EWNI: Efficient Anonymization of Vulnerable Individuals in Social Networks

Frank Nagle[1], Lisa Singh[1], and Aris Gkoulalas-Divanis[2]

[1] Georgetown University, Washington, DC 20057, USA
[2] IBM Research-Zürich, Rüschlikon, CH-8803, Switzerland

Abstract. Social networks, patient networks, and email networks are all examples of graphs that can be studied to learn about information diffusion, community structure and different system processes; however, they are also all examples of graphs containing potentially sensitive information. While several anonymization techniques have been proposed for social network data publishing, they all apply the anonymization procedure on the entire graph. Instead, we propose a local anonymization algorithm that focuses on obscuring structurally important nodes that are not well anonymized, thereby reducing the cost of the overall anonymization procedure. Based on our experiments, we observe that we reduce the cost of anonymization by an order of magnitude while maintaining, and even improving, the accuracy of different graph centrality measures, e.g. degree and betweenness, when compared to another well known data publishing approach.

1 Introduction

Social networks, patient networks, email networks, and disease transmission networks are all examples of graphs that can be studied to learn about information diffusion, community structure and different system processes; however, they are also all examples of graphs containing potentially sensitive information. For some of these networks, it is not just the personal information that is sensitive, but also the position or existence of an individual in the graph. For example, the existence of a patient in a disease transmission network may be deemed as highly sensitive. As a result, a need exists to obscure sensitive topological information while still maintaining accurate graph properties for those studying these networks. Furthermore, because researchers are typically interested in data exploration applications, like community identification and information diffusion, our goal is to publish an anonymized, transformed network that is resilient against identity attacks and can be effectively studied as a graph.

A number of approaches have been proposed for anonymizing graphs. The research literature can be separated into two groups, those that add noise to the base graph and those that generalize the base graph. The former approaches use edge insertions and deletions to either deterministically create common patterns in the graph [24,16,25,3,26,7,6,22,19] or probabilistically add uncertainty [13,23,10,4]. The generalization strategy hides the detail level of the network by partitioning the graph into subgraphs, grouping nodes into clusters [5,20], or releasing specialized data structures that are specific for answering certain types of queries [12,14,11]. In this work, we investigate ways to publish a base graph that is sufficiently anonymized, contains sufficient detail for graph mining tasks using different graph properties, and is efficiently computed.

P.-N. Tan et al. (Eds.): PAKDD 2012, Part II, LNAI 7302, pp. 359–370, 2012.

This work makes the following contributions: (1) Describes the use of simple graph metrics to guide the anonymization process; (2) Proposes a local anonymization strategy focusing on edge insertion or deletion to only the subset of nodes that are considered vulnerable and; (3) Experimentally evaluates the proposed method and shows that the released graph maintains a high degree of accuracy for different graph properties, including degree, betweenness, diameter, and average path length on five real world data sets, and are an order of magnitude more efficient than a well known approach that alters the entire graph.

The rest of the paper is organized as follows. Related literature is presented in section 2. In section 3, we provide necessary background and our privacy model. Our anonymization strategies are presented in section 4, followed by an experimental evaluation in section 5, and conclusions in section 6.

2 Related Literature

A number of previous works have shown that just removing the labels of published graphs is not sufficient for anonymization [2,12,17]. The main threats against these naïvely anonymized networks are *node re-identification* and *edge disclosures*.

Research focusing on adding noise to the base graph considers different strategies for edge insertion and edge deletion. Liu and Terzi [16] apply k-anonymity by ensuring that the degree of all nodes is k-anonymous. Zhou and Pei [25] focus on preventing neighborhood attacks by enforcing k-anonymous subgraphs based on a measure of the local neighborhood graph for all nodes. Their method relies on adding edges to the graph to make nodes that have distinct neighborhoods similar to other nodes. Wu et al [22] extend Zhou and Pei's work by introducing the k-symmetry model that accounts for anonymization based on the degrees of each node's neighbors. Similarly, Zou et al. [26] use k-automorphism to make subgraphs k-anonymous. More recently, Cheng et al. [6] developed the k-isomorphism method to preserve privacy at the subgraph level. Bhagat et al. [3] introduce the concept of *label lists* as a potential anonymity mechanism for obscuring the identity of a particular node. Zheleva and Getoor [24] present a number of anonymization strategies for avoiding sensitive edge inference breaches. Tai et al focus on a particular edge breach referred to as a friendship breach [19]. Das et al. [7] present a method for anonymizing social network graphs with weighted edges. Their linear programming anonymization method focuses on anonymizing the edge weights and preserving properties of the graph that are expressible as linear functions of the edge weights. In all these works, the anonymization procedure is applied to the entire graph. In contrast, we propose applying our anonymization procedure to a small subset of the full graph, thereby efficiently obtaining a releasable graph with comparable error.

