 vertices. But, what if the adversary knew that Carol has only one neighbor? Then, Carol can be mapped only to two vertices (namely vertices 1 and 3). With additional external knowledge such as the strength (or type) of the connection, an adversary could even reduce the candidate vertices in the hidden network to just a single one.

What about edge disclosure: can an adversary deduce that two identified individuals, say Alice and Bob, are connected in the anonymized network? With no external knowledges, Alice and Bob have a 11/28 likelihood of being connected. But what if, the adversary knew that Bob is the sole neighborhood of Alice, or the strength of their connection, or both?

The examples above show that pseudo-anonymization is vulnerable to various types of attacks. However note, that naive anonymization achieves the best utility, since the anonymized graph is isomorphic to the original network.

The approaches to OSN anonymization can be distinguished into two general categories:

1. *clustering-based* or *generalization* approaches that cluster vertices and edges into groups and replace a subgraph with a super-vertex, and



(a) (b)

Figure 3: (a) The naively anonymized graph of Figure 1(b) and (b) a 4-degree anonymous graph

2. *graph modification* approaches that modify the graph by inserting or deleting edges and vertices in the graph by either [10]:
   - *direct alteration*: adding or removing specific edges
   - *random alteration*: stochastically adding, removing or rewiring edges

Next, we present examples for each of the above approaches.

3.2 Clustering

With clustering-based approaches the edges and vertices of the OSN graph are grouped together. Then in the published graph, vertices are replaced by the formed groups.

We present next an approach that cluster vertices into groups of at least k vertices using a maximum likelihood approach. Other algorithms in this category include: grouping entities (e.g., individuals) into classes and masking the mapping between entities and the vertices that represent them in the anonymized graph [5] and (k, l)-grouping for bipartite graphs that preserve the underlying graph structure and instead anonymize the mapping from entities to vertices of the graph [6].

3.2.1 Case Study: Clustering by Vertex Partitioning

The approach of [12, 11] preserves anomymity against arbitrary structural knowledge by generalizing a naively-anonymized graph $G_a(V_a, E_a)$ by clustering its vertices to create a new anonymized graph $G_g(\mathcal{V}_g, \mathcal{E}_g)$. The vertices in V_a are parti-

tioned into disjoint sets, $\mathcal{V} \subseteq V_a$. These sets become the vertices of the generalized graph G_g. In a sense, the sets $\mathcal{V}$ can be thought of as super-vertices, since they contain vertices from G_a, but are themselves the vertices of G_g.

The superedges of $\mathcal{E}_g$ include self-loops and are labeled with non-negative weigths by the function d. G_g is a generalization of G_a under a partition $\mathcal{V}_g$, if the edge labels report the density of edges (in G_a) that exist within and across the partitions. Let $\mathcal{V}_g$ be the supervertices of G_g. G_g is a a generalization of G_a if, for all $\mathcal{V}_i, \mathcal{V}_j \in \mathcal{V}_g$, $d(\mathcal{V}_i, \mathcal{V}_j) = |\{(v_i, v_j) \in E_a \mid v_i \in \mathcal{V}_i \text{ and } v_j \in \mathcal{V}_j\}|$.

For any generalization G_g of graph G_a, we denote by $\mathcal{W}(G_g)$ all possible worlds (i.e., graphs over V_a) that are consistent with G_g. Vertices are partitioned so that the generalized graph satisfies privacy goals and maximizes utility. In the extreme case that all partitions contain a single vertex, $\mathcal{W}(G)$ contains just the graph G_a; function d encodes its adjacency matrix. At the other extreme, if all vertices are grouped into a single partition, then G_g consists of a single supervertex with a self-loop labeled with $|E_a|$ (i.e., the total number of edges in the original graph). In this case, $\mathcal{W}(G)$ is the set of all graphs over V_a with $|E_a|$ edges. In this case the generalization provides anonymity, but low utility, since it reflects only the edge density of the original graph.

