Enabling accurate analysis of private network data

Michael Hay Gerome Miklau David Jensen
Department of Computer Science
University of Massachusetts Amherst
{mhay,miklau,jensen}@cs.umass.edu

August 28, 2009

Abstract

This is a draft copy of a chapter in the forthcoming book *Privacy-Aware Knowledge Discovery: Novel Applications and New Techniques* to be published by Chapman & Hall/CRC Press.

Contents

1 Introduction 3
1.1 How are networks analyzed? 5
1.2 Why should network data be kept private? 5
1.3 Are privacy and utility compatible? 6
1.4 Organization 6

2 Attacks on Anonymized Networks 7
2.1 Threats: re-identification and edge disclosure 8
2.2 Adversary knowledge 8
2.3 Attacks 9
2.3.1 Degree signature attacks 10
2.3.2 Injection attack 10
2.3.3 Auxiliary-network attack 11
2.4 Attack effectiveness 13
2.4.1 Effectiveness of degree signature attacks 13
2.4.2 Effectiveness of injection attacks 13
2.4.3 Effectiveness of auxiliary-network attacks 13

3 Algorithms for Private Data Publication 15
3.1 Directed alteration of networks 15
3.1.1 Degree anonymity 15
3.1.2 Neighborhood anonymity 16
3.2 Network Generalization 17
3.2.1 Anonymity through clustering 18
3.2.2 Safe groupings 19
3.3 Randomly altering networks 20
3.3.1 Uniform randomization 21
3.3.2 Spectrum-preserving randomization 21

4 Algorithms for Private Query Answering 22
4.1 Differential privacy 22
4.2 Differential privacy for networks 23
4.3 Algorithm for differentially private query answering 23
4.4 Network analysis under differential privacy 24
4.4.1 Low sensitivity analyses 24
4.4.2 High sensitivity analyses 25

5 Conclusion and Future Issues 26

6 Acknowledgments 26

1 Introduction

Many phenomena can be modeled as networks in which entities (represented by nodes) participate in binary relationships (represented by edges). In a social network, nodes are individuals and edges are personal contacts or relationships. In a communication network, nodes are individuals and edges are flows of information such as phone calls or email messages. In a technological network, nodes are machines, such as computers or power stations, and edges are some means of transmission.

Research into the structure and function of networks has wide-ranging applications. Network analysis is being used by businesses to market products [31], by epidemiologists to combat the spread of diseases [32], and by financial regulatory agencies to detect fraud among securities dealers [20]. Network analysis has also been applied to national security, industrial engineering and computer network design.

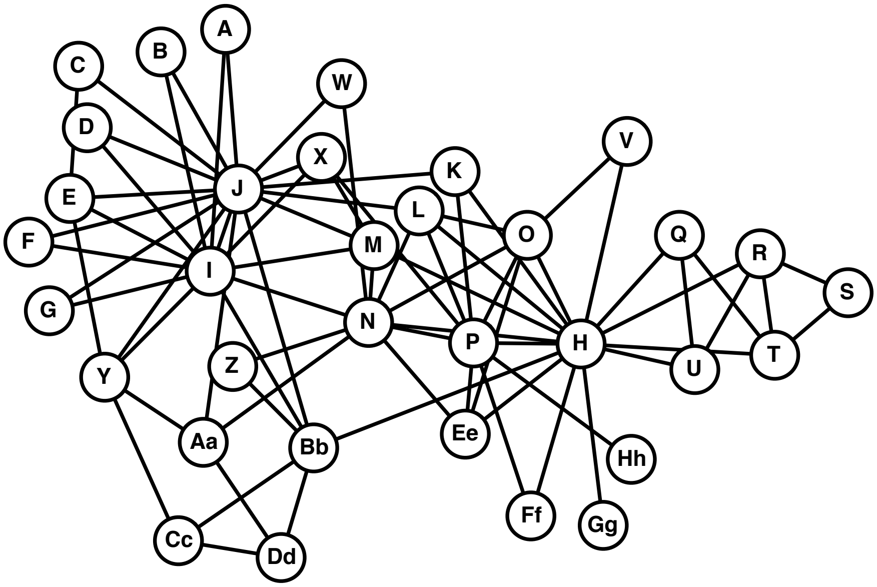
The collection, dissemination, and analysis of networks has become much easier in recent years. In the early days of social network analysis, data was often collected manually through interviews or surveys. As a result of the cost of data collection, few networks were available to researchers, and those available were small—sometimes consisting of fewer than 100 nodes. Figure 1 reproduces a widely studied 34-node social network of karate club members, taken from Zachary [66]. Now, because so many of our interactions leave electronic traces, networks can be systematically recorded, sometimes on a nearly global scale. For example, Leskovec and Horvitz's [38] study of Microsoft Messenger users examined a network of 180 million nodes and 1.3 billion edges.

Some speculate that data at this scale can revolutionize the social sciences—much the way data has transformed biology and physics—leading to a data-driven "computational social science" [35]. However, scientific progress has been impeded because much of the data is not yet available to the scientific community at large. The institutions and service providers that collect the data are often unwilling to release it. Their reluctance is due, in part, to concerns about privacy, since network data can reveal sensitive facts about participating entities and their connections. As a result, computational social science is happening, but only at the institutions that own the data or by a limited set of researchers who have negotiated access to the data (e.g., [22, 34, 38, 49, 53]). If the data is not accessible to the larger scientific community, then results cannot be replicated and findings cannot be challenged. For the field of computational social science to flourish, we must remove the obstacles to data sharing caused by privacy concerns.

This article describes techniques for the private and accurate analysis of network data. We assume the data owner possesses a graph representing a network of interest to analysts, and we consider two strategies for the safe release of the data. In *private data publication*, a transformed network is released for analysis by the data owner. In *private query answering*, an analyst submits queries to the data owner and receives answers, often transformed or perturbed by the data owner to protect privacy. In order to be useful to the analysts, the transformed data must accurately reflect the original graph, and the techniques that transform this data must be efficient enough to scale to large data sets.

Both strategies have an important role in enabling safe dissemination of network data. In data publication, the analyst receives a transformed graph and performs analysis using local resources. This means the data owner does not have to devote resources on behalf of the analyst. It also allows the analyst to carry out tasks that are difficult to do through a query interface, such as data cleaning, exploration, and transformations. Lastly, the analyst's processing of the graph is not revealed to the data owner. In query answering, the analysts receive only the answers to specific queries. The main advantage of this strategy is that it can typically offer more accurate results since the distortion introduced to protect privacy is targeted to specific queries.

Both of these approaches have been explored in the literature, but most existing work assumes that the private data can be represented using a single table. Typically each record in the table corresponds to a separate entity and, in contrast to network data, and there are no



(a)

Id	**Name**	**Gender**	**Age**	**Belt**	**Injury**
A	Alice	F	18	yellow	ankle
B	Bob	M	21	white	ankle
C	Carol	F	42	white	elbow
D	Dave	M	16	white	none
E	Ed	M	19	white	clavicle
F	Fred	M	21	white	knee
G	Gail	F	21	white	none
H	Mr. Hi	M	36	black	none
I	Ingrid	F	22	brown	none
J	John	M	45	black	ribs
		...			

(b)

Edge	**YearsKnown**
{ A, I }	2
{ A, J }	3
{ B, I }	10
{ B, J }	11
{ C, J }	20
{ C, E }	1
...	

(c)

Figure 1: (a) A social network representing friendships among karate club members, from Zachary [66]. (b) An example vertex table V. The rows correspond to vertices (nodes) in the graph shown in (a). (c) An example edge table E. The rows correspond to edges in the graph shown in (a).

relationships between the entities. Data publishing techniques for tabular data are not well-suited for network data because they fail to account for the interconnectedness of the entities. Naively applied, they tend to destroy the network structure, which can be a crucial part of the analysis of such data. At the same time, threats considered for tabular data do not account for the vulnerabilities posed by the connections in a network.

Existing query answering techniques can be applied to network data, however, the kinds of queries posed on network data differ from the typical queries posed on tabular data. We review recent work that adapts existing query answering techniques to accurately carry out important network analyses.

While the algorithms designed for tabular data do not always apply to network data, some of the definitions of privacy, such as k-anonymity [57, 58, 61] and differential privacy [13, 15], can be adapted to network data.

Much research has explored related privacy problems in network data, particularly on problems that surface with online social networks [1, 7, 18, 21, 23, 29, 33, 41, 67, 68]. While all of this work shares a common goal of keeping sensitive information private, the privacy issues that arise with data analysis are distinct from those that arise with, say, access control. Typically, the analyst is interested in facts about the network's overall topology and not with the particular connections between individuals. Thus, the overarching goal is to reveal properties of the network (in the aggregate) but hide sensitive properties of particular entities and their relations.

1.1 How are networks analyzed?

If the techniques for protecting privacy are to have practical application, they must allow an analyst to measure important network properties and carry out common analyses. After introducing a network that will serve as a running example, we briefly review some ways that networks are studied.

We represent a network as a graph consisting of vertices, described by vertex table V, and edges, described by edge table E. Figure 1 shows the vertex and edge table for the karate club network. Each node has a unique identifier (e.g., $A, B, C \dots$), and edges are unordered pairs of node identifiers (e.g., $\{A, I\}$). In some networks, nodes and edges may have attributes, which are represented as auxiliary columns in tables V and E. A directed graph can include a direction attribute in E. In the figure, the node attributes are *Name*, *Gender*, *Age*, *Belt*, and *Injury*, and the edge attribute is *YearsKnown*. (The attributes on the nodes and edges have been created for the purpose of illustration and were not present in the original network from Zachary [66].)

Many analyses focus solely on measuring the topology of the network as defined by the edge table. Such analyses include the distribution of node degrees, the distribution of path lengths (including the diameter—the longest minimum length path between two nodes), and measures of transitivity (defined as the likelihood that a node's neighbors are directly connected). Significant research effort has been devoted to models of network formation that generate graphs possessing the structural properties seen in the real world [3, 8, 6, 24, 39, 37, 36, 63].

Some analyses pinpoint specific structural features of the network. Analysis of network centrality [19] seeks to identify influential nodes. For example, in Figure 1, node H may be considered influential because a large number of shortest paths include H. In addition, community discovery [50] divides the network into meaningful clusters. Motif analysis [46] identifies interesting structures that occur repeatedly in a network.

Another category of research focuses on understanding the function of the network by modeling processes that occur within the network. Such processes include search or navigation within networks [59, 62] and diffusion across networks (e.g., rumors or epidemics spreading in a group) [31].

While the above analyses focus on the structure of the graph, the presence of attributes on edges or nodes allow for some new analyses and variants of those above. For example, homophily, the tendency for associations to form among similar individuals, can be measured in a network with attributes on nodes [44, 54]. Network models have been developed that model the correlation between structural features and attributes [24]. Finally, network data can include temporal information, allowing the study of network dynamics. This study includes the development of models of network formation and evolution [36] and models to accurately predict future links [34, 40].

The above summary is incomplete but shows the diversity of analyses that are performed on network data. More complete surveys of network analysis appear in the literature [51, 12].