3 Graph Structure and Privacy Model

3.1 Background

Let $G(V, E)$ represent a simple, undirected graph, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes and $E = \{e_{ij} = (v_i, v_j) | \, v_i \in V \text{ and } v_j \in V, \text{and } i \neq j\}$ is the set of

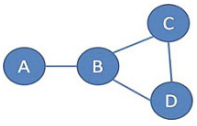


Fig. 1. A sample network graph

edges in G. Given a node v_i in V, its neighbors $N(v_i)$ are the set of vertices adjacent to v_i: $N(v_i) = \{v_j \mid (v_i, v_j) \in E,\ v_j \neq v_i,\ 1 \leq j \leq n\}$. Let $E(N(v_i))$ be the union of the edges between v_i and the nodes in $N(v_i)$ and the edges between the nodes in $N(v_i)$. Then the neighborhood subgraph of vertex v_i is $S(v_i) = \{V_S, E_S |\ V_S \in N(v_i) \cup v_i,\ E_S \in E(N(v_i))\}$.

The degree of a vertex is the size of the neighborhood, $|N(v_i)|$. Betweenness centrality is calculated by computing all of the graph's shortest paths and determining how many shortest paths a given node appears in. Sociologists measure the importance of individuals in a network using different centrality measures, including degree centrality (hubs) and betweeness centrality (brokers) [21]. Computer scientists have used these same measures in different graph mining algorithms.

3.2 Privacy Model

In this work, a data owner is interested in publishing a graph $G' = (V', E')$ that is an anonymized version of G. We assume that there is a bijective function $f : V \rightarrow V'$ that maps every vertex in V to a vertex in V'. We do not, however, require that all edges in E appear in E'. Some edges in E' may also not be present in E.

Most literature focuses on adversarial attacks on three parts of a graph: nodes, edges, and subgraphs. In this paper, our primary focus is on anonymizing nodes. We assume adversaries know the degree of one or more nodes, where the number of nodes known is small, $|V_{known}| << |V|$. However, the adversary does not know the neighborhood subgraph S of any of the nodes in V_{known}. Similar to related literature, k-degree anonymity occurs if at least k nodes have the same degree [16].

Definition 1. ***Node exposure*** *or a node identity breach occurs if any node in G is not k-degree anonymous.*

It is straightforward to determine if a node identity breach occurs using degree sets, where D_a is a set of vertices in G having degree a: $D_a = \{v_i | deg(v_i) = a \mid v_i \in V\}$. We define **D** as the set of all degree sets. For example, in Figure 1 the complete set **D** of degree sets is $D_1 = \{A\}$, $D_2 = \{C, D\}$, $D_3 = \{B\}$. If $k = 2$, then nodes A and B are exposed because D_1 and D_3 each have only one node in their sets.

Definition 2. ***Subgraph exposure*** *occurs if all the neighborhood subgraphs of nodes in a particular degree set D_j are the same.*

This results because the adversary knows $N(v_i)$, but not $S(v_i)$. However, if all the neighborhood subgraphs for a particular degree set are the same, then the adversary

will know $S(v_i)$ with certainty.[1] When we consider subgraph exposure, we must determine if all the subgraph neighborhoods are the same structure for a particular degree set. In other words, are they isomorphic? Determining if different neighborhoods in the graph are isomorphic is expensive to compute. However, because our neighborhood subgraphs are ego networks, we can use some simple social network metrics, e.g. clustering coefficient, to identify degree sets containing exposed neighborhood structures.

The clustering coefficient CC_{v_i} of a vertex v_i is a normalized value that shows how well connected the neighbors of vertex v_i are: $CC_{v_i} = \frac{2|S(v_i)|}{|N(v_i)|*(|N(v_i)|-1)}$ where $|N(v_i)| \geq 2$. The clustering coefficient of a node ranges from 0 (no neighbors connected to each other) to 1 (all neighbors connected to each other). In Figure 1, the clustering coefficient of B, C, and D are 0.333, 1 and 1, respectively.

