Chapter 1

Privacy-Preserving Data Analysis on Graphs and Social Networks

Kun Liu

IBM Almaden Research Center

Kamalika Das

University of Maryland Baltimore County

Tyrone Grandison

IBM Almaden Research Center

Hillol Kargupta

University of Maryland Baltimore County & Agnik, LLC

Abstract While literature within the field of privacy-preserving data mining (PPDM) has been around for many years, attention has mostly been given to the perturbation and anonymization of tabular data; understanding the role of privacy over graphs and networks is still very much in its infancy. In this chapter, we survey a *very recent* body of research on privacy-preserving data analysis over graphs and networks in an effort to allow the reader to observe common themes and future directions.

1.1 Introduction

The proliferation of social networks, online communities, peer-to-peer file sharing and telecommunication systems has created large, complex graphs. These graphs are of significant interest to researchers in various application

domains such as marketing, psychology, and epidemiology. Research in these areas has revealed interesting properties of the data and presented efficient ways of maintaining, mining and querying them. Distributed and ubiquitous computing over these networks, which are essentially graph structures, is also an emerging topic with increasing interest in the data mining community. However, with the exception of some recent work, the privacy concerns associated with data-analysis over graphs and networks have been largely ignored. In this chapter, we provide a detailed survey of the *very recent* work on privacy-preserving data analysis over graphs and networks in an effort to allow the reader to observe common themes and future directions.

In a network, nodes correspond to individuals or other social entities, and edges correspond to relationships between them. The privacy breaches in a network can be grouped to three categories: 1) *identity disclosure*: the identity of an individual who is associated with a node is revealed; 2) *link disclosure*: the sensitive relationships between two individuals are disclosed; and 3) *content disclosure*: the sensitive data associated with each node is compromised, *e.g.*, the email message sent and/or received by the individuals in a email communication network. A privacy-preservation system over graphs and networks should consider all of these issues. However, compared with existing anonymization and perturbation techniques of tabular data (see, *e.g.*, the survey book [3]), working with graphs and networks is much more challenging due to the following reasons:

- It is difficult to model the background knowledge and the capability of an attacker. Any topological structures of the graph can be exploited by the attacker to derive private information. Two nodes that are indistinguishable with respect to some structural metrics does not guarantee they are on other metrics. Hence, it is not clear what are the most appropriate privacy models for graphs and networks, and how to measure the privacy breach in that setting.

- It is difficult to quantify the information loss. A graph can be worth a thousand words. It contains rich information but there is no standard ways to quantify the information loss incurred by the changes of its nodes and edges. How important are those network measures (*e.g.*, degree, clustering coefficient, average path length, diameter, centrality, betweenness, etc.) to graph-mining applications (*e.g.*, clustering, community discovery, viral marketing, etc.)? How well should we preserve those measures?

- It is even difficult to devise graph-modification algorithms that balance the goals of preserving privacy with the utility of the data. Different from tabular data where each tuple can be viewed as an independent sample from some distribution, the nodes and edges in a graph are all correlated. Therefore, the impact of a single change of an edge or a node can spread across the whole network.

- It is difficult to model the behavior of the participants involved in a network-based collaborative computing environment. Some participants may be quite honest and follow the rules; some may decide to behave dishonestly and exploit the system without contributing much; some may even intentionally try to collude with other parties to expose the private data of a specific individual.

To combat these challenges, several authors have recently developed different types of privacy models, adversaries, and graph-modification algorithms. Unfortunately, none of the work is likely to solve all the problems in one shot. Protecting against each kind of privacy breaches may require different techniques or a combination of them. In this chapter, we detail a number of recently developed techniques for each type of the *disclosure* described above. We hope this survey can offer insight into the challenges and therefore opportunities in this emerging area.

The remainder of this chapter is organized as follow. Section 1.2 describes definitions and notation used throughout. Section 1.3 discusses identity disclosure. Section 1.4 details link disclosure. Section 1.5 briefs content disclosure. Section 1.6 discusses privacy issues that arise from multi-party distributed computing, which we believe can serve as a foundation for the research of content disclosure over graphs and network with user interactions. Finally, Section 1.7 outlines future directions and concludes the chapter.

1.2 Definitions and Notation

We model a social network as a graph $G = (V_G, E_G)$, with vertices $V_G = \{v_1, \ldots, v_n\}$ corresponding to individuals and edges $E_G = \{(v_i, v_j) | v_i, v_j \in V_G, i \neq j, 1 \leq i, j \leq n\}$ the social relationships among them. We use $\mathbf{d}_G$ to denote the degree sequence of G. That is, $\mathbf{d}_G$ is a vector of size n, with the i-th element $\mathbf{d}_G(i)$ being the degree of the i-th node of G. A graph isomorphism from G to H is a bijection: $f : V_G \rightarrow V_H$ such that an edge $(u, v) \in E_G$ if and only if $(f(u), f(v)) \in E_H$. A graph automorphism is a graph isomorphism with itself, *i.e.*, a mapping from the vertices of the given graph G back to vertices of G such that the resulting graph is isomorphic with G. An automorphism f is non-trivial if it is not the identity function. Through out this chapter, we use the terms "network" and "graph" interchangeably.

1.3 Identity Disclosure

The identity disclosure problem often arises from the scenario where the data owner wants to publish or share, with a third party, a network that permits useful analysis without disclosing the actual identity of the individuals involved in the network. Here each individual is represented by a node on the network. A common practice, called *naive anonymization*, removes the personally identifying information associated with each node or replaces it with a pseudo-random name. However, as we will show later in this section, this simple approach does not always guarantee privacy. Under certain conditions, the attackers can still re-identify the individuals by combining external knowledge with the observed graph structure.