1.2 Why should network data be kept private?

Many networks contain sensitive information, which may be disclosed through network analysis. In cases where entities represent individuals, network data often contains personal information. Even if the entities in a network are not individuals, network data may still be sensitive. For example, detailed topological information about the power grid may reveal vulnerabilities to potential terrorists, or records of information flow between host machines in a computer network may reveal applications running on those hosts or facts about host operators.

The vertex table of a network raises the common privacy problems of tabular data: descriptive attributes associated with an entity may be sensitive, and even the inclusion of an entity in the data can itself be sensitive. For example, the first tuple in Figure 1b reveals that Alice is in the karate club, that she is 18 years old, and that she has injured her ankle.

Of course, the distinctive feature of graph data is the presence of relationships between entities represented by the edge table. The existence of an edge may be sensitive. For example, Alice may want to hide the fact that she and Ingrid are friends. Alternatively, Dave may be willing to disclose that Ingrid and John are his friends, but he may wish to hide the fact that he has only two friends. Other aspects of connections may also be sensitive. For example, Ingrid may prefer to hide the fact that she is friends with a 45 year-old man, or that she associates primarily with white belts. Or the degree of a vertex may be sensitive: academics in a scholarly collaboration network may wish to hide their low degree, while participants in a network of romantic contacts may wish to hide their high degree.

In summary, both the attributes of a node and the structure of connections around a node can be sensitive. The diversity of sensitive information in network data pose challenges for privacy mechanisms.

1.3 Are privacy and utility compatible?

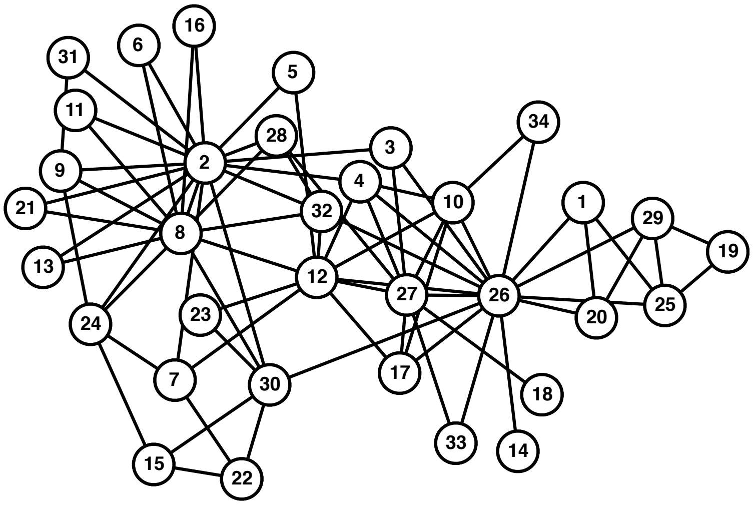
Given the range of sensitive information to be protected in networks, maintaining privacy and enabling accurate analysis may be impossible. As our review will show, existing work has encountered some limits on what analyses can be accurately studied. With each of the data publishing approaches, the transformations that are applied to protect privacy systematically distort some important network properties, and it is not clear how to lessen the distortion without weakening the privacy protection. With the query answering approaches, it has been shown that there are some network analyses that are impossible to answer accurately under existing privacy definitions.

At the same time, some positive results have emerged. Many network properties are preserved in the transformed networks, and the transformations provide practical privacy protection. In the query answering setting, it is possible to answer some queries with extreme accuracy while also guaranteeing extremely strong privacy guarantees.

These results are encouraging, but leave many open questions about the nature of the relationship between privacy and utility. Understanding what properties of networks can be safely released remains one of the main open questions in this area of research.

1.4 Organization

In the remainder of this article, we first discuss attacks on network data and then discuss strategies for protecting privacy, reviewing both data publishing techniques and query answering techniques. In Section 2, we discuss attacks on networks that are anonymized simply by removing identifiers from nodes, showing that this common approach is insecure. We review three specific attack strategies, show how they can lead to the disclosure of sensitive information, and evaluate their effectiveness through simulated attacks on real networks. In Section 3, we review approaches that protect privacy by transforming the graph either through directed alteration, generalization, or random alteration. In Section 4, we review private query answering approaches, in which perturbed query answers are returned to the analyst. We conclude in Section 5 by addressing future challenges in the private analysis of networks.



(a)

Alice	16
Bob	6
Carol	31
Dave	11
Ed	9
Fred	21
Gail	13
Mr. Hi	26
...	...

(b)

Id	Gender	Age	Belt	Injury
16	F	18	yellow	ankle
6	M	21	white	ankle
31	F	42	white	elbow
11	M	16	white	none
9	M	19	white	clavicle
21	M	21	white	knee
13	F	21	white	none
26	M	36	black	none
8	F	22	brown	none
2	M	45	black	ribs
		...		

(c)

Edge	YearsKnown
{ 16, 8 }	2
{ 16, 2 }	3
{ 6, 8 }	10
{ 6, 2 }	11
{ 31, 2 }	20
{ 31, 9 }	1
...	

(d)

Figure 2: (a) Naive anonymization of the network from Figure 1; (b) Secret mapping between names and synthetic identifiers; (c) Anonymized vertex table and (d) Anonymized edge table.

2 Attacks on Anonymized Networks

A common strategy for protecting sensitive networks is *naive anonymization.* Naive anonymization is a simple transformation of a graph in which names of nodes (or other unique identifiers) are removed and replaced by synthetic identifiers. For example, Figure 2 shows the naively anonymized version of the karate club network in Figure 1. Also shown (Figure 2b) is the random mapping between names and synthetic identifiers. This is secret and known only by the data owner.

A naively anonymized network can support many analyses accurately since the topology of the graph and non-identifying attributes are unmodified. Unfortunately, this also makes it vulnerable to attack by an adversary with some prior knowledge about entities in the network. Using knowledge of attribute values, graph structure, or both, the adversary may be able to re-identify targeted individuals in the naively anonymized network.

In this section, we categorize attacks on naively anonymized networks, and review empirical results on the effectiveness of these attacks. The results demonstrate that naive anonymization has considerable privacy risk. Equally important, the vulnerabilities of naive anonymization exposed by the attacks provide insight into strategies for improved anonymization techniques, such as those discussed in Section 3.

2.1 Threats: re-identification and edge disclosure

Most researchers have focused on one of two threats to naively anonymized data: *node re-identification* and *edge disclosure*. Node re-identification occurs when an adversary accurately identifies the named individual corresponding to a node in the naively anonymized graph. Once node re-identification occurs, further disclosures are likely. If the published network includes attributes, those attributes are now associated with the identified individual. In addition, the position of the individual in the network is now revealed, which can lead to disclosure of various structural properties, as discussed in Section 1.2.

Re-identification can either be *complete* or *partial*. Given a target individual, we say that the adversary has completely re-identified the target if the adversary knows which node in the anonymized graph corresponds to the target. For example, the adversary knows that Ed is node 9. With partial re-identification, the adversary may be uncertain about which node corresponds to the target, but the adversary has succeeded in winnowing the possibilities down to a small set of *candidates*. For example,the adversary may know that Ed is one of $\{7, 9, 17, 25, 28, 29\}$. The size of the candidate set is a natural measure of re-identification risk.

Edge disclosure occurs when an adversary is able to accurately infer the existence of an edge between two named individuals. Disclosing the *absence* of an edge may also be considered a violation of privacy. However, since most real networks are sparse—meaning most edges are absent—the adversary's ability to infer the presence of an edge tends to be a greater concern.

Node re-identification and edge disclosure are distinct threats, and researchers have often considered them separately. Of course, in a naively anonymized graph, re-identification of multiple nodes leads to edge disclosure: if both Alice and John are re-identified, then the presence of an edge between them can be determined by inspection. However, when we move beyond naive anonymization to more complex transformations in Section 3, we will see that re-identification does not necessarily imply edge disclosure since some of the edges in the anonymized network may be artificial. Edge disclosure can also occur without complete node re-identification. For example, suppose the adversary has identified a set of candidate matches for Alice—$\{6, 9, 11, 16, 28\}$—and a set of candidate matches for John—$\{2, 8\}$. The adversary can conclude that Alice and John must be connected because each candidate for Alice is connected to each of the candidates for John.

Node re-identification and edge disclosure are simple instances of disclosure and the two most commonly considered in the literature. In some settings, more complex disclosures may be of interest to an adversary, such as the disclosure of specific attributes or structural patterns. For example, in a network of sexual interactions, revealing a person's degree (i.e., the number of sexual partners) constitutes a sensitive disclosure.

2.2 Adversary knowledge

The adversary's ability to attack a naively anonymized network depends critically on the adversary's background knowledge. Here we discuss adversary knowledge and its implications for attacks on naively anonymized networks. Section 2.3 reviews some specific attacks.

The adversary may use knowledge of node attributes to attack an anonymized graph. Such attacks have been widely studied in the context of tabular data (e.g., the widely reported attack of Sweeney [60] which re-identified anonymized medical records using voter registration data). Because attribute-based attacks are relatively well-understood, research on network attacks has focused on the novel threat of an adversary obtaining knowledge about the structure of the graph.

A common assumption is that the adversary has structural knowledge of a small subgraph surrounding the target(s) [2, 43, 27, 26, 67, 69]. For example, the adversary may know that Alice has degree 2, or that Alice has two neighbors who are themselves connected. The adversary may know about multiple targets—e.g., the adversary may know that Alice and Bob share two neighbors.

The above structural patterns describe *what* the adversary knows, but not how the knowledge was acquired. The adversary's knowledge could be derived from many possible sources (specific sources are discussed in Section 2.3). The source of the knowledge, particularly its credibility, is important because it affects the strategy that the adversary uses to find matches in the naively anonymized network.

We can distinguish sources in terms of whether they provide *precise* or *approximate* information. If the adversary's source is precise, then any facts learned about a target are also true of the corresponding node in the anonymized graph. In contrast, if the adversary's source is approximate, then it may assert properties about a target that are inaccurate or only roughly describe the corresponding node in the anonymized graph. Typically, it is assumed that the adversary knows whether the information source is precise or approximate.

Example 1 *An example of an adversary with* precise *knowledge is one that learns that Alice has degree exactly 2. Therefore, the node in the anonymized network that corresponds to Alice also has degree 2.*

An example of an adversary with approximate *knowledge is one that learns that the network is dynamic and, at some point in recent time, Alice's degree was 3. Therefore, the node in the anonymized network that corresponds to Alice is likely to have degree 3 but may also have a smaller or larger degree.*

In practice, we might expect the adversary's knowledge will be approximate rather than precise. As Example 1 suggests, one reason is that networks change over time and the adversary's knowledge may be derived from a different snapshot than the published network. Approximate knowledge also arises when the adversary lacks direct access to the true network data but instead derives knowledge from a related auxiliary source, as is the case with the auxiliary-network attack discussed in Section 2.3.