To understand whether nodes in the same degree set D_a have the same or similar neighborhood structure, we can compare the variance of the clustering coefficient:

$$CC\_dif_a = \begin{cases} 0, if\ var(CC(D_a)) \leq \theta \\ 1, if\ var(CC(D_a)) > \theta \end{cases}$$

where $CC(D_a)$ is the clustering coefficient of each node in the degree set D_a, var is the variance of these values, and θ is the threshold for dissimilarity allowed in the neighborhood. If all the nodes in a degree set must have the exact same neighborhood subgraph to be exposed, then $\theta = 0$ and the exposure occurs with $var(CC\_dif_a) = 0$. However, if k is particularly large and want to extend the definition of subgraph exposure to allow for a very small percentage of the subgraphs for a degree set to not be isomorphic, then $\theta > 0$. Returning to our example in Figure 1, nodes C and D in D_2 have the same connectivity structure, $var(CC\_dif_a) = 0$; therefore, by definition both nodes have subgraph exposure.

Problem Statement: *Given a social network G, we want to publish G', a distorted version of G modified using a set of edge operations such that: 1) each vertex in V is represented in G'; 2) every vertex in V is k-degree anonymous in G'; 3) the degree of nodes that are already k-degree anonymous are not alterable; and 4) reasonable accuracy of different centrality and path measures exists in G'.*

While we focus on degree anonymity to quantify structural uniqueness in this work, any reasonable set of graph properties can be used to define unique parts of a graph, e.g. centrality measures, neighborhood measures, or subgraph structures. Therefore, we also propose a general definition for vulnerable components of a graph as follows:

Definition 3. *In a graph G, a **weak node**, v_w, is a node that is identifiable in the graph based on one or more graph properties. The presence of weak nodes in a graph reduces the overall anonymity of G. We define W as the set of all weak nodes in G. A **weak neighborhood subgraph**, S_{v_i}, occurs when the neighborhood subgraph of v_i is isomorphic to all the other neighborhood subgraphs in $D_{|N(v_i)|}$.*

[1] Note that our adversarial profile and definition of subgraph exposure differ from previous literature. Therefore, unlike previous literature, having the same neighborhood subgraphs leads to exposure.

Algorithm 1. EWNI - Anonymization Approach

1: **Input:** G, k, θ
2: **Output:** G'
3:
4: compute *degree_map*
5: $W = find\_weak\_nodes(G, k, degree\_map)$
6: $graph\_anonymization(W)$
7: $G' = graph\_construction()$
8: **return** G'

4 Anonymization Algorithms

In this section, we present our approach for graph anonymization. As described in Algorithm 1, our general approach is similar to that in [16]. The general approach proposed in [16] gathers the degrees of all nodes in G, identifies which degree sets do not have *k* nodes, and then changes the degree of those nodes to either match the closest degree that is k-degree anonymous or create a new degree set that is k-degree anonymous. After creating the new degree set, they construct a graph G' based on it.

Similarly, we begin by determining the set of weak nodes ($find\_weak\_nodes()$). We then apply a graph anonymization algorithm based on edge modification to only neighborhoods of weak nodes. We consider two strategies, edge insertion and probabilistic edge modification. After graph anonymization, a graph construction step follows that is based on the computed degree sequence. Here we follow the standard procedure proposed in Liu and Terzi [16] and will, therefore, focus on the first two parts of the task in the remainder of this section.

4.1 Finding Weak Nodes and Neighborhood Subgraphs

Our approach for calculating weak nodes is presented in Algorithm 2. For completeness, we also describe how to calculate weak subgraphs. Here, if $|D_i| < k$, a set of vulnerable nodes exists. Therefore, all nodes $v_i \in D_i$ in that degree set are added to a list of nodes (*weak_Da*) identified as weak due to their degree set length. We also track the difference between $|D_i|$ and k, allowing us to later rank the level of weakness of the nodes. When *CC_dif* $< \theta$, all nodes $v_i \in D_i$ are added to a list of node neighborhoods (*weak_cc*) identified as weak due to their *CC_dif* value. This is the $EWNI\_CC\_DIF()$ function in Algorithm 2. Running our weak node identification method returns two lists, one of weak nodes and one of nodes with weak neighborhoods. The EWNI algorithm has a time complexity of $O(|V| \cdot (|V| + |E|))$. However, in the worst case, it can require $O(|V|)$ more disk space. As will be illustrated in section 5, in practice we find that the number of weak nodes and nodes with weak neighborhood subgraphs in a network is a small proportion of the total number of nodes. It also remains small as the size of the network increases.