Let G_g be a generalized graph such that each supervertex $\mathcal{V}_i$ has at least k vertices. Then G satisfies graph k-anonymity. The reason is that the generalized graph contains no information that allows the adversary to distinguish between two vertices in the same supervertex. Therefore, each of the k or more vertices in the same supervertex are equally likely candidates for being the target.

Given G_a and k, the proposed algorithm $GraphGen$ attempts to find the generalized graph that best fits G_a. Fitness is estimatd via a maximum likelihood approach. $GraphGen$ starts with a single partition (i.e., superedge) containing all vertices, and uses simulating annealing to search the space of possible generalizations. Each valid generalized graph (i.e., one in which each supervertex has at least k vertices) is a state in the search space. *GraphGen* proposes a change of state, by splitting a partition, merging two partitions, or moving a vertex to a different partition. Each proposed change is evaluated based on the change in likelihood that results. The proposal is always accepted if it improves the likelihood and accepted with some probability if it decreases the likelihood.

3.3 Graph Modifications

In general, graph modification techniques work by adding (or deleting) edges and vertices. We present next a number of direct alteration techniques that add (delete) specific vertices or edges to achieve k-degree, k-neighborhood and k-automorphic anonymity. We also present an approach that uses random alteration to preserve edge weight anonymity.

Approaches that use random alteration for identity disclosure include: spectrum preserving edge randomization by edge swapping [22] and using low rank approximation techniques to reconstruct the graph topology from the randomized network [21].

3.3.1 Case Study: k-Degree Anonymity

In this case study, privacy refers to identity disclosure and the background information of the adversary is the degree of the target vertex [13]. For example, assume that an adversary knows that the target user has 421 connections in a social network. If in the pseudo-anonymized graph, there is only one vertex with degree 421, the attacker can identify this vertex as being the targeted individual. This privacy criterion is expressed through k-degree anonymity.

Definition 4 (k-degree anonymity) *A graph $G(V, E)$ is k-degree anonymous if every vertex in V has the same degree as k-1 other vertices in V.*

The proposed algorithm uses direct graph modifications. Given a graph $G(V, E)$ and an integer k, the algorithm modifies G via a set of edge addition and deletion operations to construct a new k-degree anonymous graph $G'(V, E')$ in which every vertex has the same degree with at least k-1 other vertices

The utility objective is to keep the symmetric difference, *symdiff*, between G and G' defined as *symdiff*(G, G') = $|E/E' \cup E/E'|$, as small as possible, so that degree-anonymization does not destroy the structure of the graph. Utility is empirically evaluated through experiments that compare the transformed graph G and G' in terms of their average path length, clustering coefficient, and the exponent of power-law distribution.

The algorithm uses degree sequences. A degree sequence d is k-anonymous, if each distinct degree value in d appears at least k times. Clearly, a graph whose degree sequence is k anonymous is k-degree anonymous. For example, the degree

sequence of the graph of Figure 1(b), is [4, 4, 4, 4, 2, 2, 1, 1], meaning that the graph is 2-degree anonymous. The algorithm uses dynamic programming to construct a new sequence that is k-anonymous with the minimum number of alterations. For example, the above sequence can be made 4-degree anonymous by adding 2 edges, resulting in [4, 4, 4, 4, 2, 2, 2, 2]. Then, a graph is created that realizes this sequence by adding edges to the original G. A degree sequence d is realizable, if there exists a simple undirected graph with vertices having degree sequence d. For the example, the above sequence can be realized, by just adding an edge from Alice to Fred and from Carol to Harry, as shown in Figure 3(b). However, not all degrees sequences are realizable. Take, for example, degree sequences [1, 1, 1] or [3, 3, 3].

A heuristic is proposed that tries to realize a graph by adding noise (edges) in the sequence. In addition, greedy edge swaps are considered to improve the achieved *symmdiff*.

3.3.2 Case Study: k-Neighborhood Anonymity

In this case study, privacy again refers to identity disclosure, however the background information now is the immediate neighborhood of a vertex [25]. Take for example the case that the attacker knows that Bob has two neighbors for whom, he is the only neighbor.

Formally, the neighborhood of vertex $u \in V$ is the induced subgraph of the neighbors of u, denoted by $Neighbor_G(u) = G(N_u)$ where $N_u = \{v|(u,v) \in E\}$.