1.3.1 Active Attacks and Passive Attacks

Backstrom *et al.* [6] considered two different types of attacks on a naively-anonymized social network. The first is an *active attack*, where an attacker creates new user accounts and edges in the original network and uses them to find targets and their relations in the anonymized network. The second is a *passive attack*, where users of the system find themselves in the anonymized network and discover identities and edge relations of other connected users. These attacks are based on the uniqueness of small random subgraphs embedded in an arbitrary network, using ideas related to those found in arguments from Ramsey theory [13]. Interested readers may observe that *identity disclosure* often leads to *link disclosure*. However, in this section we focus on identity disclosure and will discuss the latter in Section 1.4.

Next, we give the formal definition of the problem Backstrom *et al.* studied:

Problem 1.1 *Given a social network $G = (V_G, E_G)$ and an arbitrary set of targeted users $U = \{u_1, \ldots, u_b\}$, identify U in the naively-anonymized copy of G and hence determine whether edge-relation (u_i, u_j) exists.*

The *active attack* proceeds as follows. Before the anonymized graph is produced and published, the attacker registers k new user accounts $\{x_1, \ldots, x_k\}$ in the system, and it connects them together to create a subgraph H. The attacker then creates links between these new accounts to nodes in the target set $\{u_1, \ldots, u_b\}$, and potentially other nodes in G as well. These links are created depending on the specific application scenario, *e.g.*, by sending messages to the targeted users or adding targeted users to the friends list or the address book of these new accounts. After the anonymized version of G is released, the attacker solves a special instance of the *subgraph isomorphism problem* to find H that is planted in G. Having identified H, the attacker can locate targeted users $\{u_1, \ldots, u_b\}$, thereby determining all the edge relations among them.

It should be noted that to make the above framework work, the subgraph H has to satisfy the following properties: 1) it is uniquely identifiable in G with high probability, regardless of G's structure and regardless of how it is attached to G; 2) it can be efficiently found from G by the attacker; and 3) H has no non-trivial automorphisms. The proof of the correctness and efficiency of the attacks is rather complicated, and we refer interested readers to [6] for a better treatment. It has been shown that with $|V| = n$ and $k = \Theta(\log n)$ new accounts, a randomly generated subgraph H will be unique with high probability. Moreover, if the maximum node degree in H is $\Theta(\log n)$, then H can be recovered efficiently, as well as the identities of up to $\Theta(\log^2 n)$ targeted nodes to whom the attacker created links from H. In practice, k can be set to values even smaller than the suggested bounds.

The experiments on a 4.4-million-node and 77 million-edge social network extracted from LiveJournal.com show that, the creation of 7 nodes by an attacker can reveal an average of 70 targeted nodes, and hence compromise the privacy of approximately 2400 edge relations among them. The authors further showed that, in the worse case, at least $\Omega(\sqrt{\log n})$ nodes are needed in any active attack to begin compromising the privacy of arbitrary targeted nodes.

The *passive attack* is based on the observation that most nodes in a real social network already belong to a small uniquely identifiable subgraph. Therefore, if a user u is able to collude with a coalition of $(k-1)$ friends after the release of the network, he or she will be able to identify and compromise the privacy of neighbors connected to this coalition. We refer readers to [6] for more details.

1.3.2 k-Candidate Anonymity and Graph Randomization

Hay *et al.* [18] considered the problem of re-identifying a known individual in the naively-anonymized network. They observed that the structural similarity of the nodes in the graph and the background knowledge an attacker obtains jointly determines the extent to which an individual can be distinguished. For example, if the attacker knows that somebody has exactly 5 social contacts, then he can locate all the nodes in the graph with degree 5. If there are very limited nodes satisfying this property, then the target might be uniquely identified.

Along this direction, the authors proposed a privacy model for social networks, which is based on the notion of k-anonymity [27].

Definition 1.1 (k-candidate anonymity) *A graph satisfies k-candidate anonymity with respect to a structural query if the number of the matching candidate nodes is at least k.*

Alternatively, an anonymized graph satisfies *k-candidate anonymity* if for a given structural query, no individual can be identified with a probability higher

than $1/k$.

The query evaluates the existence of the neighbors of a node or the structure of the subgraph in the vicinity of a node. It implicitly models the background knowledge (or the power) of an attacker. In their work [18], Hey *et al.* studied two types of queries: 1) *vertex refinement query*, which defines a class of queries of increasing power to report the structural information about a node's position in the network. The weakest query $\mathcal{H}_0(x)$ simply returns the identifier (or the pseudo-random name) of node x; $\mathcal{H}_1(x)$ returns the degree of x; $\mathcal{H}_2(x)$ returns the degree of each neighbor of x, and so on. 2) *subgraph knowledge query*, which verifies the existence of a specific type of subgraph around the target node. The descriptive power of such a query is measured by counting the number of edges (also known as *edge facts*) contained the subgraph.

To protect against these types of attacks, the authors studied a random-perturbation technique that modifies the graph through a sequence of random edge-deletions followed by edge-insertions. While this approach can potentially reduce the risk of re-identification, it does not guarantee that the modified graph satisfies k-candidate anonymity, neither does it guarantee that the utility of the original graph can be well preserved. This technique is further studied by Ying and Wu [34] in the context of sensitive link/relationship protection. They evaluated the impact of edge randomization on some spectrum properties of the graph, and developed a new strategy to better preserve these properties without sacrificing much of the privacy. We will detail their technique in Section 1.4.3.