While approximate knowledge may be more realistic, precise knowledge is nevertheless very useful to study. First, when assessing privacy risk, it makes sense to evaluate the worst-case scenario, and an adversary is most powerful when the knowledge is precise. Second, in some cases, the adversary can, in fact, acquire precise knowledge (see the injection attack discussed in Section 2.3). Finally, approximate knowledge is difficult to model: it is not always clear what kinds of uncertainty will arise in practice.

When knowledge is precise, the adversary can execute an attack on the anonymized network by simply looking for matching subgraphs. The anonymized network will contain at least one match, and if it contains only one match, then the adversary has re-identified the target. However, with approximate knowledge, the adversary must somehow account for the uncertainty of the knowledge about the target. Thus, the adversary may consider subgraphs that only partially match, and perhaps rank them based on the quality of the match. An appropriate measure of match quality will depend on the information source and what kinds of errors are likely.

2.3 Attacks

We highlight three attacks that have been proposed in the literature. First, we describe a family of attacks, proposed by Hay et al. [27, 26], that cover a range of adversaries and can be a useful tool for assessing re-identification risk. Second, we describe the injection attack of Backstrom et al. [2], in which an adversary injects a distinctive subgraph into the network *prior* to anonymization. Finally, we describe a recent attack of Narayanan and Shmatikov [48] which exploits the availability of noisy, approximate knowledge to conduct a large-scale attack.

2.3.1 Degree signature attacks

Hay et al. [27, 26] describe a sequence of attacks, of increasing power, that use knowledge of local graph structure to re-identify a single target node.

The attacks model an adversary who is capable of learning about node degrees. Specifically, given a target x, the adversary learns the *degree signature* of x, which is denoted $\mathcal{H}_i(x)$ for $i = 1, \ldots$. At $i = 1$, the signature $\mathcal{H}_1(x)$ simply reveals the degree of target x to the adversary. Each signature in the sequence reveals an increasingly detailed description of the neighborhood around the target: $\mathcal{H}_2(x)$ reveals the degrees of the target's neighbors, $\mathcal{H}_3(x)$ reveals the degrees of the neighbors' neighbors, etc. The signatures can be defined iteratively, where $\mathcal{H}_i(x)$ is a multi-set of the $\mathcal{H}_{i-1}$ signatures of x's neighbors:

$$\mathcal{H}_i(x) = \{\mathcal{H}_{i-1}(z_1), \mathcal{H}_{i-1}(z_2) \ldots, \mathcal{H}_{i-1}(z_m)\}$$

where $z_1 \ldots z_m$ are the neighbors of x. Degree signatures, which are called vertex refinement queries by Hay et al. [27, 26], are inspired by a process called iterative vertex refinement that was originally developed to efficiently test for the existence of graph isomorphisms [11].

Example 2 *If Ed is the adversary's target, then* $\mathcal{H}_1(Ed) = \{4\}$ *since Ed has 4 neighbors.* $\mathcal{H}_2(Ed) = \{2, 12, 16, 5\}$ *because these are the degrees of Ed's neighbors* $\{C, I, J, Y\}$*.*

Suppose the adversary attacks the naively anonymized network (Figure 2) using knowledge of Ed's $\mathcal{H}_1$ *signature. Any node who has degree 4 is considered a match, so the candidates for Ed are* $\{7, 9, 17, 25, 28, 29\}$*. However, if the adversary attacks using knowledge of Ed's* $\mathcal{H}_2$ *signature, the adversary can re-identify Ed as node* 9 *because Ed's* $\mathcal{H}_2$ *signature is unique.*

Although degree signatures $\mathcal{H}_i$ for $i \geq 2$ might be considered unrealistically powerful knowledge, they are appealing as a tool for assessing re-identification risk. First, since they are parameterized, the data owner can assess risk across a range of adversaries. Second, it can be shown that for almost all graphs, the sequence of degree signatures converges to complete knowledge of graph structure, representing a worst-case adversary [26, 27]. Finally, degree signatures can be computed efficiently; in contrast, to measure risk using models based on subgraph patterns requires solving instances of the subgraph isomorphism problem, which is NP-Hard.

Degree signatures have also been used to understand what properties of a graph make it more or less vulnerable to adversary attacks. For example, density appears to play a key role, with re-identification risk increasing as graphs become more dense [26].

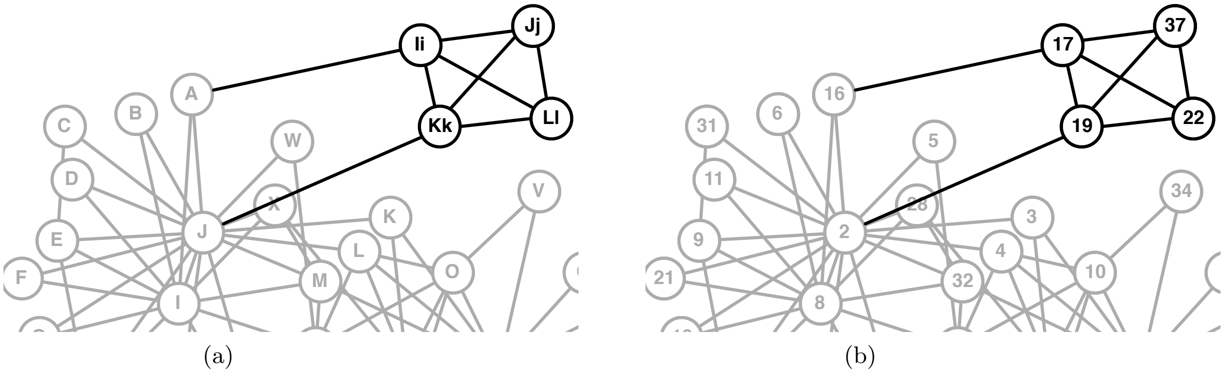
2.3.2 Injection attack

Backstrom et al. [2] introduce the idea of an *injection attack*. It is a special case of a subgraph matching attack where the adversary is a participant in the network and capable of altering the network structure prior to its publication. Specifically, the adversary can create new nodes and, from these nodes, add edges to any other node in the network. For example, in a network of email communication, the adversary could add a node by creating a new user account and add an edge by sending an email from this account to another account.

The idea behind the attack is to insert a group of nodes with a distinctive pattern of edges among them. The adversary then links this distinctive structure to some set of targeted individuals. When the naively anonymized network is published, the adversary uses precise knowledge of the injected subgraph to perform a matching attack. If successful, the targets, who are connected to the subgraph, are re-identified and the edges between them are disclosed.

Example 3 *To illustrate the injection attack, we imagine that the karate club network of of Figure 1 is, in fact, a* virtual *karate club. A person can join the virtual karate club by creating an account on the club website; one can also create friendship links to other accounts. Thus, the graph represents user accounts connected by friendships.*

The adversary performs an injection attack to determine whether there is an edge between Alice and John. First, the adversary creates four new accounts—$\{Ii, Jj, Kk, Ll\}$*—and links*



(a)

(b)

Figure 3: Example of injection attack: (a) injected subgraph $\{Ii, Jj, Kk, Ll\}$ connects to targets A and J; (b) Naive anonymization of the graph shown in (a). The injected subgraph is unique and can be re-identified.

them together in a distinctive pattern. Then the injected subgraph is connected to the targets, Alice and John. Figure 3a shows the injected subgraph.

When the anonymized network is published (Figure 3b), the adversary can re-identify the injected subgraph as the subgraph of nodes $\{17, 19, 22, 37\}$ *because it is the only 4-clique in the graph. In turn, this allows the adversary to re-identify Alice and John as nodes* $\{2, 16\}$*, disclosing the edge between them.*

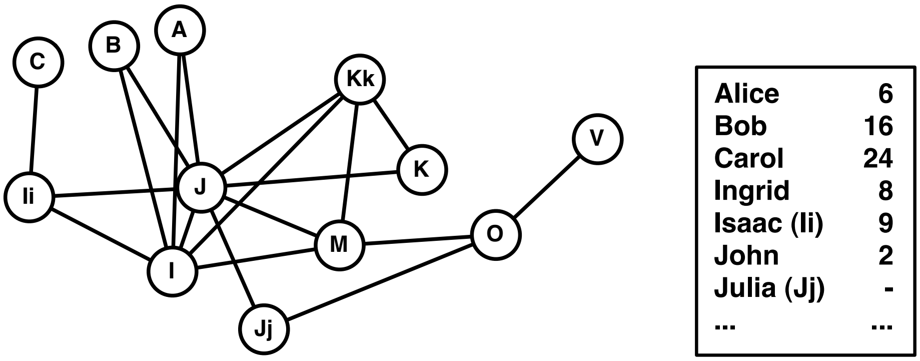
Designing a successful injection attack is challenging. The injected subgraph must be distinctive so it can be re-identified, but the adversary must inject the subgraph *before* seeing the rest of the graph. So the adversary must somehow create a subgraph that is likely to be distinctive regardless of the structure of the graph. In addition, the adversary must create a structure that can be re-identified efficiently, as solving the subgraph isomorphism problem is intractable in general. Finally, while the inserted structure should be distinctive to the adversary, it should not be so distinctive that the data owner can see that the network has been compromised. At the very least, this suggests that the inserted subgraph should be small relative to the size of the network.

Backstrom et al. [2] describe an injection attack that relies on randomness to ensure that the injected subgraph is distinct. They show theoretically that for a graph with n nodes and an injected subgraph with $k = \Theta(\log n)$ nodes, it is possible to construct a subgraph H that is unique with high probability regardless of both the structure of G and how H is connected to G. The argument relies on the fact that the number of subgraphs of size k in G is small relative to the the number of possible subgraphs of size k; thus, by choosing a subgraph uniformly at random, the adversary is unlikely to choose a subgraph that already exists in G. Furthermore, the subgraph is efficiently re-identifiable. This identifiable subgraph can be linked to as many as $\Theta(\log^2 n)$ targets.

Injection attacks are particularly effective for online networks, where it is possible to create new accounts and connections among them. However, executing an injection attack in a network of human contacts can be very difficult as it requires forming a coalition of adversaries who then physically interact with the targets.

2.3.3 Auxiliary-network attack

Most proposed attacks (including the ones described above) assume that (a) the adversary only has information about a small number of targets and (b) the information is precise. An exception is the attack recently proposed by Narayanan and Shmatikov [48]. It is a large-scale attack in which potentially all of the nodes in the graph are re-identified. Further, it relies on information



Alice	6
Bob	16
Carol	24
Ingrid	8
Isaac (Ii)	9
John	2
Julia (Jj)	-
...	...

(a) (b)

Figure 4: Example of an auxiliary-network attack: (a) Auxiliary network of the bridge club, which overlaps partially with the karate club of Figure 1; (b) Adversary's proposed mapping between the bridge club and the nodes in the anonymized karate club of Figure 2.

that may be noisy and imprecise, thus the adversary must attempt to re-identify the targets using approximate knowledge.

The basis of the attack is that the adversary has access to an auxiliary network whose membership overlaps with the anonymized network. For example, suppose an online social network site decides to publish an anonymized version of its network. Often a single person has accounts on multiple sites. Furthermore, some sites are public and it is possible for the adversary to obtain a copy of a public network through crawling [47]. The adversary could use knowledge of this auxiliary network to re-identify nodes in the anonymized network. An auxiliary-network attack can lead to breaches of privacy if for instance the anonymized network includes sensitive attributes and/or additional edges that are not present in the auxiliary network.

