k-Anonymization of Social Networks By Vertex Addition

Sean Chester[1], Bruce Kapron[1], Ganesh Ramesh[2], Gautam Srivastava[1], Alex Thomo[1], and S. Venkatesh[1]

[1] University of Victoria, Victoria, B.C.
[2] Yahoo! Labs, U.S.A.

Abstract. With an abundance of social network data being released, the need to protect sensitive information within these networks has become an important concern of data publishers. In this paper we focus on the popular notion of k-anonymization as applied to node degrees in a social network. Given such a network N, the problem we study is to transform N to N', such that the degree of each node in N' is attained by at least $k-1$ other nodes in N'. Apart from previous work, we permit modifications to the node set, rather than the edge set, and this offers unique advantages with respect to the utility of the released anonymized network. We study both vertex-labeled and unlabeled graphs, since instances of each occur in real-world social networks. Under the constraint of minimum node additions, we show that on vertex-labeled graphs, the problem is NP-complete. For unlabeled graphs, we give an efficient (near-linear) algorithm and show that it gives solutions that are optimal modulo k, a guarantee that is novel in the literature. Additionally, we demonstrate empirically that commonly-studied structural properties of the network, such as clustering coefficient, are quite minorly distorted by the anonymization procedure.

Keywords: privacy, k-anonymization, complexity, dynamic programming, social networks

1 Introduction

Social networks are a natural phenomenon and as such have been studied for a long time by sociologists, anthropologists, and biologists. But the recent explosion of web applications that have social links either implicit (e.g., Amazon and IMDB) or explicit (e.g., Facebook and Twitter) has substantially simplified the difficult preprocessing step of producing their underlying graphs. Consequently, the opportunity for analyzing and mining these networks has become widespread.

From a high level view, there are two general families of methods for achieving network data privacy. The first family encompasses "data anonymization" methods. These methods first transform the data and then release them. The data is thus made available for unconstrained analysis. The second family encompasses "privacy-aware computation" methods, which do not release data, but, rather, only the output of an analysis computation. The released output is such that it is very difficult to infer from it any information about an individual input datum. The relatively recent differentially-private methods (cf. [3–5, 11]) all belong in this family. Both families of methods have natural pros and cons. Methods in the first family give complete freedom to the analysts to perform any analysis they wish on the released data. However, they can be more vulnerable to attack. On the other hand, methods in the second family can protect the data better,

but in the end do not release data, only carefully computed private outputs of specific computations. This obviously limits further analysis. Our approach belongs in the first family. Our goal is to anonymize social networks without significantly distorting them.

Research into network anonymization has gathered some momentum in the past five years, with an evolution towards stronger adversarial models, that is, towards assuming the adversary has more knowledge. While requisite, it is outpacing research into protecting against the adversaries that have already been proposed; our hardness results in this paper combined with the hardness result of Zhou and Pei [18] demonstrate that understanding simpler models is very important because extending them could potentially be infeasible. Additionally, in some settings, simpler adversarial models may be sufficient and if the anonymization procedures have different complexities, then the ability to consider trade-offs is paramount. Thus, in this research we return to a more modest adversary, one who knows only degree-based information, so as to more thoroughly develop the foundations on which research into stronger adversarial models and the interplay between levels of adversarial knowledge can be built. While most anonymization research studies unlabeled graphs, existing algorithms do not provide optimality guarantees, so the only known complexity results are for vertex-labeled graphs. Here, we study both important classes.

In previous studies, networks are typically anonymized by exclusively introducing new edges into the network. This approach is justified under the assumption that one does not wish to add new entities into the network, but we challenge this assumption. Even for microdata, the analysis that one wishes to do with the network data is at the aggregate level, so the introduction of new nodes does not necessarily have an adverse affect. To the contrary, adding new nodes with similar properties could better preserve aggregate measures than will distorting the existing nodes.