1.3.3 k-Degree Anonymity and Minimal Edge Modifications

Liu and Terzi [25] studied a specific graph-anonymity model called *k-degree anonymity*, which prevents the re-identification of individuals by adversaries with *a priori* knowledge of the degrees of certain nodes. Note that this is related to the *vertex refinement query* discussed in Section 1.3.2.

Definition 1.2 (k-degree anonymity) *A graph $G = (V_G, E_G)$ is k-degree anonymous if every node $v \in V_G$ has the same degree with at least $(k-1)$ other nodes.*

Based on this privacy model, the authors addressed the following problem:

Problem 1.2 *Given a graph $G = (V_G, E_G)$ and an integer k, modify G via a set of edge-addition operations in order to construct a new graph $G' = (V_{G'}, E_{G'})$ such that 1) G' is k-degree anonymous; 2) $V_{G'} = V_G$; and 3) $E_{G'} \cap E_G = E_G$.*

It is easy to see that one could transform G to the complete graph, in which all nodes share the same degree. Although such an anonymization would preserve privacy, it would make the anonymized graph useless for any study. For that reason, the authors imposed the additional requirement that the

minimum number of edge-additions is made. This constraint tries to capture the requirement of structural similarity between the input and output graphs. Note that minimizing the number of additional edges can be translated into minimizing the L_1 distance of the degree sequence of G and G', since it holds that $|E_{G'}| - |E_G| = \frac{1}{2}L_1(\mathbf{d}_{G'} - \mathbf{d}_G)$. With this observation, the authors proposed a two-step framework for the graph-anonymization problem. The algorithms proceed as follows:

1. First, starting from the original degree sequence $\mathbf{d}_G$, construct a new degree sequence $\mathbf{d}'$ that is k-anonymous and the cost $L_1(\mathbf{d}' - \mathbf{d}_G)$ is minimized.

2. Given the new degree sequence $\mathbf{d}'$, construct a graph $G'(V_{G'}, E_{G'})$ such that $\mathbf{d}_{G'} = \mathbf{d}'$, $V_{G'} = V_G$ and $E_{G'} \cap E_G = E_G$ (or $E_{G'} \cap E_G \approx E_G$ in the relaxed version).

The first step is solved by a linear-time dynamic programming algorithm; the second step is solved by a set of graph-construction algorithms which are related to the realizability of degree sequences. The authors also extended their algorithms to allow for edge deletions as well as simultaneous edge additions and deletions. Experiments on a large spectrum of synthetic and real network datasets demonstrate that their algorithms are efficient and can effectively preserve the graph utility while satisfying k-degree anonymity.

1.3.4 k-Neighborhood Anonymity and Graph Isomorphism

Zhou and Pei [36] considered the subgraph constructed by the immediate neighbors of a target node. The assumption is that the unique structure of the neighborhood subgraph can be used by the attacker to distinguish the target from other nodes. This observation is closely related to the *subgraph knowledge queries* discussed in Section 1.3.2. Based on this assumption, the authors defined a new notion of the anonymity on graphs, which we call the *k-neighborhood anonymity*.

Definition 1.3 (k-neighborhood anonymity) *A node is k-anonymous in a graph G if there are at least $(k-1)$ other nodes $v_1, \ldots, v_{k-1} \in V_G$ such that the subgraphs constructed by the neighbors of each node $v_1, \ldots, v_{k-1}$ are all isomorphic. A graph satisfies k-neighborhood anonymity if all the nodes are k-anonymous as defined above.*

Following this definition, the authors specifically considered the following problem:

Problem 1.3 *Given a graph $G = (V_G, E_G)$ and an integer k, construct a new graph $G' = (V_{G'}, E_{G'})$ such that 1) G' is k-neighborhood anonymous; 2) $V_{G'} = V_G$; 3) $E_{G'} \supseteq E_G$; and 4) the information loss incurred by anonymization is not too much.*

The algorithm for solving the above problem consists of three steps. First, it marks all the nodes as "unanonymized" and sorts them in descending order of their neighborhood size. Here the "neighborhood size" is defined as the number of edges and nodes of the subgraph constructed by the immediate neighbors of a node. Then, the algorithm picks up the first "unanonymized" node u from the sorted list, finds the top $(k-1)$ other nodes $\{v_1, \ldots, v_{k-1}\}$ from the list whose neighborhood subgraphs are most similar to that of u (we call it *subgraph similarity computation*). Next, the algorithm iteratively considers every pair of nodes (u, v_i), $i = 1, \ldots, k-1$, and for each pair (u, v_i), the algorithm modifies the neighborhood subgraph of u and the neighborhood subgraph of v_i to make them isomorphic to each other. The modification is performed by adding extra edges while keeping the nodes intact (we call it *subgraph isomorphism modification*). After all the neighborhood subgraphs of these k nodes are pair-wise isomorphic, the algorithm marks these k nodes as "anonymized". The process continues until all the nodes in the graph are "anonymized".

The information loss is measured by three factors: 1) extra edges added to the neighborhood; 2) nodes that were not in the neighborhood of the anonymized nodes but are now in; and 3) information loss due to the value generalization of the node's label if there is any such operations. Since the *subgraph similarity computation* and *subgraph isomorphism modification* are all based on greedy heuristics, there is no guarantee that the information loss is minimal, therefore, the utility of the anonymized graph can only be evaluated empirically.

1.3.5 Personally Identifying Information on Social Networking Sites

So far we have restricted our discussion to the problem of privacy-preserving graph publishing and sharing, and have largely ignored the privacy risks associated with personal information sharing in the real social networks such as Facebook and MySpace.