Example 4 *Suppose an adversary wants to re-identify nodes in the anonymized karate club network (Figure 2). While the adversary lacks access to the private karate club network (Figure 1), the adversary does have knowledge of the social network of the bridge club, shown in Figure 4a. Some individuals, specifically* $\{A, B, C, I, J, M, V\}$*, appear in both networks. Further, the friendships among these nodes in the bridge club are similar to those of the karate club. However, they are not identical: e.g., Carol is connected to John in the karate club but not the bridge club. Figure 4b shows a possible mapping between the auxiliary bridge network and the anonymized network, where "-" indicates that Julia is not mapped to any node in the anonymized karate club.*

An attack based on auxiliary network raises interesting challenges for the adversary. One such challenge is addressing the approximate nature of the information source. As Example 4 suggests, the networks are likely to have different sets of entities, plus, even where the entities overlap, the relationships in the anonymized network may differ from the relationships in the auxiliary network. This requires a more flexible matching strategy than when the knowledge is assumed to be precise. It also requires some way of scoring or ranking an inexact match.

A related challenge is the complexity of the attack. As discussed previously, a subgraph matching attack is in general NP-Hard. Earlier attacks circumvent the problem complexity by either using efficiently computable structural signatures (the degree signatures) or by considering small subgraphs with a highly constrained structure (the injection attack). With an auxiliary network attack, the subgraph can be arbitrarily large. Further, the matching is not restricted to exact matches.

Narayanan and Shmatikov [48] propose a heuristic, two-stage algorithm for auxiliary-network attacks. First, a small set of "seed" nodes are re-identified in the anonymized network. The assumption is that seeds can be re-identified using, for example, the injection attack of Backstrom et al. [2]. Importantly, the seeds must also appear in the auxiliary network, thereby giving the adversary a partial mapping between the auxiliary network and the anonymized network. In the second stage, called *propagation*, the partial mapping is iteratively extended. At each step, an arbitrarily chosen unmapped node u is mapped to a node v in the anonymized network, based on how many neighbors of u have been mapped to neighbors of v.

To help mitigate the uncertainty in the adversary's knowledge, Narayanan and Shmatikov [48] propose various heuristic strategies, such as early termination, revisiting mapped nodes and reverse mapping. However, the final output is a single mapping and does not reflect the adversary's uncertainty about its accuracy.

2.4 Attack effectiveness

The effectiveness of these proposed attacks has been demonstrated on real networks. Below, we highlight some of the results.

2.4.1 Effectiveness of degree signature attacks

Hay et al. [27, 26] simulate degree signature attacks on three real-world datasets: HepTh, a dataset of co-authorship relations among high-energy physicists; Enron, a network derived from email communication among Enron employees; and NetTrace, a bipartite graph derived from an IP packet trace. For each graph, a separate attack is simulated for each node and each degree signature $\mathcal{H}_i$ for $i = 1, \ldots, 4$.

Figure 5 summarizes the results, showing for what percentage of nodes the attack succeeds. For each degree signature $\mathcal{H}_i$ along the horizontal axis, the nodes are categorized based on the size of their candidate set under that $\mathcal{H}_i$. For example, the black region shows the percentage of nodes that have a candidate set of size 1 (i.e., are re-identified) under the specified $\mathcal{H}_i$. For those nodes not uniquely re-identified, the other shades of gray give an indication of their re-identification risk. For example, the lightest gray region corresponds to nodes that have a candidate set of size 21 or larger, and are therefore at relatively low risk of re-identification.

Overall, we observe that the vulnerability to attack varies significantly across different datasets. However, across all datasets, the most significant change in re-identification is from $\mathcal{H}_1$ to $\mathcal{H}_2$, illustrating the increased power of adversaries that can explore beyond the target's immediate neighborhood. Re-identification tends to stabilize after $\mathcal{H}_3$—more information in the form of $\mathcal{H}_4$ does not lead to an observable increase in re-identification in any dataset. Finally, even though many nodes are re-identified, a substantial number of nodes are *not* uniquely identified even with $\mathcal{H}_4$ knowledge.

In addition, to simulating attacks on real networks, Hay et al. [26] also study random graphs to gain insight into what properties of a graph make it more or less vulnerable to re-identification attacks.

2.4.2 Effectiveness of injection attacks

Backstrom et al. [2] evaluate their injection attack on a real network, finding that in practice the attack can be successful even when the injected subgraph is smaller than required by their worst-case theoretical analysis. On a 4.4 million-node graph derived from LiveJournal, an online blogging and social network site, attacks succeed over 90% of the time with as few as $k = 7$ injected nodes. A successful attack re-identifies the injected subgraph along with roughly 70 targets, disclosing the presence or absence of the $\binom{70}{2}$ possible edges among the targets.

2.4.3 Effectiveness of auxiliary-network attacks

Narayanan and Shmatikov [48] demonstrate their auxiliary-network attack using graphs derived from two online social networks, Twitter (224K nodes) and Flickr (3.3M nodes). Since both of these networks are in the public domain, the authors simulate an attack by naively anonymizing

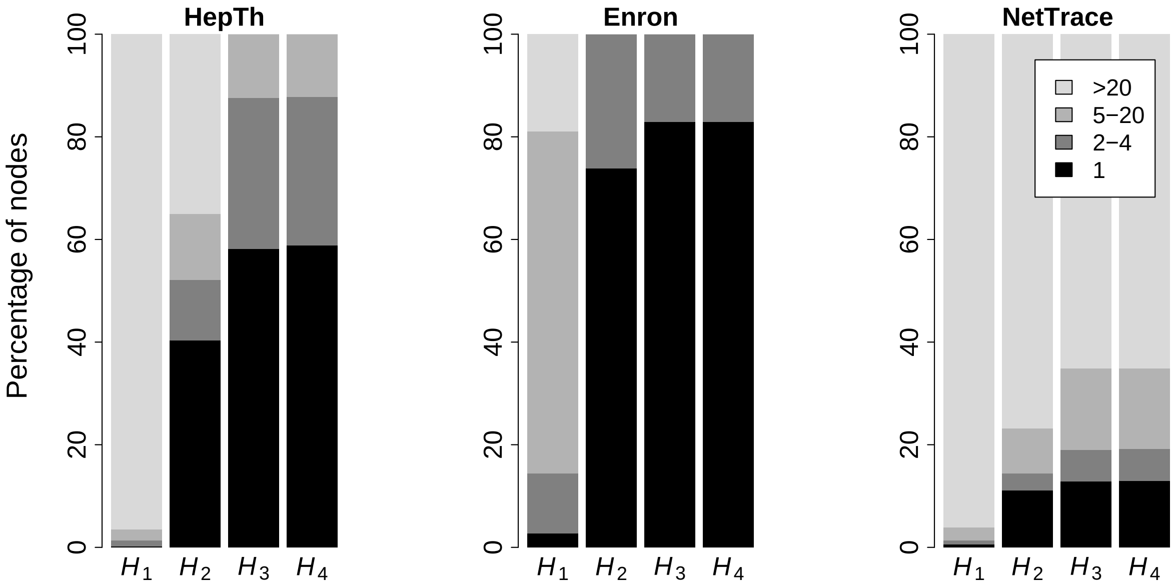


Figure 5: Effectiveness of degree signature attacks across three network datasets. For each node in a given network, we simulate an attack and measure the size of the resulting candidate set. The attack is simulated for increasingly refined degree signatures $\mathcal{H}_i$ from $i = 1, \ldots, 4$ (horizontal axis). Nodes are partitioned based on the size of their candidate set (vertical axis). The size of the candidate set is indicated by color: 1 (black), $2 \ldots 4$ (dark gray), $5 \ldots 20$ (gray), $21 \ldots$ (light gray).

the Twitter graph, using the Flickr graph as the auxiliary network. A total of 150 nodes (less than 0.1% of the anonymized graph) were randomly selected as seeds.

To evaluate the attack, the authors must establish ground truth—the true mapping between nodes in Twitter and nodes in Flickr. This is done by matching the identifying attributes, such as username and geographic location. These attributes are available in both networks, but are not used in the attack, only to establish ground truth. Matching on identifying attributes produces a mapping of roughly 27K nodes, about 12% of Twitter users.

To measure the success of the attack, the authors argue that simply reporting the fraction of re-identified nodes is a "meaningless" metric since many nodes cannot be re-identified by graph structure. Instead, they propose a measure that gives more weight to correctly re-identifying nodes that are "central" in the graph. The *success rate* of an attack is defined as:

$$\frac{\sum_{v \in \mathcal{V}} \mathbf{I}\left[\mu(v) = \mu_{\mathcal{A}}(v)\right] \lambda(v)}{\sum_{v \in \mathcal{V}} \lambda(v)}$$

where $\mathcal{V}$ is the set of nodes that appear in both the auxiliary network and the anonymized network, μ is the ground-truth mapping, $\mu_{\mathcal{A}}$ is the adversary's mapping, and λ is the measure of centrality (the authors use degree).

Using this weighted measure of re-identification success, the attack has a success rate of over 30%. Furthermore, they report that many of the incorrectly mapped entities were mapped to a node that was close, either graphically or geographically, to the true node.

3 Algorithms for Private Data Publication

In this section, we review data publishing techniques for graphs. The goal of data publishing is to construct a graph or graph-like object that resists attacks and can be studied accurately by analysts. As discussed above, simply removing identifiers fails to provide privacy protection since an adversary can use background knowledge to re-identify target individuals. All techniques begin by removing identifiers, and then apply further transformations prior to publication so that the adversary's ability to re-identify nodes or infer edges is greatly diminished. The transformed graph should protect privacy and be useful, and the transformation process should be efficient to execute.

The graph transformations proposed in the literature to date can be categorized as *directed alteration*, *generalization*, and *random alteration*. With directed alteration, the graph structure is altered, using operations such as edge insertions, to create common structural patterns. Nodes in the output graph are more likely to look more similar to one another, but the graph may be missing data or contain spurious information. With generalization, the structure of the graph is generalized at a granularity that is coarse enough to provide some privacy but fine enough to reveal the essential features of the network's topology. Most approaches to generalization are based on clustering nodes into groups and then describing the graph at the group level. Finally, with random alteration, the graph is altered stochastically, through random edge additions and deletions. Structural patterns in the original graph are disguised by the random alteration.

The transformations are applied to produce an output graph satisfying some privacy or anonymity criterion. Many techniques measure privacy in terms of the transformed graph's resistance to subgraph matching attacks and are variants of k-anonymity [58, 57, 61]. Essentially, they ensure that, for given assumptions about an adversary's external information, any target node will have at least k matches in the transformed graph. This uncertainty hinders adversary's ability to infer sensitive information.