4.2 Anonymizing *G*

Our localized strategy focuses on changes to only the weak nodes and weak neighborhood subgraphs. The remaining parts of the graph remain the same. While we can our

Algorithm 2. Efficient Weak Node Identification

```
1: function find_weak_nodes(G, k, degree_map)
2:     i ← 1
3:     weak_cc ← {}
4:     weak_Da ← {}
5:     while (i <max_degree(G)) do
6:         D_i ← DEGREE_SET(i, degree_map)
7:         (cc_dif, weak_cc) ← EWNI_CC_DIF(D_i, weak_cc)
8:         if (|D_i| < k) then
9:             weak_Da ← weak_Da + D_i, k − |D_i|
10:    return (weak_cc, weak_Da)
11: function EWNI_CC_DIF(D_a, weak_cc)
12:    if (|D_a| = 0) then
13:        return (0,weak_cc)
14:    if (var(CC(D_a)) > 0) then
15:        return (1,weak_cc)
16:    else
17:        weak_cc ← weak_cc +D_a
18:        return (0,weak_cc)
```

weak node techniques to different anonymization algorithms, we choose to explain the proof of concept using the general framework proposed in [16]. We first describe our approach for anonymization that applies edge insertion to the weak nodes in the graph. We then explain a variation that considers both insertions and deletion.

Weak Node Edge Insertion: Let D_W be the degree sequence of the weak nodes. For this method, we directly apply a greedy algorithm similar to [16] to only weak nodes, W. This algorithm creates a group of the first k highest degree nodes that are not k-degree anonymous and assigns them all the highest degree in the group. The algorithm then computes two costs, the cost of merging the $(k+1)$-th node with the current group and the cost of starting a new group, where the cost is based on the number of edges that need to be inserted in each case. In order to help with the decision, the algorithm looks ahead to k other nodes. The algorithm continues recursively until all the weak nodes are considered. The run time is $O(|W| \cdot k)$. The proof of correctness is straightforward.

Probabilistic Weak Node Anonymity: For this method, we mimic the Weak Node Edge Insertion algorithm described above, but instead of always adding edges to make the k-sized group of nodes k-degree anonymous, we randomly determine whether to add or delete edges. Let p be the probability for deleting an edge and $q = 1 - p$ be the probability for adding an edge. We first sort the degrees of the nodes in W. Then given the sorted set of weak nodes, during each iteration for a size k group we randomly insert or delete edges to the nodes in each weak degree set based on p and q. If the decision is to insert an edge, we insert edges to the nodes in the group until all the degrees of the nodes are equal to the highest degree in the group. If the decision is to delete an edge, we delete edges and decrease the degrees in the group until all the degrees are equal to the lowest degree in the group. This process continues until all the weak degree

sets are members of k-degree anonymous degree sets. This method combines both a deterministic and probabilistic adding of noise to obtain k-degree anonymity.

Proof. (Sketch) Nodes that are not in D_W are already k-degree anonymous by definition. Since the nodes in D_W are considered in groups of k and, in each group, all nodes are assigned the same degree, these nodes become k-degree anonymous. Consequently, the produced graph (using [16]) satisfies k-degree anonymity as it consists of k-degree anonymous nodes.

Probabilistic Weak Neighborhood Subgraph Anonymity: While our focus is on node exposure, our probabilistic weak node anonymity algorithm could be extended for neighborhood subgraphs by only considering nodes and edges that are in W based on both $weak\_cc$ and $weak\_Da$. Generally, the algorithm would focus on adding and removing edges that exist between neighbors that are weak. We save this analysis for future work.