While specific goals and patterns vary significantly across these social networking sites, the most common model is based on the presentation of the user's profile and the visualization of his connections to others. As the profile and connection often reveal vast amounts of personal and sometimes sensitive information (*e.g.*, photo, birth date, phone number, current residence, dating preference, current relationship status, political views, and various interests), it is highly likely that a user can be uniquely identified even if he does not openly expose his identity.

In an effort to quantify the privacy risk associated with these networks, Acquisti and Gross [2] combined online social network data and other publicly available data sets in order to estimate whether it is possible to re-identify PII (personally identifying information) from simple PI (personal information). This re-identification may happen, through photos, demographic data,

category-based representations of interests that indicate unique or rare overlaps of hobbies. Their research supports the claim that large amounts of private information are available publicly.

1.4 Link Disclosure

The link disclosure problem is centered around the protection of the connection between vertices in a network. Two entities in a social network may have a myriad of connections. Some that are safe for the public to know and others that should remain private. Techniques to solve this problem, while still extracting analytic value from the network, have just started to emerge in the literature. In this section, we describe some recent work in this area.

1.4.1 Link Re-identification

Zheleva and Getoor [35] focused on the problem of link re-identification, which they define as inferring sensitive relationships from anonymized graph data. Graph nodes represent entities that are assumed to have multiple relationships, which are modelled as edges, between them. Edges may be of different types and can be classified as either sensitive or observed. The core problem addressed was how to minimize the probability of predicting sensitive edges based on the observed edges. The goal is to attain privacy preservation of the edge information, while still producing anonymized data that is useful. Utility is measured by the number of observational edges removed. The higher the number of removed observations, the lower the overall utility.

This goal is achieved by employing one of the five anonymization approaches outlined in the paper. Their first algorithm, called *Intact edges*, removes all sensitive edges and leaves all the observational ones. The second algorithm, called *Partial edge removal*, deletes observational edges that may contribute to the inference of a sensitive relationship. The criteria is left up to the reader to set. They demonstrate this algorithm using a random removal strategy. In the first two approaches, the number of nodes in the graph was unchanged and the edges constructed as links between their anonymized versions. In the *cluster-edge anonymization* approach, all the anonymized nodes are collapsed into a single node (per cluster) and a decision is made on which edges to include in the collapsed graph. The *Cluster-edge anonymization with constraints* approach uses a more restrictive sanitization technique for the observed edges, by creating edges between equivalence classes if and only if the equivalence class nodes have the same constraints as any two nodes in the original graph. The final approach, called *Removed edges*, removes all relationships/edges from the graph. They recognize that the effectiveness of the approaches depend on the

structural and statistical characteristics of the underlying graph. The experiments were carried out on a variety of graphs with varying characteristics and confirmed intuitive expectations, *e.g.*, as the number of observational edges decreased, so did the number of correctly identified sensitive relationships.

In short, Zheleva and Getoor concentrated on an often unexamined aspect of link disclosure - mitigating the risk of link re-identification.

1.4.2 Privacy-Preserving Link Analysis

Duan *et al.* [12] proposed an algorithm that enables link analysis in situations where there is no stated link structure between the nodes. They constrained their discussion to the domain of expert identification and authoritative document discovery and leverage the observation that a user's level of expertise is reflected by the document they access. Their *Secure Online HITS* algorithm is an extension of Kleinberg's HIT algorithm [21], where they replaced the 0-1 hyperlink property with a non-negative value, *i.e.*, a weight, which models the user's behavior.

Given users and their behaviors, whether through access logging systems or other means, they construct a graph such that the users are vertices and log entries represent edges between two users. Then an eigengap (difference between the largest and the second largest eigenvalues) is computed using their *online eigenvector calculation* method, which performs in environments where frequent updates are the norm, by estimating the perturbation upper bound and delaying applying updates when possible. Due to the fact that they were logging (possibly) sensitive information from which they build the graph, they augmented their basic algorithm to address the privacy concerns. This was done by leveraging public key encryption to ensure that only aggregate or encrypted data was exposed.

To empirically test the algorithm, they ran it on the Enron Email Dataset [9]. They used the message count between the sender and the recipient as the weight in order to determine if the algorithm could identify the central figures in the social network. The experiments demonstrated that their algorithm provided estimated rankings that closely matched the actual ones.

In short, Duan *et al.* furthered the state of the art by demonstrating how core principles, like access pattern inference, can be used to construct graph structure, when none appears to exist.

1.4.3 Random Perturbation for Private Relationship Protection

Ying and Wu [34] studied two randomization techniques to protect private relationships. The first one, called *Rand Add/Del*, modifies the graph by a sequence of random edge-additions followed by edge-deletions. The second, called *Rand Switch*, randomly switches a pair of edges to produce a new edge set $\widetilde{E} \leftarrow E \backslash \{(t,w),(u,v)\} \cup \{(t,v),(w,u)\}$ provided that $(t,v) \notin E$ and

$(w, u) \notin E$, and repeats this process many times. The first randomization preserves the total number of edges in the original graph, while the second one maintains the degree of each node.

The authors evaluated, both empirically and theoretically, the impact of randomization on the eigen-spectrum of the graph. In particular, they focused on two important eigenvalues: 1) the largest eigenvalue of the adjacency matrix, which is closely related to the maximum degree, chromatic number, clique number and subgraph centrality of a graph; and 2) the second smallest eigenvalue of the Laplacian matrix (also known as algebraic connectivity [14]), which reflects how well connected the overall graph is, and has implications for properties such as clustering and synchronizability.