Of course, the transformed graph should be useful to analysts. Each approach defines (sometimes implicitly) a measure of utility which is typically used to guide the transformation process. Utility is an important concern in existing work, but is typically secondary to privacy. While privacy is often guaranteed—for any input, an algorithm's output will satisfy the privacy condition—utility is typically only evaluated empirically and worst-case analysis is not considered.

Finally, the techniques presented below differ on the kind of network data considered. Some allow attributes on nodes or focus on limited graph structures, such as bipartite graphs. As a default case, we assume a graph without attributes and therefore an adversary with only structural knowledge. Differences are noted below.

3.1 Directed alteration of networks

One of the findings from Section 2 was that nodes can be distinguished by the local structure around them, making them vulnerable to re-identification in an naively anonymized graph. To counter the threat of re-identification, directed alteration techniques insert edges to make the nodes more structurally uniform. Uniformity ensures that when an adversary attacks the anonymized network, many nodes will match the target's structural pattern, hence preventing the adversary from uniquely identifying the target.

3.1.1 Degree anonymity

The algorithm developed by Liu and Terzi [43] alters the graph to resist matching attacks by an adversary with knowledge of node degree. Edges are inserted into the graph until every node has the same degree as at least $k-1$ other nodes—thus, the graph can be considered k-anonymous with respect to degree. This privacy condition can be satisfied trivially by inserting edges until the graph is complete, but this destroys the utility of the data. To preserve utility, the objective of the algorithm is to find the minimal set of edge insertions to render the graph k-anonymous with respect to degree.

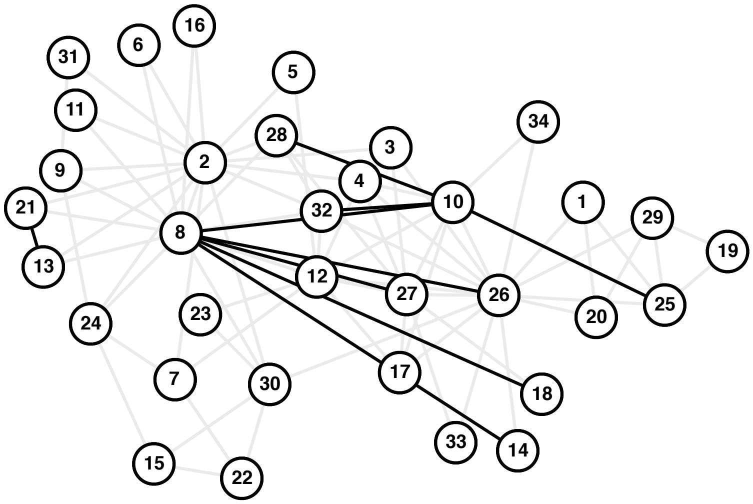


Figure 6: Examples of directed alteration applied to the anonymized karate club network of Figure 2. The original edges are shown in light gray. With directed alteration, edges are inserted until each degree occurs at least $k = 3$ times (inserted edges in solid black).

Example 5 *Figure 6 gives an example of a graph that has been altered to ensure the graph is* 3*-anonymous with respect to degree.*

While this privacy condition may not protect against more powerful adversaries (whose knowledge extends beyond node degree), this problem formulation satisfies another important criterion: scalability. A key insight is that the privacy condition can be assessed using only the graph's *degree sequence*, motivating a two-stage approach: first find the minimum change to the degree sequence to satisfy the privacy condition; then alter the graph to match the new degree sequence. This decoupling leads to a simple, efficient, and scalable solution. The first stage can be computed using an $O(kn)$ dynamic program over the degree sequence. In addition to being efficient, the output sequence has "optimal" utility in the sense that it corresponds to the minimum change necessary to meet the privacy condition.

However, some challenges remain in completing the second stage of altering the graph to match the k-anonymous sequence. First, it may be impossible to realize a graph whose degree sequence matches the k-anonymous sequence. Degree sequences must obey certain constraints (e.g., the sum of the degrees must be even) and the first stage may produce a sequence that is not realizable. Second, even if the sequence is realizable, the second stage requires that it is possible to realize the degree sequence by inserting edges to the original graph. This may not be possible (unless one allows self-loops or multiple edges between a pair of nodes). To address these challenges, Liu and Terzi draw on existing graph theoretic results and devise some heuristic strategies that are efficient and appear to work well on realistic graphs. The algorithm has a worst-case running time that is quadratic in the size of graph, but in practice, run-time is closer to linear. Finally, they also present an extension that alters the graph using edge deletions as well as insertions.

3.1.2 Neighborhood anonymity

Zhou and Pei [69], in some of the earliest work in this area, propose a similar problem formulation as Liu and Terzi [43] but with a stronger privacy condition. The condition requires that for each node in the graph, its *neighborhood*—the subgraph induced by the node and its neighbors—is isomorphic with at least $k-1$ other neighborhoods. Any graph satisfying this condition will also satisfy the condition of Liu and Terzi [43] because if two nodes have isomorphic neighborhoods, then they must have equal degrees. Another difference is that the data model includes labels on the nodes, which must also be anonymized.

Zhou and Pei seek to create isomorphic neighborhoods by inserting edges into graph. Their utility objective, similar to Liu and Terzi [43], is to minimize the number of edge insertions. They show that the problem of determining the minimal set of edge insertions that satisfy the privacy condition is NP-Hard.

Since computing the optimal solution is intractable, Zhou and Pei give a greedy algorithm that iteratively anonymizes the nodes in batches of size k. In each round, a seed node is chosen from the remaining un-anonymized nodes, then $k-1$ nodes are chosen greedily based on their structural similarity to the seed node. Edges are inserted until the batch of neighborhoods becomes isomorphic. To facilitate comparisons between neighborhoods, existing graph coding techniques, called minimum depth-first-search (DFS) codes, are employed. Minimum DFS-codes have the property that graphs are isomorphic if and only if they have matching minimum DFS-codes [64]. While computing the minimum code can require time that is super-polynomial in the size of the neighborhood, the average-case complexity appears to be much lower: the algorithm can anonymize graphs with 25K nodes in under 10 minutes.

With this iterative approach, a subtle issue can arise when an edge is inserted that connects to a node that has already been marked as anonymized. This causes the marked node to no longer be isomorphic with the other nodes in its batch. The algorithm addresses this by unmarking these nodes and returning them to the list of un-anonymized nodes. In the worst case, the algorithm returns a complete graph. This highlights a challenge with anonymizing graph data: the alterations that modify the local structure around a node can have unintended consequences on the global structure of the network.

The problem formulations of Zhou and Pei [69] and Liu and Terzi [43] are quite similar, and it is not clear which approach provides better utility. Since Zhou and Pei impose a more stringent privacy condition, one would expect more edge insertions, and thus perhaps lower utility. In general, the literature contains few direct comparisons of graph anonymization techniques. A thorough evaluation of their privacy-utility tradeoffs would be valuable to data owners who must choose among these competing techniques.

The empirical results of Liu and Terzi [43] provide some insight into how these techniques affect graph structure. First, edge insertion introduces bias into some common measures: degrees increase and average path length becomes shorter. The magnitude of the bias depends on the topology of the graph. For instance, Liu and Terzi evaluate their approach on power-law graphs—random graphs with power-law degree distributions—and on small-world graphs—random graphs with binomial degree distributions. For the same level of privacy, more edges must be inserted into power-law graphs than small-world graphs because the degree distribution of a power-law graph contains some large, outlying degrees. For example, at $k = 15$, the average degree increases by about 1 for a power-law graph but by less than 0.2 for a small-world graph [43].

Finally, the utility objective of both algorithms is to minimize the number of edge insertions. This assumes that the utility cost of an edge insertion is uniform across all edges. In fact, given that many graphs exhibit “community” structure—relatively dense subgraphs that are sparsely interconnected—the insertion of some edges (say, within community edges) would distort the graph topology “less” than others (say, between community edges). The generalization-based approaches (described next) present alternative utility objectives that explicitly attempt to minimize some measure of information loss.

3.2 Network Generalization

The goal of generalization techniques is to obscure the details of local structure while preserving global properties of the graph. Rather than create structural uniformity through alteration, node identity is hidden and local structure is summarized.

3.2.1 Anonymity through clustering

Hay et al. [26] introduce a network anonymization algorithm that is based on *generalizing* the network to resist re-identification attacks. The algorithm outputs a generalized graph, which represents a coarse summary of the original input graph. Since a similar idea of graph generalization is also considered elsewhere [10, 5, 67], we now describe it in some detail.

The generalized graph, denoted $\mathcal{G}$, is based on a partition of the nodes into disjoint sets. Each subset can be thought of as a *super-node* since it contains nodes from G but is itself a node in $\mathcal{G}$. The edges of $\mathcal{G}$ are called *super-edges*, and there is a super-edge between two super-nodes if there is at least one edge in the original graph between their corresponding sets of nodes. If two nodes in the same super-node share an edge in the original graph, then their super-node has a self-loop in $\mathcal{G}$. Finally, the super-edges are assigned non-negative weights which report the number of edges in the original graph that exist within and between the partitions.

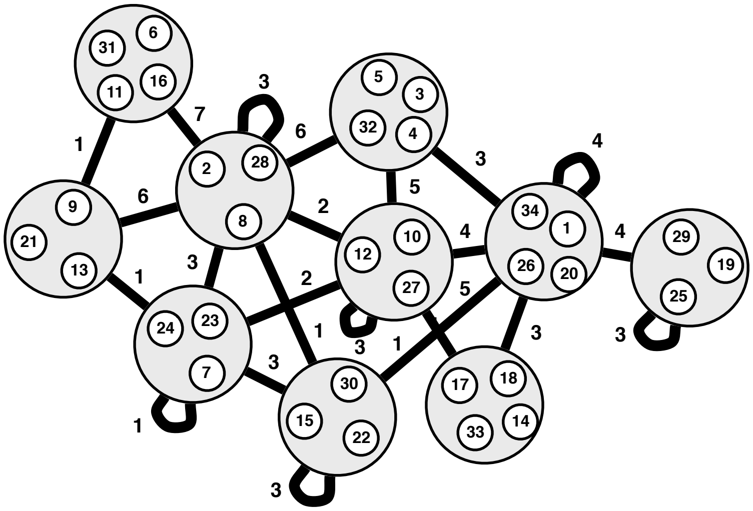


Figure 7: Examples of generalization applied to the anonymized karate club network of Figure 2. With generalization, nodes are grouped into super-nodes until each group contains at least $k = 3$ nodes (super-nodes in gray; super-edges in black).

Example 6 *A generalized graph is shown Figure 7. The gray enclosing circles denote super-nodes and the thick black edges denote super-edges. The super-edge between super-node* $\{1, 20, 26, 34\}$ *and super-node* $\{19, 25, 29\}$ *has a weight of 4 because there are 4 edges between these two sets of nodes in the original graph.*

A generalized graph summarizes the structure of the original graph, but the accuracy of the summary depends on the choice of super-nodes. At one extreme, if the generalized graph contains a single super-node, then the only facts revealed about the input graph are its size (number of nodes) and density (number of edges). This provides considerable privacy but very low utility. At the other extreme, if there is a one-to-one correspondence between nodes and super-nodes, then the generalized graph encodes the original graph, providing perfect utility but no additional privacy. A generalized graph defines a set of graphs that are consistent with its summary description. The graphs in this set are called *possible worlds*.