5 Experiments

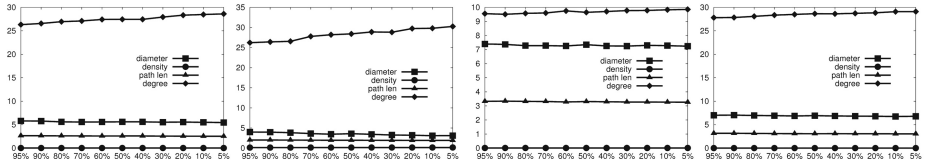
In this section we evaluate our approach in terms of 1) graph edit distance between G and G'; 2) accuracy of graph properties; and 3) efficiency of anonymization on five real world networks (graph properties shown in Table 1). The *PolBlogs* graph represents a network of hyperlinks between weblogs about US politics in 2005 [1]. The *Jazz* graph shows connections between different jazz musicians [8]. The *Email* graph is a network of email interchanges between members of the University Rovira i Virgili [9]. The *Wiki* graph is a network representing user participation in different elections [15]. Last, the *Facebook* graph is from a crawl of a subset of public Facebook pages. Because this crawl followed a snowball sampling protocol with multiple seeds, we remove nodes that only have a single degree since they are an artifact of the sampling approach.

Sensitivity Analysis of Probabilistic Anonymization: Before comparing our anonymization algorithm to other algorithms, we want to understand the sensitivity of the percentage of additions vs. deletions of edges. In other words, does the actual percentage of additions versus deletions affect the accuracy of the graph properties of interest?

Figure 2 shows the percentage of error introduced for our probabilistic method when we vary the percentage of edges that are inserted and deleted. Each experiment was run 10 times and the average results are presented. The x-axis shows the probability of deleting an edge as opposed to inserting it. The y-axis shows the amount of error introduced for each measure. This figure shows that the amount of error introduced is relatively constant, but does rise some as the probability of insertions becomes higher than deletions. Therefore, for the remainder of our experiments, we will set the probability of deleting an edge to 95% and the probability of adding an edge to 5%.

Table 1. Graph properties of data sets

Network Name	Nbr of Vertices	Nbr of Edges	Average Degree	Average Betweenness
Jazz	198	2742	27.7	121.65
Email	1133	5451	9.62	1475.01
PolBlogs	1222	16714	27.36	1060.76
Wiki	7115	100762	28.32	7884.65
Facebook	40531	157054	7.75	71262.98

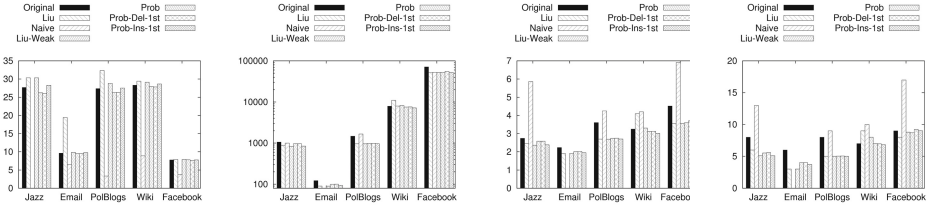


(a) Political Blogs (b) Jazz (c) Email (d) Wiki

Fig. 2. Graph properties percentage error as the probability of deletion decreases ($k = 3$)

Accuracy of Graph Properties: We now consider the accuracy of the released, perturbed graphs. The methods we consider are as follows: the original Liu and Terzi algorithm [Liu], removal of weak nodes from the graph [Naïve], Liu and Terzi applied to only weak nodes [Liu-Weak], our probabilistic anonymization [Prob], [Prob] with the first operation forced to be a deletion [Prob-Del-1st] or with the first operation forced to be an insertion [Prob-Ins-1st]. A naïve approach for anonymizing weak nodes is to produce a graph G' that simply removes each weak node, v_w in W and incident edges to nodes in $N(v_w)$ from G. In the last two methods, to reduce the variance of the basic probability anonymization procedure, we force the first operation to be consistent across runs. This is necessary because of the generally large variance in the degrees of the nodes in the first k weak nodes. Because they are the highest degree nodes, deleting or inserting has the largest amount of impact on this 1st group of nodes. Forcing the first action to always be the same reduces the variance to under 5%. The different algorithms were run ten times and the average error introduced by each method for $k = 10$ is shown in Figure 3. If a bar is missing, then G' was disconnected. We measured the error for varying values of k and the results were similar to those when $k = 10$. The x-axis shows the different data sets and the y-axis shows the value of the graph property as computed on the original graph G [Original], and on the anonymized graph G' produced by each tested method.