Using some theoretical results from Cvetkovic *et al.* [10], the authors developed the spectrum-preserving versions of *Rand Add/Del* and *Rand Switch*. The new algorithms selectively choose the edges that should be added, removed, or switched in order to control the changes of the eigenvalues. The privacy is evaluated by the prior and posterior belief of the existence of an edge. The authors developed closed-form expressions for evaluating *Rand Add/Del* and *Rand Switch*, and claimed that their spectrum-preserving counterparts should not differ much in protecting the privacy.

1.4.4 Cryptographic Protocols for Private Relationships Protection

Carminati *et al.* [7] considered an access control model where only authorized users who satisfy some *access conditions* are granted right to the resources owned by another user in a social network. Here the resources can be personal profiles, blogs, photos, etc.

The *access conditions* specify the type of the relationship between the requestor and owner (*e.g.*, colleagues, alumni), the depth of this relationship (*e.g.*, length of the friendship chain), and the trust level (*e.g.*, fully trusted, semi-trusted). Since knowing who is trusted by a user and to what extent disclose a lot about that user's personal interests, it is desirable to protect that information during the authentication process.

For this reason, the authors developed a symmetric-key protocol to enforce a selective dissemination of the relationship information during the authentication. This problem is further studied by Domingo-Ferrer [11], who developed a public-key protocol that does the same job as [7], without requiring a trusted third party.

1.4.5 Deriving Link Structure of the Entire Network

Korolova *et al.* [22] considered the problem that an attacker wants to derive the link structure of the entire network by collecting neighborhood information of some compromised users, who are either bribed or whose accounts are broken by the attacker. These users are chosen using different criteria, *e.g.*,

uniformly at random (*Random*), in the descending order of their node degrees (*Highest-Degree*), etc.

Analysis shows that the number of users needed to be compromised in order to cover a constant fraction of the entire network drops exponentially with increase in the lookahead parameter ℓ. Here a network has a lookahead ℓ if a registered user can see all of the links and nodes incident to him within distance ℓ from him. For example, a social network has $\ell = 0$ if a user can only see who are his immediate neighbors; has $\ell = 1$ if a user an see who are his immediate neighbors as well as his neighbors' immediate neighbors. A good example of a social network with $\ell = 1$ is LinkedIn. Experiments on a 572,949-node friendship graph extracted from LiveJournal.com show that 1) *Highest-Degree* yields the best performance while *Random* performs the worst; and 2) in order to obtain 80% coverage of the graph using lookahead 2, *Highest-Degree* needs to bribe 6,308 users; to obtain the same coverage using lookahead 3, *Highest-Degree* only needs to bribe 36 users.

1.4.6 Synthetic Graph Generation

Instead of modifying the graph to have it satisfy some k-anonymity criteria, Leskovec and Faloutsos [23] considered the problem of synthetic-graph generation. That is, given a large graph G, compute the most likely parameters Θ that would generate a synthetic-graph G' having the same properties as G. Hence, the data owner can publish G' without revealing the exact information about the original graph G.

The parameter $\Theta = [\theta_{ij}]$ defined in [23] is a $n_1 \times n_1$ probability matrix, where $n_1 << n$ and the element $\theta_{ij} \in [0, 1]$ indicates the probability that edge (i, j) is present. Given the original graph G, Θ is calculated by maximum likelihood estimation: $\arg\max_\Theta P(G|\Theta)$. To evaluate this formula, the authors developed a linear-time algorithm (a naive approach would take super-exponential time) by exploiting the structure of Kronecker product and by using a sampling technique.

Given the estimated parameter Θ, one can sample an initiator graph G_1 with n_1 nodes, and by recursion produce successively larger graphs $G_2, \ldots, G_k$ such that the k-th graph G_k is on $n_k = n_1^k$ nodes. To be more specific, let A_G denote the adjacency matrix of a graph G, we have $A_{G_k} = A_{G_1}^k = A_{G_{k-1}} \otimes A_{G_1}$, where $\otimes$ is the Kronecker product and the graph corresponding to A_{G_k} is called Kronecker graph. Note that this approach assumes that Kronecker Graphs, which is self-similar and based on a recursive construction, is a good model for the real graph G. We refer interested readers to [23] and the references wherein for more details.

1.5 Content Disclosure

Content disclosure is normally an issue when the private data associated with a user on the network is disclosed to others. A very interesting example recently arose from Facebook's "Beacon" service, a "social ads" system where your own expressed brand preferences and Internet browsing habits, and even your very identity are used to market goods and services to you and your friends. For example, adding the latest season of LOST to your queue on Blockbuster.com might have Facebook place an ad for Blockbuster straight on your friends' news feeds. This helps Facebook and its partners (Blockbuster in this example) make money because, as Facebook's CEO Mark Zuckerberg extols, "nothing influences a person more than a recommendation from a trusted friend." This may be fine in some situation, but there may be some things that one is not prepared to share with the entire world. From the users perspective, they want to ask how to avoid the disclosure of their personal private information while still enjoying the benefit of social advertisement, *e.g.*, promise of free iTunes songs and movies. From the company's perspective, they want to know how to assure the users that their privacy is not compromised while doing social advertisement. Privacy concerns regarding content disclosure exist in other application scenarios such as social recommendation, etc.

Protecting against this kind of disclosure is an important research and engineering problem. However, the work in the literature thus far does not take into account the impact of graph structures as other two types of disclosures, but mostly focuses on 1) simple opt-in and opt-out setting and 2) standard data perturbation and anonymization for tabular data. The first approach allows the registered user to determine whether he wants to disable the service, and it is being used in limited application scenarios. The second approach is more generic and it relies on traditional privacy-preserving data masking techniques [3] to change the data that is to be shared.