Hay et al. [26] specify a privacy condition that each partition (super-node) must contain at least k nodes. Since nothing in the published output allows an adversary to distinguish between two nodes in the same partition, the output is k-anonymous with respect to *any* knowledge of graph structure. Therefore, this privacy condition is stronger than the conditions of the directed-alteration approaches, which assume bounds on adversary knowledge.

Example 7 *In Figure 7, the generalized graph has a minimum group size of $k = 3$, thus ensuring that an adversary cannot re-identify a node beyond 3 possible candidates.*

To balance the competing goals of privacy and utility, Hay et al. [26] introduce an algorithm that outputs a generalized graph $\mathcal{G}$ such that each super-node contains at least k nodes and the number of possible worlds is minimized. The number of possible worlds is a measure of the utility of the output: more possible worlds implies greater uncertainty about the input graph. The algorithm uses local search over the exponential space of generalized graphs, applying operations such as merging/splitting super-nodes and moving a node between super-nodes. The output is a local optimum in the search space. The algorithm appears to scale roughly quadratically with the size of the graph.

In terms of utility, an analyst can study the generalized graph by sampling a graph from the set of possible worlds. This graph is a standard graph (not a summarized graph) and can be analyzed using standard techniques. The experiments of Hay et al. [26] suggest that graph generalization appears to preserve some important global properties such as network resiliency and the distribution of path-lengths. However, some local properties, such as clustering coefficient can be substantially diminished. This is because within a super-node, the sampled graph structure is a random graph, and random graphs tend to have low clustering coefficients.

Campan and Truta [5] Concurrently with the above work, Campan and Truta [5] devise a similar approach. In terms of graph structure, their *masked social network* appears to be equivalent to the generalized graph described above. Campan and Truta also include identifying attributes on the nodes. These attributes are generalized using conventional techniques from tabular data anonymization with the result that each super-node is assigned a vector of generalized attribute values. Aside from the additional condition on attribute values, the privacy condition is the same as above. Thus, the k-anonymous masked social network is k-anonymous with respect to knowledge of attributes and structure.

Campan and Truta formulate a slightly different utility objective than Hay et al. [26], with the main differences having to do with the inclusion of attributes. Subject to the privacy condition, the objective is to minimize a measure of *information loss* that combines both attribute and structural information loss. Campan and Truta adopt a measure of attribute information loss from existing work in tabular data anonymization. For structural information loss, they use a measure that is approximately equivalent to minimizing the number of possible worlds. These two loss functions are combined linearly, with weights that can be chosen by the user. So if the attribute information loss is given zero weight, this formulation appears to be roughly equivalent to the formulation of Hay et al. [26].

To generalize the graph, Campan and Truta apply an iterative, greedy algorithm that anonymizes nodes in batches of k, similar to the algorithm of Zhou and Pei [69]. At each iteration, a new super-node is created by choosing a seed node. Then $k - 1$ other nodes are added to the super-node based on their similarity (in terms of attributes and neighborhood) to the super-node.

3.2.2 Safe groupings

Cormode et al. [10] also present a generalization algorithm for anonymizing graph data; however, the graphs considered have different semantics than the graphs discussed above. The data model is a bipartite graph (V, W, E) where V and W correspond to two distinct types of entities and E corresponds to the associations between them. (In addition, there is an attribute table associated with each of V and W.) For example, V could be customers at a pharmacy and W could be medications, and the edges in E connect each customer with the medications they have have purchased. In this setting, the private information is the association between customer and product. Their generalization algorithm prevents the disclosure of edges (e.g., who bought what medication). This is a departure from the above approaches which focus on preventing node re-identification.

The approach taken by Cormode et al. assumes that the adversary will use knowledge of attributes rather than graph structure in a matching attack. As a result, it is not considered a privacy risk to publish the exact graph structure. This is also a departure from the adversary models considered above. To prevent matching attacks based on attributes, their technique masks the mapping between nodes in the graph and real-world entities (with their associated attribute vectors). This is done by partitioning the nodes, and the corresponding entities, into groups. Within a group, the mapping between node and entity is secret. The most basic privacy definition used is a (k, ℓ)-grouping, essentially requiring that the minimum group size is k and ℓ for V and W, respectively. For a graph with a (k, ℓ)-grouping, the set of possible worlds corresponds to every possible bijective mapping of nodes and entities in each group.

Of course, a (k, ℓ)-grouping may not necessarily prevent edge disclosure. For example, if every customer in a group purchased the same group of medications, then the subgraph between this customer group and the medication group would be complete. Cormode et al. propose a notion of a *safe* grouping, which requires that no two nodes in the same group share a neighbor—in other words, the edges in the subgraph between any pair of groups must form a matching (i.e., edge-independent set). This definition necessarily imposes some constraints on the graph—it must be sparse for such a grouping to exist. Cormode et al. demonstrate the advantage of a safe grouping and prove that with a safe grouping, associations cannot be inferred using the published data alone. Further, they present a greedy algorithm for anonymizing the graph using groupings. While the algorithm is not guaranteed to find a safe-grouping (even if one exists), in practice it appears to work well for small values of k, ℓ.

3.3 Randomly altering networks

In contrast with the above approaches—which thwart the adversary by systematically transforming the graph to make it more general or structurally uniform—random-alteration approaches alter the graph structure randomly, thereby making the adversary uncertain about the true structure of the graph. It is an appealing approach that has been successfully applied to protect privacy in tabular data [4, 17, 56]. The algorithms developed for tabular data are quite simple, yet provide strong privacy guarantees and good utility. However, extending these techniques to network data appears challenging. We review one technique for tabular data and then discuss the challenges that arise with network data.

For tabular data anonymization, Rastogi et al. [56] give a simple randomized algorithm in which each record from the table is added to the output table with probability less than one. In addition, synthetic records are randomly sampled from the table's domain and added to the output. This ensures privacy because the adversary cannot determine whether a particular record in the output is genuine or synthetic. It also provides utility: one can estimate answers to any counting query—the number of records satisfying a predicate—with reasonably high accuracy (error scales with the square-root of the domain size). Note that computing query answers requires some post-processing: simply evaluating the query on the randomized output will give an inaccurate answer because the output contains many synthetic records. Instead, one must use statistical inference techniques to derive an estimate.

While such random alteration preserves utility for tabular data, a number of challenges arise in devising random-alteration techniques for network data. The first problem is choosing a natural randomization operation. A direct application of the techniques of Rastogi et al. [56] would randomly remove some edges and insert others, resulting in an output graph that is a mixture of the input graph and a random graph. This limits the disclosure of individual edges, but the utility of the output is also limited. With network data, the queries of interest are not typically counting queries over the set of edges but instead often involve computing paths (self-joins on the edge table). Since the insertion or removal of a single edge can make or break many paths, accurate estimates are difficult to achieve. In fact, for a query such as the number of triangles (cycles of length 3), one can construct worst-case input graphs where the expected

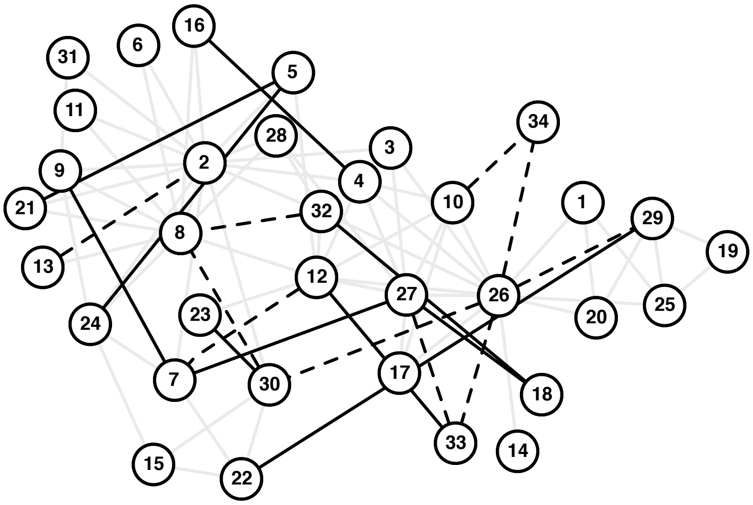


Figure 8: Examples of random alteration applied to the anonymized karate club network of Figure 2. The original edges are shown in light gray. With random alteration, edges are randomly chosen and rewired until $m = 10$ edges have been altered (deleted edges in dashed black; inserted edges in solid black).

error is linear in the size of graph [55].

3.3.1 Uniform randomization

There have been some initial efforts to apply random alterations to network data. Hay et al. [26] considered randomly permuting some fraction of the network's edges.

Example 8 *An example of random alteration is shown in Figure 8. The network is altered by deleting $m = 10$ edges uniformly at random and then inserting m edges uniformly at random.*

Hay et al. show that this was effective at limiting an adversary's ability to re-identify a target based on node degree (see Ying and Wu [65] for a theoretical analysis of privacy). However, the graph structure was changed considerably: a 10% change in the edge structure resulted in a roughly 33% change in the value of some important graph metrics. Although the analyst may be able to reduce the error through statistical post-processing, the results suggest that the gain in privacy is offset by a substantial loss in utility.

3.3.2 Spectrum-preserving randomization

To address the loss of utility, Ying and Wu [65] considered a more complex randomization strategy that was guided by the graph structure, choosing a random alteration that preserves key properties of the network. The technique is based on the observation that many important network properties are related to the graph's spectrum—i.e., the set of eigenvalues of the graph's adjacency matrix or other matrices derived from it. Thus, they develop a random-alteration algorithm where edges are randomly added and deleted, but the random choice is guided based on how the change affects the graph's spectrum. They show that the utility of the randomly altered network—measured both in terms of common metrics and spectral properties—is much improved. However, they do not assess what impact spectrum-based randomization has on privacy. The protection must necessarily be weaker than pure randomization: the noise is influenced by the structure of the graph which means the adversary may be able to use his knowledge of graph structure to infer likely edge swaps. It is unclear how much this improves the adversary's ability to breach privacy.

4 Algorithms for Private Query Answering

The above approaches protect privacy by transforming the data prior to publication. The published output is an altered or a generalized graph. In this section, we review an alternative approach, in which a graph is never published. Instead, the only published information consists of the answers to specific queries. To ensure privacy, often the exact answer is not revealed; instead, the data owner publishes an approximate answer, perhaps by adding random noise to the true answer.

The distinction between these two approaches can be blurred, since a "query" could, in principle, be any request for data. Typically, however, the query is a summary statistic, such as the clustering coefficient of the graph, rather than a request for raw data. An advantage with the query answering approach is that the distortion can be tailored to the specific query, allowing for better utility. Intuitively, statistics that summarize the data at a coarse granularity can be answered more accurately than a query that asks about specific records.

Much of the recent work in query answering techniques has used a notion of privacy called *differential privacy* [13, 15]. It is an extremely robust notion of privacy that guarantees protection even against extremely powerful adversaries. Despite this strong guarantee, it is possible to satisfy differential privacy and yet provide accurate answers in a variety of practical applications.