Definition 5 (k-neighborhood anonymity) *Assume a graph $G(V, E)$. A vertex $u \in V$ is k-neighborhood anonymous in G, if there are at least k-1 other vertices u_1, ... $u_{k-1} \in V$ such that $Neighbor_G(u)$, $Neighbor_G(u_1)$, ..., $Neighbor_G(u_{k-1})$ are isomorphic. G is k-anonymous if every vertex in G is k-anonymous.*

The algorithm uses direct alterations and works in two steps. During the first step, the neighborhoods of all vertices in the network are extracted and encoded to facilitate comparison between neighborhoods. The neighborhood $Neighbor_G(u)$ of each vertex u is divided into into components that correspond to the maximal connected subgraphs of $Neighbor_G(u)$. Each component is then encoded using a technique that utilizes a depth-first pre-order traversal of the graphs. During the second step, the vertices are greedily, organized into groups, and the neighborhoods

of vertices in the same group are anonymized. Anonymization is possible by both (a) adding edges and (b) generalizing edge labels.

The OSN model in [25] includes vertex labels from a label set L. The label set L forms a hierarchy. For example, if occupations are used as vertex labels, in addition to specific occupations such as dentist, general physician, high school teacher, and primary school teacher, L also contains general categories such as medical doctor and teacher. L also includes a special label, denoted by $*$, that corresponds to most general category generalizing all other labels. For any two labels l_1, $l_2 \in L$ we write $l_1 \prec l_2$, if l_1 is more general than l_2. For example, *medical doctor* $\prec$ *dentist*. In addition, $l_1 \preceq l_2$, if and only if, $l_1 \prec l_2$ or $l_1 = l_2$. Relation $\preceq$ is a partial order on L.

Having labels allows generalization by replacing specific categories by more general ones.

Utility is empirically measured by considering queries that compute some aggregate on some paths or subgraphs satisfying some given conditions E.g., Average distance from a medical doctor to a teacher

3.3.3 Case Study: k-Automorphic Anonymity

In this case, identity disclosure is studied, but now the general case is considered, where the adversary knows the subgraph of the OSN graph. This is modeled by an adversary query Q that corresponds to a subgraph of the OSN graph G [26]. The goal is to anonymize G so that there are at least k different results or *matches* for Q in the anonymized graph.

This is formalized using the notion of graph authomorphism. An automorphism of a graph $G = (V, E)$ is a permutation f of the vertex set V, such that for any edge $e = (u, v) \in E$, $f(e) = (f(u), f(v))$ is also an edge in G. That is, it is a graph isomorphism from G to itself under f.

If there exist k automorphisms in G, it means that there exists k-1 different automorphic functions.

Definition 6 (Sub-Graph Isomorphism) *Given two graphs Q and G, graph Q is sub-graph isomorphic to graph G, if there exists at least one sub-graph X in graph G such that Q is isomorphic to X under a bijective function f. Subgraph X is called a sub-graph match (match for short) of Q in G, while the vertex $f(u)$ in G is called the match vertex with regard to vertex u in Q.*

Definition 7 (k-automorphic graph) *A graph G is called a k-automorphic graph, if (a) there exist k-1 automorphic functions F_α (α = 1, ..., k-1) in G, and (b) for each vertex v in G, $F_{\alpha_1}(v) \neq F_{\alpha_2}(v)$, ($1 \leq \alpha_1 \neq \alpha_2 \leq k-1$).*

Definition 8 (Different Matches) *Given a sub-graph query Q, a graph G and two matches m_1 and m_2 of Q in G that are isomorphic to Q under functions f_1 and f_2 respectively, are called different, if there exists no vertex v in query Q, such that $f_1(v) = f_2(v)$.*

Definition 9 (k-different match principle) *Given a graph G any sub-graph query Q, G obeys the k-different match principle, if (a) there exist at least k matches of Q in G, and (b) any two of the k matches are different matches.*

The proposed K-Match (KM) algorithm uses direct alterations and works in three steps. In the first step, the graph is partitioned into n blocks which are then clustered in m groups so that each of the m groups contains at least k blocks. In the second step, the blocks are aligned to attain isomorphic blocks by adding edges. In the third step, an "edge-copy" technique is applied to handle matches that cross blocks.