1.6 Privacy in Multi-Party Distributed Computing

Since users and companies on a social network usually share and exchange some information, or jointly perform some task, we can see a connection between online activities and multi-party distributed computing. Here the graph structure may not play as an important role as in *identity and link disclosure* problems, but rather the behavior of users on the network and the task they want to achieve determines the extent to which the privacy is breached. Therefore, we believe that the privacy-preservation research in

distributed computing can form a foundation for research on content disclosure for graphs and networks. Next, we introduce some work in that area aimed at offering insights into the solutions to content disclosure for graphs and networks.

1.6.1 Secure Multi-Party Computation

Privacy-preservation objectives in distributed computing can often be framed as instances of secure multi-party computation (SMC) [33, 16], wherein multiple parties, each having a private input, want to compute some function of these inputs without revealing any information other than the function's output. For example, the private input could be each party's income and the computation would return who is the richest. This example is known as the *millionaire's problem* and was first discussed by Yao [33]. Usually, it is assumed that 1/3 or 1/2 of the parties may be "bad" (or called "malicious"), while everyone else is assume to be good (or called "semi-honest") and they execute the computation protocol as instructed. Although general approaches to SMC were proposed for a variety of settings in the 1980s, the computational and communication complexities hindered the application of SMC to privacy-preserving distributed data mining. In 2000, Lindell and Pinkas [24] designed an two-party SMC version of the ID3 algorithm for constructing a classification tree. They showed that a privacy-preserving data-mining task does not have to be cast as a monolithic SMC problem which requires an expensive general SMC solution. Instead, the task may be decomposed into small modules, with each module being implemented with special-purpose efficient SMC protocols. The key to such construction is that we are able to ensure secure chaining of the small SMC components. We prevent information from leaking at the seams between the components by having them produce not public intermediate outputs but rather individual party shares of the outputs. These shares may be fed as inputs to further SMC components. Since Lindell and Pinkas' pioneering work, a variety of SMC solutions for privacy-preserving distributed data mining have been proposed, questioned, and refined. We refer interested readers to [26, 8, 31, 3] for a thorough treatment. However, it should be noted that, as of today, a majority of the research in this area are still limited to two-party computation with the assumption of semi-honest behavior. Therefore they may not scale well in an application scenario with many malicious participants and large data sets.

A relatively new area of research is the application of game theory to analyze the *rational behavior* of the participants. Here, we would like to consider what happens if the participants are all trying to maximize their own benefits, rather than being simply "bad" and "good". In the next section we briefly mention some work in this area.

1.6.2 Game-Theoretic Framework for Privacy-Preserving Computation

1.6.2.1 Preliminaries of Game Theory

Before describing the game-theoretic framework for privacy-preserving distributed computing, we first provide a brief background of game theory.

A game is an interaction or a series of interactions between players, which assumes that 1) the players pursue well defined objectives (they are *rational*) and 2) they take into account their knowledge or expectations of other players' behavior (they *reason strategically*). For simplicity, we start by considering the most basic game - the *strategic game*.

Definition 1.4 (Strategic game) *The strategic game consists of*

- *a finite set P: the set of players,*
- *for each player $i \in P$ a nonempty set A_i: the set of actions available to player i,*
- *for each player $i \in P$ a preference relation $\succeq_i$ on $A = \times_{j \in P} A_j$: the preference relation of player i.*

The preference relation $\succeq_i$ of player i can be specified by a utility function $u_i : A \to \mathbb{R}$ (also called a payoff function), in the sense that for any $a \in A, b \in A$, $u_i(a) \geq u_i(b)$ whenever $a \succeq_i b$. The values of such a function is often referred to as utilities (or payoffs). Here a or b is called the *action profile*, which consists of a set of actions, one for each player. Therefore, the utility (or payoff) of player i depends not only on the action chosen by himself, but also the actions chosen by all the other players. Mathematically, for any action profile $a \in A$, let a_i be the action chosen by player i and a_{-i} be the list of actions chosen by all the other players except i, the utility of player i is $u_i(a) = u_i(\{a_i, a_{-i}\})$.

One of the fundamental concepts in game theory is the Nash equilibrium:

Definition 1.5 (Nash equilibrium) *A Nash equilibrium of a strategic game is an action profile $a^* \in A$ such that for every player $i \in P$ we have*

$$u_i(\{a_i^*, a_{-i}^*\}) \geq u_i(\{a_i, a_{-i}^*\}) \text{ for all } a_i \in A_i.$$

Therefore, Nash equilibrium defines a set of actions (an action profile) that captures a steady state of the game in which no player can do better by unilaterally changing her action while all other players do not change their actions. A game can have zero, one, or more than one Nash equilibrium.

Next, we introduce game-theoretic approaches in three different settings: *secret sharing*, *sovereign information sharing* and *multi-party privacy-preserving data mining*.

1.6.2.2 Rational Secret Sharing

Secret sharing is one of the main building blocks in modern cryptography. Shamir's secret sharing scheme [29] allows one to share a secret s (a natural number) among n other parties, so that any m of them may reconstruct it. The idea is as follows: party 0, who wants to share the secret, chooses an $(m-1)$ degree polynomial f such that $f(0) = s$, and tells party i the value of $f(i)$, $i = 1, \ldots, n$. Thus $f(i)$ is party i's *share* of the secret. Any m of parties $\{1, \ldots, n\}$ can jointly recover the secret by reconstructing the polynomial using Lagrange interpolation. However, any subset of parties with size less than m do not have any idea what the secret is. The underlying assumption of this protocol is that, at most $n - m$ parties are "bad" and "bad" parties cannot prevent the "good" parties from reconstructing the secret.