From Figure 3 we see that the best performer is [Prob-Del-1st] and the naïve removal of weak nodes generally results in the highest error across all data sets and measures. The exception to that is the betweenness calculation, where it generally outperforms the other methods. We suspect this occurs because the other approaches are adding at least a small fraction of edges to the graph, thereby creating new paths and potentially reducing the number of shortest paths each node lies on. With the exception of the political blogs data set, applying Liu and Terzi to just the weak nodes introduces less error than



(a) Average Degree (b) Average Betweenness (c) Average Path Length (d) Diameter

Fig. 3. Graph properties for different anonymization techniques ($k = 10$)

Table 2. Graph edit distance comparison to the algorithm of [Liu] with $k = 3$ and $k = 10$

	[Liu]	[Liu-Weak]	[Prob]
Jazz	0	-669, -13	-681, -324
Email	0	-84, -114	-131, -305
PolBlogs	0	-3529, -1242	-3529, -2547
Wiki	0	-11089, -6554	-11831, -10501
Facebook	0	-1138, -596	-2297, -6347

applying the algorithm to the entire graph. In general, our probabilistic methods perform comparable or better than the other methods on these data sets.

Since our methods do not guarantee a minimum number of edge modifications to G, we also compare the graph edit distance between G and G'. We set the baseline to be the Liu and Terzi method and compare the graph edit distances based on that method. Table 2 shows these results for each of the data sets averaged over 10 runs with $k = 3$ and $k = 10$, respectively. The table should be read as follows. For the Jazz dataset, the [Liu-Weak] algorithm needs 669 fewer graph edit operations than [Liu] when $k = 3$ and 13 fewer graph edit operations when $k = 10$. Looking at the entire table, we see that [Prob] has the smallest edit distance in all cases.

Finally, Singh and Zhan propose a measure called topological anonymity that uses node and subgraph exposure to quantify the level of risk associated with releasing a particular graph G' [18]. It is computed as follows:

$$ta = \frac{\sum_{i=1}^{\max(\deg(G))} \left[(|D_i| \times CC\_dif_i) - \begin{cases} 0, \text{if } |D_i| \geq k \\ |D_i|, \text{if } |D_i| < k \end{cases} \right]}{n}$$

where k represents the required number of nodes in a degree set and n is the number of nodes in G. The ta score, ranging from -1 to 1, with -1 indicating that the graph is highly susceptible to both node and neighborhood subgraph exposure, and 1 indicating that the nodes are well anonymized.

As another method to quantify the level of anonymity of G' compared to the other approaches, we compute the topological anonymity of G' for the different methods. Figure 4 (shown below) illustrates the improvement in the ta score after anonymization. All the algorithms improve with the exception of the naïve one.