Utility, called anonymization cost, is defined similarly with the symmetric distance as $Cost(G, G')$ = $(E \cup E') - (E \cap E')$.

3.3.4 Case Study: Privacy of Edge Weights

In this case study, we consider a social network with edge weights [15]. Here the goal is to protect the privacy of the weights.

Anonymization is achieved by modifying the edge weights. Utility is measured in terms of the shortest paths and their lengths, where the shortest path between two vertices is defined as the path with the minimum sum of weights. The authors show that there does not exist a perturbation schema such that the shortest paths and the corresponding lengths between any pair of vertices are both preserved. They present two complementary schemes that partially achieve these goals.

With Gaussian perturbation, the weight $w_{i,j}$ of each edge (v_i, v_j) in graph G is replaced in the anonymized graph G^* by a new weight $w^*_{i,j}$ computed as: $w^*_{i,j} = w_{i,j} \times x_{i,j}$, where $x_{i,j}$ is a randomly generated number from the Gaussian distribution $N(0, \sigma^2)$. Number $x_{i,j}$ can be generated locally by v_i and v_j. Vertices v_i and

v_j may independently generate random numbers $x^1_{i,j}$ and $x^2_{i,j}$ respectively from the Gaussian distribution $N(0, \sigma^2)$. Then, $x_{i,j}$ may be set as the average value of $x^1_{i,j}$ and $x^2_{i,j}$. However, Gaussian perturbation does not maintain the same shortest path when the perturbation (i.e., σ) gets large.

Greedy perturbation achieves the utility objective for a given set H of shortest paths in a static network. Edges in G are divided into three types with respect to set H: (1) non-visited edges, that are edges that no shortest path in H passes through them, (2) all-visited edges, that are edges that all shortest paths in H pass through them, and (3) partially-visited edges that are edges such that some but not all shortest paths in H passes through them. The greedy algorithm carefully increments or decrements the weights of an edge based on its type using the following observations. Increasing the weight of a non-visited edge does not change any shortest path in H or its corresponding length. Decreasing the weight of an all-visited edge does not change any shortest path in H, but may change their lengths. For partially visited edges, both increasing and decreasing their values may lead to modifying shortest paths in H. Thus, such modifications are appropriately constrained.

3.4 Extensions

We discuss next two important extensions with regards to anonymization algorithms: (a) personalization and (b) dynamic networks.

Personalization All approaches presented so far assume that all users have similar privacy needs. In practice, however, users have different privacy protection requirements. For example, in Facebook, a user can specify what part of her profile or her connections should be made visible to others.

The authors of [23] present a personalized framework that offers three different levels of protection requirements based on gradually enhancing the background knowledge of the attacker. Specifically, for a vertex v in a published labeled graph: *Level 1* assumes that the attacker knows only the label of v, *Level 2* assumes that the attacker knows both the label and the degree of v and *Level 3* assumes that, besides the label and degree of v, the attacker also knows the labels on the edges adjacent to v. The proposed algorithms combine label generalization and direct graph alteration methods. Specifically, for Level 1 protection, vertex label generalization is used. For Level 2 protection, vertex/edge adding methods are combined with the

protection for Level 1, while for Level 3 protection, an edge label generalization is used in addition.

Dynamic Networks Social networks evolve over time. Thus, often multiple snapshots (or, versions) of the same network at different time instances may be published to allow analysis of the evolution of the network and perform longitudinal data analysis. This creates the potential for a new kind of attack, since an adversary may deduce private information by combining information from different versions of the same network. For example, knowing that the target has reduced its connections by 7 can lead to identity disclosure if there is just a single vertex with this characteristics.