While in some situations, it makes sense to consider that some parties are "good" and some are "bad"; for other applications, it may be more realistic to view parties as rational individuals who are trying to maximize their benefits. The parties have certain preference over outcomes and can be expected to follow the protocol if and only if doing so increases their expected benefits. In this spirit is the work of Halpern and Teague [17], who considered the secret sharing problem where all parties are rational: 1) they prefer to get the secret to not getting it; 2) they prefer that as few as possible of the other parties get it. The authors showed that, under these assumptions, parties running Shamir's protocol will not cooperate. Using game-theoretic terminology, we say that for any party, not sending his share *weakly dominates* sending his share. To cope with this situation, the authors developed a randomized secret-sharing mechanism with constant expected running time, where the recommended strategy is a Nash equilibrium that survives iterated deletion of weakly-dominated strategies. The results were extended to secure multi-party computation with rational participants.

Abraham *et al.* [1] later introduced k-resilient Nash equilibrium, a joint strategy where no member of a coalition of size up to k can do better even if the whole coalition defects. The authors showed that such k-resilient Nash equilibrium exist for Shamir's secret sharing problem [29], which can be viewed as an extension of Halpern and Teague's work [17] since they did not consider collusion among the parties.

1.6.2.3 On Honesty in Sovereign Information Sharing

Sovereign information sharing [4] allows autonomous entities to compute queries across their databases in such a way that no extra information is revealed other than the result of the computation. Agarwal and Terzi [5] took a game-theoretic approach to address the following problem in a sovereign information-sharing setting: *how to ensure that all the rational participants behave honestly by providing truthful information, even though they can benefit from cheating.* They modelled the problem as a strategic game and showed that if nobody is punished for cheating, honest behavior cannot be an equi-

librium of the game. They therefore added a central auditing device that periodically checks whether any participant has cheated by altering his input. Whenever the device finds out a cheating participant, it penalizes him. The authors derived conditions under which a *unique* Nash equilibrium is achieved such that every participant provides truthful information. The relationship between the frequency of auditing and the amount of punishment in terms of benefits and losses from cheating was also derived.

A related work is the one by Kleinberg *et al.* [20], who considered different information-exchange scenarios and quantified the willingness of the participants to share their private information using solution concepts from coalition games. Note that Agarwal and Terzi are interested in quantifying when people are willing to provide truthful information in a game, while Kleinberg *et al.* are interested in quantifying whether people are willing to participate in the game at all.

1.6.2.4 Game-Theoretic Framework for Secure-Sum Computation

In a multi-party privacy-preserving data mining environment, each participant has certain responsibilities in terms of computation, communication and privacy protection. However, depending on the characteristics of these participants and their objectives, they can quit the process prematurely, provide bogus inputs, and collude with others to derive private information they should not know. Kargupta et al. [19] also took a game-theoretic approach to analyze this phenomenon and presented Nash equilibrium analysis of a well-known multi-party secure-sum computation [28, 8]. The basic idea is again to model the strategies and utilities of the participants as a game and penalize malicious behavior by increasing the cost of computation and communication. For example, if a participant suspects a colluding group of size k', then he may split the every number used in a secure sum into $\alpha k'$ pieces, $\alpha > 0$, and demand $\alpha k'$ rounds of secure-sum computation one for each piece. This simple strategy increases the computation and communication cost by $\alpha k'$-fold, which may counteract the possible benefit that one may receive by joining a team of colluders.

1.7 Conclusion and Future Work

This chapter provides a detailed survey of the very recent research on privacy-preserving data analysis over graphs and networks. Due to space constraints, we refer interested readers to [15, 32, 30] for other related work on this topic.

Before concluding this chapter, we present a set of recommendations for future research in this emerging area.

- Develop identity anonymity models for graphs and networks. Much of the existing research for identity disclosure is built upon the notion of k-anonymity. The fundamental research question remains "What is the ideal base model for privacy-preserving analysis of graphs and networks?"
- Develop efficient and effective graph-modification algorithms for sensitive link protection. A lot of the existing work leverages randomization techniques that change the graph, which is rather heuristic and does not preserve the utility of the graph very well.
- Understand the privacy constraints in the Web 2.0 environment. Develop privacy-preserving techniques to enable core value-added Web 2.0 services, such as social advertisement and recommendation.
- Develop workload-aware metrics that adequately quantify levels of information loss of graph data.
- Create a benchmark graph data repository. This would let researchers compare algorithms to more clearly understand the differences among various approaches.

It is our belief that the future will see a growth in the demand of privacy-protection techniques for not only social network but also other types of networks, such as communication and peer-to-peer networks. As more researchers, engineers and legal experts delve into this area, standards and theory will begin to take shape. As these are established, the next generation of privacy-preserving data analysis will be a fertile ground for all concerned with the privacy implications in our society.

References

[1] Ittai Abraham, Danny Dolev, Rica Gonen, and Joe Halpern. Distributed computing meets game theory: Robust mechanisms for rational secret sharing and multiparty computation. In *Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing (PODC'06)*, pages 53–62, Denver, CO, July 2006.

[2] Alessandro Acquisti and Ralph Gross. Privacy risks for mining online social networks. In *NSF Symposium on Next Generation of Data Mining and Cyber-Enabled Discovery for Innovation (NGDM'07)*, Baltimore, MD, October 2007.

[3] Charu C. Aggarwal and Philip S. Yu. *Privacy-Preserving Data Mining: Models and Algorithms*, volume 34 of *Advances in Database Systems*. Springer, 2008.

[4] Rakesh Agrawal, Alexandre Evfimievski, and Ramakrishnan Srikant. Information sharing across private databases. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data (SIGMOD'03)*, pages 86–97, San Diego, CA, 2003.