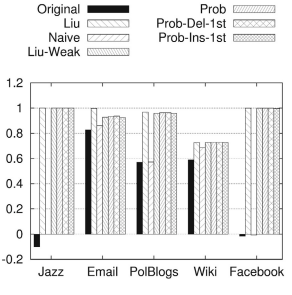


Fig. 4. Topological anonymity comparison

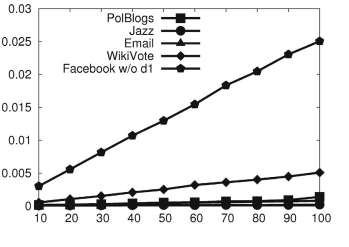


Fig. 5. Run times (seconds) as the percentage of weak nodes increases

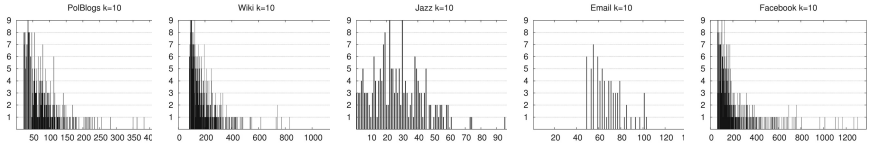


Fig. 6. Weak node distribution for $k = 10$

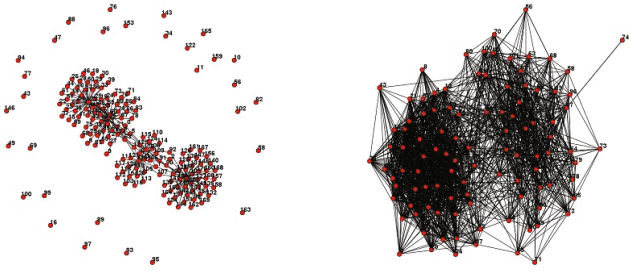
Weak Nodes Distribution: We now compare the distribution of weak nodes to determine if they are similar or different across our data sets. Figure 6 shows the distribution of weak nodes when $k = 10$. The x-axis shows the degree of each weak node, and the y-axis shows the number of nodes with that degree. In all cases, the maximum of the y-value is k-1, since degrees that have k or more nodes are not considered weak. These graphs show a number of interesting properties about the distribution of weak nodes. The main similarity among the graphs is that the far left side does not have any weak nodes, indicating that low degree nodes are generally k-degree anonymous. The Jazz network is the one exception. Second, we observe that as the degree increases, the number of nodes with that degree decreases and the bars become sparser as we move from left to right along the x-axis. This may be an indication that there are fewer nodes with high degree and that those nodes are not always weak. Both of these observations support theories that state social networks often follow a power law distribution. Finally, the figures highlight that the distribution of the weak nodes differs from data set to data set.

In addition to considering the number of weak nodes with each degree, we are also interested in the subgraphs formed by these weak nodes. They represent the portion of the graph that is most vulnerable to attack. Figure 7 shows that weak nodes tend to be highly connected, with all weak nodes in the Political Blogs network contained in one component, and the majority of weak nodes in the Facebook network contained in one component. We measure the vulnerability associated with subgraphs by considering their size and connectivity to other weak nodes. The *subgraph vulnerability index* is defined as: $SVI = \frac{|S_w|}{|W| \times |C_w|}$, where $|S_w|$ is the number of nodes in the weak subgraph(s), $|W|$ is the total number of weak nodes, and $|C_w|$ is the number of weak components. SVI is 1 and 0.822 for Political Blogs and Facebook, respectively.

Efficiency Results: Table 3 compares the run time of the Liu and Terzi algorithm to our delete first probabilistic anonymization algorithm with the probability of deleting

Table 3. Run Time Comparison (milliseconds)

Network Name	Preprocessing		Anonymization		Total	
	Liu	Weak Prob	Liu	Weak Prob	Liu	Weak Prob
Jazz	0.036	10.956	0.451	0.146	0.487	11.102
Email	0.062	25.239	8.423	0.089	8.485	25.328
PolBlogs	0.065	97.48	7.326	0.284	7.391	97.764
Wiki	0.25	1705.361	171.962	0.463	172.212	1075.824
Facebook without Degree 1	1.453	5187.103	4852	0.479	4853.453	5187.582
Facebook	8.484	73967.523	209784.52	1.125	209793.004	73968.648
Facebook doubled	17.217	155035.336	861238.351	5.277	861255.568	155040.613
Facebook quadrupled	33.781	319397.488	4392412.422	6.196	4392446.203	319403.684



(a) Facebook (b) Political Blogs

Fig. 7. Weak Subgraphs

an edge set to 95% and the probability of adding an edge set to 5% when $k = 10$. The second and third columns compare the preprocessing cost of the two approaches. The next two columns compare the anonymization approaches followed by the total run time in milliseconds.

Our preprocessing cost is higher than the original Liu and Terzi algorithm and is always the dominant cost of the approach. The Liu and Terzi algorithm precomputes a degree set map. Our algorithm precomputes a degree set map and clustering coefficients. While for small graphs our preprocessing cost is high, as the size of the graph increases, it increases linearly, resulting in an overall run time that is still less than the overall run time of the Liu and Terzi algorithm.

Table 3 shows that our anonymization run time increases sublinearly and is orders of magnitudes faster than Liu and Terzi as the size of the graph increases. This is because our cost is related to the number of weak nodes in the graph, which is a small fraction of the total number of nodes. To evaluate the run time of the anonymization algorithm as the number of weak nodes increases, we simulate an increase in the number of weak nodes for each data set. Figure 5 shows that the run time increases linearly. Therefore, even when the number of weak nodes increases, the algorithm performs efficiently.