The class-based clustering anonymization method of [5] is extended in [2] for the dynamic case using link prediction algorithms to model the evolution of the social network. Based on the predicted edges, a grouping of vertices is chosen so that not only do existing edges meet an appropriate grouping condition for privacy, but also future edges are unlikely to violate it either. The KM algorithm of [26] also supports dynamic releases by a vertex ID generalization method that allows lists of IDs to be assigned to each vertex so that each particular ID is hidden within the list.

4 Beyond Data Publishing

In the previous sections, we focused on data anonymization achieved through publishing a transformed graph. In this section, we discuss other approaches to privacy for OSNs.

A different line of research has been focusing on the users of OSNs. Various tools have been proposed to allow users *specify* their privacy requirements and *measure* the potential privacy risks of the information that they share.

Most of the available social network management systems offer to theirs users only coarse control in specifying their privacy requirements. In general, users are allowed to characterize a given piece of information only as being public, private, or accessible to a limited set of other users (e.g., their fiends). More advanced specification languages have been proposed. For example, an access control mechanism that adopts a rule-based approach in specifying access policies is proposed in [3]. In this approach, authorized users are denoted in terms of the type, depth,

and trust level of the relationships they have formed in the social network. Access to a resource is granted only if the requester succeeds in demonstrating that she is authorized to do so by providing a proof.

While fine-grained privacy control is useful, it is difficult for the average user to specify such detailed policies. To address this problem, a template for the design of a social networking privacy wizard is proposed [8]. The intuition for the design of the wizard comes from the observation that privacy preferences of the users (such as which friends should be able to see which information) usually follow some implicit set of rules. Thus, it is possible to build a machine learning model to concisely describe such preferences using only limited amount of user input. This model can then be used to configure the privacy settings of each user automatically.

Another approach in assisting users to compose and manage their access control policies is proposed in [19]. This approach is based on a supervised learning mechanism that in order to build classifiers uses as training sets example policy settings provided by the users. These classifiers are then used to auto-generate access control policies. In addition, users are given the possibility to fuse policy decisions provided by their friends or other users of the OSN.

There has been a lot of work also on the topic of accessing the privacy risks of the information revealed by users in social networks. The authors of [24] show how an adversary can exploit an OSN where both public and private user profiles exist to predict the private attributes of a user. Prediction is achieved by mapping the problem to a relational classification problem and using friendship and group membership information which is often not hidden to infer sensitive attributes.

The authors of [14] propose a framework for computing the privacy score of each user in an OSN. This score indicates the potential risk for the user caused by her participation in the OSN. The definition is based on the assumption that the privacy score of a user increases as she discloses more sensitive information. In addition, the risk to privacy becomes higher if the information is more visible. A mathematical model is developed to estimate both sensitivity and visibility of the information.

Beyond anonymization, privacy can be achieved by restricting access to data. Instead of publishing a transformed graph, access to social network data can be provided through querying such data. Prior work on querying private data considers either (i) auditing, or (ii) adding noise to answers [10]. With *query auditing*, a query is denied, if its answer may lead to privacy breaches [16]. With *perturba-*

tions, random noise is added to the result of a query to hide private data.

5 Summary

In this article, we have focused on privacy issues in online social networks (OSN). Publishing OSN data even when the identity of the users is hidden may lead to privacy leaks. To protect privacy, various data anonymization techniques have been proposed. Anonymization techniques fall into two general categories: (i) generalization by clustering and (ii) graph transformations. The complexity of such techniques depends on the range of attacks they can handle. Such attacks are usually structural: the attacker has prior knowledge of the social connections in the neighborhood of the target. Since published data are used for analysis, anonymization must be such that the utility of the resulting data is not compromised.

Acknowledgment

This research was supported by a Marie Curie International Outgoing Fellowship (PIOF-GA-2009-237524) within the 7th European Community Framework Programme

References

[1] Lars Backstrom, Cynthia Dwork, and Jon M. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International Conference on World Wide Web, (WWW)*, pages 181–190, 2007.

[2] Smriti Bhagat, Graham Cormode, Balachander Krishnamurthy, and Divesh Srivastava. Privacy in dynamic social networks. In *Proceedings of the 19th International Conference on World Wide Web, (WWW)*, pages 1059–1060, 2010.