[5] Rakesh Agrawal and Evimaria Terzi. On honesty in sovereign information sharing. In *10th International Conference on Extending Database Technology (EDBT'06)*, pages 240–256, Munich, Germany, March 2006.

[6] Lars Backstrom, Cynthia Dwork, and Jon M. Kleinberg. Wherefore art thou R3579X?: Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International Conference on World Wide Web (WWW'07)*, pages 181–190, Alberta, Canada, May 2007.

[7] Barbara Carminati, Elena Ferrari, and Andrea Perego. Private relationships in social networks. In *Private Data Management Workshop (held in conjunction with ICDE'07)*, Istanbul, Turkey, April 2007.

[8] Chris Clifton, Murat Kantarcioglu, Jaideep Vaidya, Xiaodong Lin, and Michael Y. Zhu. Tools for privacy preserving distributed data mining. *ACM SIGKDD Explorations*, 4(2):28–34, 2003.

[9] William W Cohen. Enron email dataset, http://www-2.cs.cmu.edu/ enron/.

[10] Dragos Cvetkovic, Peter Rowlinson, and Slobodan Simic. *Eigenspaces of Graphs (Encyclopedia of Mathematics and its Applications)*. Cambridge University Press, 1997.

[11] Josep Domingo-Ferrer. A public-key protocol for social networks with private relationships. In *Modeling Decisions for Artificial Intelligence*, volume 4617, pages 373–379. Springer Berlin/Heidelberg, 2007.

[12] Yitao Duan, Jingtao Wang, Matthew Kam, and John Canny. Privacy preserving link analysis on dynamic weighted graph. *Computational & Mathematical Organization Theory*, 11:141–159, 2005.

[13] Paul Erdös. Some remarks on the theory of graphs. *Bullein of the AMS*, 53:292–294, 1947.

[14] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(98):298–305, 1973.

[15] Keith B. Frikken and Philippe Golle. Private social network analysis: How to assemble pieces of a graph privately. In *Proceedings of the 5th*

ACM Workshop on Privacy in Electronic Society (WPES'06), pages 89–98, Alexandria, VA, 2006.

[16] O. Goldreich. *The Foundations of Cryptography*, volume 2, chapter 7. Cambridge University Press, 2004.

[17] Joseph Halpern and Vanessa Teague. Rational secret sharing and multiparty computation: Extended abstract. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing (STOC'04)*, pages 623–632, Chicago, IL, June 2004.

[18] Michael Hay, Gerome Miklau, David Jensen, Philipp Weis, and Siddharth Srivastava. Anonymizing social networks. Technical report, University of Massachusetts Amherst, 2007.

[19] Hillol Kargupta, Kamalika Das, and Kun Liu. A game theoretic approach toward multi-party privacy preserving distributed data mining. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07)*, pages 523–531, Warsaw, Poland, September 2007.

[20] Jon Kleinberg, Christos H. Papadimitriou, and Prabhakar Raghavan. On the value of private information. In *Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge (TARK VIII)*, Italy, July 2001.

[21] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46:604–632, 1999.

[22] Aleksandra Korolova, Rajeev Motwani, Shubha U. Nabar, and Ying Xu. Link privacy in social networks. In *Proceedings of the 24th International Conference on Data Engineering (ICDE'08)*, Cancun, Mexico, April 2008.

[23] Jure Leskovec and Christos Faloutsos. Scalable modeling of real graphs using kronecker multiplication. In *Proceedings of 2007 International Conference on Machine Learning (ICML'07)*, pages 497–504, Corvallis, OR, June 2007.

[24] Yehuda Lindell and Benny Pinkas. Privacy-preserving data mining. In *Advances in Cryptology (CRYPTO'00)*, volume 1880 of *Lecture Notes in Computer Science*, pages 36–54. Springer-Verlag, 2000.

[25] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of ACM SIGMOD*, Vancouver, Canada, June 2008.

[26] B. Pinkas. Cryptographic techniques for privacy preserving data mining. *SIGKDD Explorations*, 4(2):12–19, 2002.

[27] Pierangela Samarati and Latanya Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proceedings of the Seven-*

teenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98), page 188, Seattle, WA, 1998.

[28] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 2nd edition, 1995.

[29] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, November 1979.

[30] Lisa Singh and Justin Zhan. Measuring topological anonymity in social networks. In *Proceedings of IEEE International Conference on Granular Computing (GRC'07)*, page 770, November 2007.

[31] Jaideep Vaidya, Chris Clifton, and Michael Zhu. *Privacy Preserving Data Mining*, volume 19 of *Series: Advances in Information Security*. Springer, 2006.

[32] Da-Wei Wang, Churn-Jung Liau, and Tsan sheng Hsu. Privacy protection in social network data disclosure based on granular computing. In *Proceedings of 2006 IEEE International Conference on Fuzzy Systems*, pages 997–1003, 2006.

[33] A. C. Yao. How to generate and exchange secrets. In *Proceedings 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.

[34] Xiaowei Ying and Xintao Wu. Randomizing social networks: a spectrum preserving approach. In *Proceedings of 2008 SIAM International Conference on Data Mining (SDM'08)*, Atlanta, GA, April 2008.

[35] Elena Zheleva and Lise Getoor. Preserving the privacy of sensitive relationships in graph data. In *Proceedings of the International Workshop on Privacy, Security, and Trust in KDD (PinKDD'07)*, San Jose, CA, August 2007.

[36] Bin Zhou and Jian Pei. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the 24th International Conference on Data Engineering (ICDE'08)*, Cancun, Mexico, April 2008.