6 Conclusions

This paper investigates anonymization of social graphs for data publishing. Current approaches apply anonymization techniques to the entire graph. We introduce the concept of weak nodes and propose approaches that only anonymize those nodes. We show that the number of weak nodes tends to be small in real world networks and anonymization focusing on these nodes is orders of magnitude faster and maintains the same level of accuracy and a low edit distance when compared to traditional methods. By not distributing the noise uniformly across the graph, more of the original distribution and properties are well maintained. Future work will investigate weak subgraph anonymization and try to understand the impact of releasing a partially generalized graph.

Acknowledgments. We would like to give a special thanks to our paper reviewers. Their insightful comments help improve the paper significantly. Finally, the experiments reported in this work were conducted at Georgetown University.

References

1. Adamic, L., Glance, N.: The political blogosphere and the 2004 US Election. In: WWW 2005 Workshop on the Weblogging Ecosystem (2005)
2. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In: WWW (2007)
3. Bhagat, S., Cormode, G., Krishnamurthy, B., Srivastava, D.: Class-based graph anonymization for social network data. In: VLDB (2009)
4. Bonchi, F., Gionis, A., Tassa, T.: Identity obfuscation in graphs through the information theoretic lens. In: ICDE (2011)
5. Campan, A., Truta, T.: Anonymization of centralized and distributed social networks by sequential clusterings. In: PinKDD (2008)
6. Cheng, J., Fu, A., Liu, J.: K-isomorphism: privacy preserving network publication against structural attacks. In: SIGMOD (2010)
7. Das, S., Egecioglu, O., Abbadi, A.: Anonymizing weighted social network graphs. In: ICDE (2010)
8. Gleiser, P., Danon, L.: List of edges of the network of jazz musicians. Adv. Complex Systems 6, 565 (2003)
9. Guimera, R., Danon, L., Diaz-Guilera, A., Giralt, F., Arenas, A.: Network of emal interchanges. Physical Review E 68 (2003)
10. Hanhijarvi, S., Garriga, G., Puolamaki, K.: Randomization techniques for graphs. In: SDM (2009)
11. Hay, M., Miklau, G., Jensen, D.: Enabling accurate analysis of private network data. Chapman & Hall, CRC Press (2010)
12. Hay, M., Miklau, G., Jensen, D., Towsley, D.: Resisting structural re-identification in anonymized social networks. In: VLDB (2008)
13. Hay, M., Miklau, G., Jensen, D., Weis, P., Srivastava, S.: Anonymizing social networks. Technical Report 19, University of Massachusetts (2007)
14. LeFevre, K., Terzi, E.: Grass: Graph structure summarization. In: SDM (2010)
15. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Signed networks in social media. In: ACM Conference on Human Factors in Computing Systems (2010)
16. Liu, K., Terzi, E.: Towards identity anonymization on graphs. In: SIGMOD (2008)
17. Narayanan, A., Shmatikov, V.: De-anonymizing social networks. In: IEEE Symposium on Security and Privacy (2009)
18. Singh, L., Zhan, J.: Measuring topological anonymity in social networks. In: IEEE Conference on Granular Computing (2007)
19. Tai, C.-H., Yu, P.S., Yang, D.-N., Chen, M.-S.: Privacy-preserving social network publication against friendship attacks. In: KDD (2011)
20. Tassa, T., Cohen, D.: Anonymization of centralized and distributed social networks by sequential clusterings. In: TKDE (2011)
21. Wasserman, S., Faust, K.: Social network analysis: methods and applications. Cambridge University Press, Cambridge (1994)
22. Wu, W., Xiao, Y., Wang, W., He, Z., Wang, Z.: k-symmetry model for identity anonymization in social networks. In: EDBT (2010)
23. Ying, X., Wu, X.: Randomizing social networks: a spectrum preserving approach. In: SDM (2008)
24. Zheleva, E., Getoor, L.: Preserving the privacy of sensitive relationships in graph data. In: KDD 2007 Workshop on Privacy, Security, and Trust (2007)
25. Zhou, B., Pei, J.: Preserving privacy in social networks against neighborhood attacks. In: ICDE (2008)
26. Zou, L., Chen, L., Ozsu, M.: KAutomorphism: A general framework for privacy preserving network publication. In: VLDB (2009)