While most work in differential privacy focuses on the query-answering setting, the privacy definition can also be applied to the data publishing setting as well. In fact, some differentially private data publishing algorithms have been proposed recently [4, 16, 56], but they are designed for tabular data. Developing differentially private algorithms for publishing network data is an interesting direction for future work.

While differential privacy has been the basis of an active area of data privacy research, most work assumes the data is tabular. For a survey of the tabular data results, see Dwork [13]. In this section, we highlight some of the results oriented specifically to network analysis. After formally defining differential privacy, we discuss how the definition can be adapted to network data. Then, we review the differentially private algorithm of Dwork et al. [15] that can be used to answer arbitrary queries. Finally, we describe some positive results for network analyses as well as some the remaining challenges.

4.1 Differential privacy

The formal definition of differential privacy assumes that the database consists of a single table of records where each record describes an individual's private information. We review the formal definition and then discuss how it can be adapted to network data.

Differential privacy is a property of an algorithm. Informally, it requires that an algorithm be insensitive to small changes in the input, such as the addition or removal a single record. The formal definition uses the concept of neighboring databases: two databases I and I' are *neighbors* if they differ in at most one record, i.e., $|(I - I') \cup (I' - I)| = 1$. Let $nbrs(I)$ to denote the neighbors of I. Differential privacy is defined as follows:

Definition 4.1 (ϵ-differential privacy) *An algorithm A is ϵ-differentially private if for all instances I, any $I' \in nbrs(I)$, and any subset of outputs $S \subseteq Range(A)$, the following holds:*

$$Pr[A(I) \in S] \leq \exp(\epsilon) \times Pr[A(I') \in S]$$

where probability Pr is over the randomness of A.

An example illustrates why a differentially private algorithm protects privacy. Suppose a trusted party is conducting a poll. Once responses have been collected, the pollster plans to analyze the responses using a differentially private algorithm and publish the output. The individual's concern is that if they respond, the output will reveal something about them personally and thus violate their privacy. The above definition assuages this concern because whether the individual opts-in or opts-out of the survey, the probability of a particular output is almost the same (differing by a factor of at most $\exp(\epsilon)$). Clearly, any observed output cannot reveal much about their particular record if that output could occur (with a similar probability) even when their record is excluded from the database.

This is a different notion of privacy than the ones considered in Section 3, which define privacy in terms of protection against an adversary with specific knowledge. The differential privacy guarantee is a condition on the algorithm, and thus it is independent of adversary knowledge. In fact, it can be shown that even if an attacker knows every record in the database except one, the adversary cannot use the output of a differentially private algorithm to infer much about the remaining record (more formally, the difference between the adversary's prior and posterior beliefs is bounded [15, 55]). Thus, it affords extremely strong privacy protection.

Differential privacy has been defined inconsistently in the literature, where neighboring databases are sometimes defined in terms of Hamming distance [14, 15] and sometimes defined, as it is above, in terms of symmetric difference [13, 45]. The technical implications are outside the scope of the present discussion, but suffice it to say that the definition based on symmetric difference is more general.

4.2 Differential privacy for networks

The above definition is predicated on a data model where an individual's private information is encapsulated within a single record. Thus, neighboring databases differ by the addition or removal of a person's private information. With network data, which is primarily about relationships among individuals, the correspondence between private data and database records is less clear. To adapt the definition to graphs, one must appropriately define the concept of neighboring graphs.

In the literature, neighboring graphs are assumed to differ by a single edge. Under this approach, a differentially private algorithm essentially protects against edge disclosure. This is a similar privacy objective as some of the data publishing techniques described in Section 3. However, differential privacy places no limiting assumptions on the input or on adversary knowledge. An adversary with knowledge of all of the edges in the graph except one, cannot use the differentially private algorithm to infer the presence or absence of the unknown target edge [55].

While individual edges are protected, this adaptation of differential privacy for graphs does not capture the same "opt-in/opt-out" notion of privacy that was described in the polling example (Section 4.1). It may still be possible to learn specific facts about a node. For example, this adaptation admits the disclosure of aggregate properties of sets of edges—e.g., a node's *degree* can be approximately revealed. Further, some networks, such as the example in Figure 1, also have attributes on the nodes. There may be alternative ways to adapt the definition to protect both edges and node attributes. Understanding the privacy-utility tradeoffs of these alternatives is an important area for future work.

4.3 Algorithm for differentially private query answering

Dwork et al. [15] present a differentially private algorithm for answering any query or sequence of queries. We illustrate the approach using the following example. Consider an analyst who wants to learn about the degrees of nodes. Let $\mathbf{D}[u]$ denote the query that returns the degree of node u if $u \in V$, and otherwise returns -1. Let $U = u_1, \ldots, u_n$ be a sequence of node identifiers. The query sequence $\mathbf{D}[U]$ returns a vector corresponding to the degrees of the nodes in U—i.e., $\mathbf{D}[U] = \langle \mathbf{D}[u_1], \ldots, \mathbf{D}[u_n] \rangle$.

Example 9 *Given the network from Figure 1, the query* $\mathbf{D}[Alice]$ *returns 2. The query sequence* $\mathbf{D}[Alice, Ed, Frances]$ *returns* $\langle 2, 4, -1 \rangle$ *since 2 and 4 correspond to the the degrees of Alice and Ed respectively and -1 reveals that Frances is not in the network.*

The algorithm of Dwork et al. [15] works by adding random noise to the answer, where the amount of noise depends on the query's *sensitivity.* Sensitivity is a worst-case notion that measures the maximum change in the query answer between any two neighboring databases.

Definition 4.2 (Sensitivity) *Let* $\mathbf{Q}$ *be a sequence of queries where each query returns a number in* $\mathbb{R}$*. The sensitivity of* $\mathbf{Q}$*, denoted* $S_{\mathbf{Q}}$*, is defined as*

$$S_{\mathbf{Q}} = \max_{I,I' \in nbrs(I)} \left\| \mathbf{Q}(I) - \mathbf{Q}(I') \right\|_1 .$$

We illustrate this concept by computing the sensitivity of some degree queries.

Example 10 *The sensitivity of query* $\mathbf{D}[Alice]$ *is 1: Neighboring graphs differ by exactly one edge and there exist pairs of neighboring graphs where that edge is incident to Alice. So for those pairs of neighboring graphs, the query answer differs by 1.*

The sensitivity of the query sequence $\langle \mathbf{D}[Alice], \mathbf{D}[Ed], \mathbf{D}[Frances] \rangle$ *is 2 because neighboring graphs can differ by an edge that connects two of these individuals, causing each of their degrees to differ by one.*

Dwork et al. [15] have shown that the following algorithm is ϵ-differentially private. Given a query sequence $\mathbf{Q}$, the algorithm first computes the answer to the query on input I and then adds independent random noise to each answer in the sequence. The noise is sampled independently from a Laplace distribution with mean zero and scale $\sigma = S_{\mathbf{Q}}/\epsilon$. The magnitude of the scale controls the amount of noise: as σ increases, the answers become noisier. Thus, the noise in the answer increases with increasing query sensitivity, $S_{\mathbf{Q}}$, or with decreasing ϵ (corresponding to greater privacy).

Example 11 *Let the query* $\mathbf{Q}$ *be* $\mathbf{D}[Alice]$*. To compute the answer to the query under differential privacy, the algorithm first computes the true answer, which is 2. Then it adds Laplace random noise with scale* $\sigma = S_{\mathbf{Q}}/\epsilon$*. Recall that for this query,* $S_{\mathbf{Q}} = 1$*.*

When $\epsilon = 1.0$*, the scale is* $\sigma = S_{\mathbf{Q}}/\epsilon = 1.0$*. With a 95% probability, the noisy answer will lie in the interval* 2 ± 2.995*. However, when* $\epsilon = 0.1$*, the scale becomes* $\sigma = 1/0.1 = 10.0$ *and the 95% probability interval is* 2 ± 29.957*.*

For a fixed query, or query sequence, the above algorithm is simple to implement and it is guaranteed to satisfy a strong privacy guarantee. For the output to be useful, however, the amount of random noise must be small relative to the query answer. Since the noise is determined by the sensitivity of the query, a key question is whether common network analyses have low sensitivity. The next section looks at some specific analyses and the extent to which they can be accurately estimated under differential privacy.

4.4 Network analysis under differential privacy

Enabling accurate analysis of social networks is an often mentioned goal in the differential privacy literature, but relatively few concrete results exist that demonstrate the feasibility of differential privacy for network data. Below we highlight a few results and discuss some of the challenges.

4.4.1 Low sensitivity analyses

The previous section showed that the accuracy of Dwork et al.'s [15] algorithm depends on the query's *sensitivity*, with lower sensitivity yielding greater accuracy. Some analyses of networks can be computed with queries that are low sensitivity. For example, network resiliency can be approximated with a low sensitivity query. The query asks how many edges must be removed until the network becomes, say, disconnected, and it has a sensitivity of one [15]. In addition, for weighted graphs with edge weights in $[0, 1]$, the weight of a minimum edge-cut or a minimum spanning tree are both low-sensitivity queries [15].

However, the fact that an analysis can be computed using a query, or sequence of queries, with low sensitivity does not necessarily imply that the analysis will be accurate under differential privacy. We present a more detailed look at one particular analysis: measuring the degree distribution of a network.

As discussed above, the query that asks for an individual degree, or a sequence of degrees, is a low sensitivity query. However, typically, an analyst is not concerned with individual degrees but with the distribution of degrees. While there are some natural strategies for obtaining the entire degree distribution—such as asking for each node's degree or asking for the number of nodes with a given degree—these approaches require asking many queries, and the total amount of noise grows linearly with the size of the graph. The consequence of asking so many queries is that the error introduced can substantially distort the degree distribution.

Hay et al. [25] give an accurate and efficient algorithm for estimating the degree distribution of a graph. It capitalizes on a recent innovation in differentially private algorithms that has been shown to boost accuracy without sacrificing privacy [28]. The technique performs a post-processing step on the differentially private output, using a set of known constraints to infer a more accurate result. Hay et al. [25] demonstrate that the post-processing step can reduce the error by orders of magnitude and the resulting degree distributions are extremely accurate. Also they show the post-processing requires only linear time and thus it scales to the large social networks commonly analyzed today.

4.4.2 High sensitivity analyses

While these are promising results, open questions remain about the accuracy obtainable for many common network analyses. For some important analyses, the prospects seem poor. Computations such as transitivity, clustering coefficient, centrality, and path-lengths involve joins on the edge table. It is not hard to construct examples showing that the sensitivity of such statistics is extremely high. We cannot hope to guarantee accurate answers for high sensitivity queries under differential privacy. (For a formal statement, see Rastogi et al. [55].)

To address these limitations, some alternative approaches have been proposed. We discuss two approaches for a particular high sensitivity query: counting the number of triangles in a graph.