[3] Barbara Carminati, Elena Ferrari, and Andrea Perego. Enforcing access control in web-based social networks. *ACM Trans. Inf. Syst. Secur.*, 13(1), 2009.

[4] Bee-Chung Chen, Daniel Kifer, Kristen LeFevre, and Ashwin Machanavajjhala. Privacy-preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167, 2009.

[5] Graham Cormode, Divesh Srivastava, Smriti Bhagat, and Balachander Krishnamurthy. Class-based graph anonymization for social network data. *PVLDB*, 2(1):766–777, 2009.

[6] Graham Cormode, Divesh Srivastava, Ting Yu, and Qing Zhang. Anonymizing bipartite graph data using safe groupings. *VLDB J.*, 19(1):115–139, 2010.

[7] Cynthia Dwork. Differential privacy: A survey of results. In *Theory and Applications of Models of Computation, 5th International Conference, (TAMC) Proceedings*, pages 1–19, 2008.

[8] Lujun Fang and Kristen LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th International Conference on World Wide Web, (WWW)*, pages 351–360, 2010.

[9] Ralph Gross, Alessandro Acquisti, and H. John Heinz III. Information revelation and privacy in online social networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society, (WPES)*, pages 71–80, 2005.

[10] Michael Hay, Kun Liu, Gerome Miklau, Jian Pei, and Evimaria Terzi. Privacy-aware data management in information networks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, (SIGMOD)*, pages 1201–1204, 2011.

[11] Michael Hay, Gerome Miklau, David Jensen, Donald F. Towsley, and Chao Li. Resisting structural re-identification in anonymized social networks. *VLDB J.*, 19(6):797–823, 2010.

[12] Michael Hay, Gerome Miklau, David Jensen, Donald F. Towsley, and Philipp Weis. Resisting structural re-identification in anonymized social networks. *PVLDB*, 1(1):102–114, 2008.

[13] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, (SIGMOD)*, pages 93–106, 2008.

[14] Kun Liu and Evimaria Terzi. A framework for computing the privacy scores of users in online social networks. *TKDD*, 5(1):6, 2010.

[15] Lian Liu, Jie Wang, Jinze Liu, and Jun Zhang 0001. Privacy preservation in social networks with sensitive edge weights. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*, pages 954–965, 2009.

[16] Shubha U. Nabar, Krishnaram Kenthapadi, Nina Mishra, and Rajeev Motwani. A survey of query auditing techniques for data privacy. In *Privacy-Preserving Data Mining*, pages 415–431. Springer, 2008.

[17] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, pages 173–187, 2009.

[18] Pierangela Samarati. Protecting respondents' identities in microdata release. *IEEE Trans. Knowl. Data Eng.*, 13(6):1010–1027, 2001.

[19] Mohamed Shehab, Gorrell P. Cheek, Hakim Touati, Anna Cinzia Squicciarini, and Pau-Chen Cheng. Learning based access control in online social networks. In *Proceedings of the 19th International Conference on World Wide Web, (WWW)*, pages 1179–1180, 2010.

[20] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[21] Leting Wu, Xiaowei Ying, and Xintao Wu. Reconstruction from randomized graph via low rank approximation. In *Proceedings of the SIAM International Conference on Data Mining, (SDM)*, pages 60–71, 2010.

[22] Xiaowei Ying and Xintao Wu. Randomizing social networks: a spectrum preserving approach. In *Proceedings of the SIAM International Conference on Data Mining, (SDM)*, pages 739–750, 2008.

[23] Mingxuan Yuan, Lei Chen, and Philip S. Yu. Personalized privacy protection in social networks. *PVLDB*, 4(2):141–150, 2010.

[24] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of*

the 18th International Conference on World Wide Web, (WWW), pages 531–540, 2009.

[25] Bin Zhou and Jian Pei. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the 24th International Conference on Data Engineering, (ICDE)*, pages 506–515, 2008.

[26] Lei Zou, Lei Chen, and M. Tamer Özsu. K-automorphism: A general framework for privacy preserving network publication. *PVLDB*, 2(1):946–957, 2009.