The query that reports the number of triangles (i.e., cycles of length 3) is an important query in social network analysis and is related to properties such as clustering coefficient. It has a sensitivity of $n-2$ because, in the worst case, a single edge participates in a triangle with each of the remaining $n-2$ nodes. Removing that edge changes the number of triangles by $n-2$.

Nissim et al. [52] give an algorithm for approximating the number of triangles in a graph. It is based on a general technique called *smooth sensitivity*. The motivation is that a query can have high sensitivity because some worst-case inputs yield substantially different answers from their neighboring databases, but typical inputs yield only small changes. A tempting solution is to use the *local sensitivity* of I—i.e., the maximum change between $\mathbf{Q}(I)$ and $\mathbf{Q}(I')$ for any $I' \in nbrs(I)$. However, this can leak information because the local sensitivity itself can change substantially between neighboring instances and thus an approach that uses it directly would fail to satisfy differential privacy. Smooth sensitivity is a upper bound on local sensitivity that varies smoothly over the space of possible databases. Nissim et al. [52] show that adding noise according to the smooth sensitivity satisfies a slightly relaxed definition of differential privacy.

Nissim et al. [52] apply the smooth sensitivity idea to the problem of computing the number of triangles. They show how to efficiently compute the smooth sensitivity for a given graph and also show that random graphs are likely to have low smooth sensitivity.

Rastogi et al. [55] present an alternative approach for estimating the number of triangles. While the algorithm is not differentially private, it does guarantee a natural definition of privacy called *adversarial privacy*. Interestingly, they also characterize the relationship between differential privacy and adversarial privacy by defining the class of adversaries for which a differentially private algorithm is adversarially private. This class includes adversaries that are extremely powerful and arguably unrealistic. By restricting the protection to a weaker (and more realistic) class of adversaries, they are able to accurately estimate the frequency of triangles, as well as other subgraph patterns.

Finally, another potential solution for high sensitivity queries is to avoid them. High sensitivity means that for some networks, the change of a single edge can profoundly alter the query answer. Given that network data is often incomplete and noisy, analysts need measures that are robust to minor perturbations of network structure. In fact, robustness to small perturbations has been proposed as a way of evaluating the significance of the communities found by a community discovery algorithm [30]. The connection between robust statistics and differentially private algorithms has been explored, but existing results are limited to high sensitive queries of tabular data [14].

5 Conclusion and Future Issues

The investigation of the private and accurate analysis of network data is still in its early stages and many challenges remain. New attacks are being discovered and new protection mechanisms are actively being developed. Some of the outstanding challenges include: establishing formal guarantees of utility, devising methods to handle richer data representations, and scaling techniques to large networks.

In general, existing data publication techniques do not provide guarantees of accuracy for specific analyses, only empirical evidence that certain properties are maintained in the output. An analyst forced to study a surrogate data set may be reluctant to trust conclusions drawn without guarantees of utility. In addition, a precise notion of network utility has yet to be defined, and the evaluation of utility has been somewhat ad hoc in existing work.

Further, few of the current privacy mechanisms support the release of attributes on nodes. Privacy protection for attributed networks deserves more study since attributes are crucial to many analyses. In addition, many networks encountered in the real world are derived from time-stamped streams of connections or contacts (e.g., email graphs, network traces, online social networks). Techniques proposed thus far do not support dynamic networks.

As mentioned in the introduction, network data is now collected on very large scales. Networks with over 100 million nodes are being collected and studied. Analysis of such networks can be challenging even in the absence of privacy concerns, as some analyses do not scale well. However, scalability of privacy mechanisms is a significant challenge. Most of the data publication schemes described above were tested on networks with fewer than 100 *thousand* nodes, and do not appear to scale to larger graphs. Unfortunately, the proposed attacks on networks scale better than some of the publication mechanisms. The query answering techniques based on differential privacy have an advantage here, as in most cases they do not add much overhead above the cost of computing the released query.

This review highlights some of the main challenges of protecting privacy while enabling accurate network analysis, but our coverage of this active area of research is admittedly incomplete. Two recent surveys [42, 70] provide additional perspective on this topic. In addition, more work on this topic is forthcoming, including new data publishing techniques by Cormode et al. [9] and Zou [71].

6 Acknowledgments

We thank the reviewers for insightful comments, Cindy Loiselle for editorial assistance, and Don Towsley, Dan Suciu, and Vibhor Rastogi for helpful discussions.

This material is based on research sponsored by the National Science Foundation (NSF CNS 0627642) and by the Air Force Research Laboratory and the Intelligence Advanced Research Projects Activity (IARPA), under agreement number FA8750-07-2-0158. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusion contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and the Intelligence Advanced Research Projects Activity (IARPA), or the U.S. Government.

References

[1] A. Acquisti and R. Gross. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In *Privacy Enhancing Technologies Workshop*, 2006.

[2] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou R3579X? Anonymized social networks, hidden patterns, and structural steganography. In *WWW*, 2007.

[3] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 1999.

[4] A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *STOC*, 2008.

[5] A. Campan and T. M. Truta. A clustering approach for data and structural anonymity in social networks. In *PinKDD*, 2008.

[6] R. F. Cancho and R. V. Sole. Optimization in complex networks. In *ArXiv cond-mat/0111222*, 2001.

[7] B. Carminati, E. Ferrari, and A. Perego. Rule-based access control for social networks. In *Workshop on Reliability in Decentralized Distributed Systems*, 2006.

[8] A. Clauset, C. Moore, and M. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 2008.

[9] G. Cormode, D. Srivastava, S. Bhagat, and B. Krishnamurthy. Class-based graph anonymization for social network data. In *VLDB*, 2009.

[10] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. In *VLDB*, 2008.

[11] D. Corneil and C. Gotlieb. An efficient algorithm for graph isomorphism. *Journal of the ACM*, 1970.

[12] L. da F. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas. Characterization of complex networks: A survey of measurements. *Advances In Physics*, 2007.

[13] C. Dwork. Differential privacy: A survey of results. In *Conference on Theory and Applications of Models of Computation*, 2008.

[14] C. Dwork and J. Lei. Differential privacy and robust statistics. In *STOC*, 2009.

[15] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, 2006.

[16] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *STOC*, 2009.

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

[18] A. Felt and D. Evans. Privacy protection for social networking APIs. In *In Web 2.0 Security and Privacy Workshop*, 2008.

[19] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 1977.

[20] L. Friedland and D. Jensen. Finding tribes: Identifying close-knit individuals from employment patterns. In *KDD*, 2007.

[21] K. Frikken and P. Golle. Private social network analysis: How to assemble pieces of a graph privately. In *WPES*, 2006.

[22] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 2008.

[23] R. Gross and A. Acquisti. Information revelation and privacy in online social networks (the facebook case). In *WPES*, 2005.

[24] M. Handcock, G. Robins, T. Snijders, P. Wang, and P. Pattison. Recent developments in exponential random graph (p*) models for social networks. *Social Networks*, 2006.

[25] M. Hay, C. Li, D. Jensen, and G. Miklau. Accurate estimation of the degree distribution of private networks. Technical report, University of Massachusetts Amherst, 2009.

[26] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. In *VLDB*, 2008.

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

[28] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially private queries through consistency. Technical report, University of Massachusetts Amherst, 2009.

[29] J. He, W. W. Chu, and Z. Liu. Inferring privacy information from social networks. In *ISI*, 2006.

[30] B. Karrer, E. Levina, and M. E. J. Newman. Robustness of community structure in networks. *Physical Review E*, 2008.

[31] J. Kleinberg. Cascading behavior in networks: Algorithmic and economic issues. *Algorithmic Game Theory*, 2007.

[32] A. Klovdahl, J. Potterat, D. Woodhouse, J. Muth, S. Muth, and W. Darrow. Social networks and infectious disease: the Colorado Springs study. *Social science & medicine*, 1994.

[33] A. Korolova, R. Motwani, S. Nabar, and Y. Xu. Link privacy in social networks. In *CIKM*, 2008.

[34] G. Kossinets and D. Watts. Empirical analysis of an evolving social network. *Science*, 2006.

[35] D. Lazer, A. Pentland, L. Adamic, S. Aral, A.-L. Barabasi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. V. Alstyne. Computational social science. *Science*, 2009.

[36] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *KDD*, 2008.

[37] J. Leskovec and C. Faloutsos. Scalable modeling of real graphs using Kronecker multiplication. In *ICML*, 2007.

[38] J. Leskovec and E. Horvitz. Planetary-scale views on a large instant-messaging network. In *WWW*, 2008.

[39] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: Densification laws, shrinking diameters and possible explanations. In *KDD*, 2005.

[40] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM*, 2003.

[41] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Inferring private information using social network data. In *WWW*, 2009.

[42] K. Liu, K. Das, T. Grandison, and H. Kargupta. *Privacy-Preserving Data Analysis on Graphs and Social Networks.* 2008.

[43] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD*, 2008.

[44] M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 2001.

[45] F. McSherry. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *SIGMOD*, 2009.

[46] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 2002.

[47] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *IMC*, 2007.

[48] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *IEEE Symposium on Security and Privacy*, 2009.

[49] M. Newman, S. Forrest, and J. Balthrop. Email networks and the spread of computer viruses. *Physical Review E*, 2002.

[50] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 2004.

[51] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[52] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, 2007.

[53] J.-P. Onnela, J. Saramaki, J. Hyvonen, G. Szabo, D. Lazer, K. Kaski, J. Kertesz, and A.-L. Barabasi. Structure and tie strengths in mobile communication networks. *PNAS*, 2007.

[54] J. Park and A.-L. Barabasi. Distribution of node characteristics in complex networks. *Proceedings of the National Academy of Sciences*, 2007.

[55] V. Rastogi, M. Hay, G. Miklau, and D. Suciu. Relationship privacy: Output perturbation for queries with joins. In *PODS*, 2009.

[56] V. Rastogi, S. Hong, and D. Suciu. The boundary between privacy and utility in data publishing. In *VLDB*, 2007.

[57] P. Samarati. Protecting respondent's privacy in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 2001.

[58] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: *k*-anonymity and its enforcement through generalization and suppression. Technical report, SRI International, 1998.

[59] O. Simsek and D. Jensen. Navigating networks by using homophily and degree. *PNAS*, 2008.

[60] L. Sweeney. Uniqueness of simple demographics in the U.S. population. Technical Report LIDAP-WP4, Carnegie Mellon University, Laboratory for International Data Privacy, 2000.

[61] L. Sweeney. *k*-anonymity: a model for protecting privacy. *Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, 2002.

[62] D. Watts, P. Dodds, and M. Newman. Identity and search in social networks. *Science*, 2002.

[63] D. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 1998.

[64] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *ICDM*, 2002.

[65] X. Ying and X. Wu. Randomizing social networks: a spectrum preserving approach. In *SIAM Conference on Data Mining*, 2007.

[66] W. Zachary. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*, 1977.

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

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

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

[70] B. Zhou, J. Pei, and W.-S. Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *SIGKDD Explorations*, 2008.

[71] L. Zou, L. Chen, and T. Ozsu. K-Automorphism: A general framework for privacy preserving network publication. In *VLDB*, 2009.