In fact, one ought to consider the intended use of the anonymized network prior to conducting the anonymization, because this affects which characteristics should be preserved. To facilitate this choice, it is important to develop alternate approaches with respective advantages. We introduce the natural complement to current k-anonymization, an approach of augmenting the network with new vertices which are connected to themselves and to the original graph. In this manner, one guarantees, for example, not to increase the size of any clique by more than one, and is very unlikely to do anything but introduce new 2- and 3-cliques. If large cliques are of particular interest to an analyst, this is clearly preferable, because adding edges among original vertices can produce false positives. Alternatively, for analysis tasks that involve monotone properties such as independent sets, the distortion is more controlled in that vertex addition can only introduce false positives, whereas edge addition can introduce false negatives, too.

1.1 Related Work

The most closely related work to this paper is the research of Liu and Terzi [10] who also study the same adversarial model. The distinction from our work is that they constrain the problem by assuming an immutable vertex set, whereas we place constraints on the edge set. This distinction renders the work incomparable, each preserving different structural properties as stated above.

Zhou and Pei [18] define a notion of k-anonymity on graphs so that nodes in an anonymized group will have isomorphic neighbourhoods. They show that anonymizing a graph under their definition using a minimal number of edge additions is NP-hard. Other landmark papers in the field [7, 17, 14, 2, 16] have introduced models of protecting from progressively stronger adversarial knowledge. These are summarized in Table 1. For all of these adversarial models, it is important to understand the challenges in producing networks anonymized with respect to those models. Deepening the understanding of k-anonymity here in this paper is an important step in that direction.

The other work that is very closely related to our research is of König, Erdős, and Kelly [6, 9], since our work extends their graph theoretic results. König showed that, given a graph G with maximum degree d, it is always possible to make a d-regular graph H by adding vertices and adding edges whose endpoints must include at least one new vertex. In a subsequent paper, Erdős and Kelly gave an efficient algorithm to determine how many vertices must be added to obtain such a graph H. We generalize this problem with two relaxations. First, we may not require that all nodes of a graph be anonymized. For example, in the Amazon database, there are two types of vertices, customers and products, and the database owner could be interested only in anonymizing the customer vertices. Second, we k-anonymize the graph for an arbitrary k (which we typically assume to be some reasonably small value $\ll d$).

Table 1. Summarisation of Related Work

Authors	Adversarial Model	Graph Type	Permitted Operations
König, Erdős & Kelly [6, 9]	n-Anonymization	Unlabeled	Vertex/Edge Addition
Liu and Terzi [10]	k-Anonymization	Unlabeled	Edge Addition/Removal
Zhou & Pei [18]	k-N'hood Anon.	Vertex-Labeled	Edge Addition
Cheng et al. [2]	k-Isomorphism	Unlabeled	Edge Addition/Removal
Hay et al. [7]	Automorphic Equiv.	Unlabeled	Label Modifications
Wu et al. [16]	k-Symmetry	Unlabeled	Vertex/Edge Addition
Tripathy and Panda [15]	k-N'hood Anon	Vertex-Labeled	Edge Addition
Kapron et al. [8]	k-Label Seq. Anon.	Vertex/Edge Labeled	Edge Addition
This Paper	k-Anonymization	Vertex-Labeled **and** Unlabeled	Vertex Addition

1.2 Our Results

We study the problem of k-anonymizing a graph by augmenting it with additional vertices, both for vertex-labeled and for unlabeled graphs. Our main results are as follows:

- We prove that on vertex-labeled graphs, k-anonymization with a constant number of vertex additions is NP-complete by giving a reduction to a hard table anonymization problem (§ 2).
- For unlabeled graphs, we introduce an efficient (i.e., $\mathcal{O}(nk)$) k-anonymization algorithm based on dynamic programming and prove that it produces a solution that is optimal modulo k (§ 3).
- We perform experiments with several well-known network datasets to demonstrate empirically that our vertex-addition approach to k-anonymization quite minimally distorts the original graph with respect to standard parameters like clustering coefficient, average path length and connectivity, even as k approaches d (§ 4).

2 A Hardness Result for Vertex-Labeled Graphs

In this paper, we consider simple, undirected graphs in two settings: one in which the vertices are labeled and the other in which they are not. The labeled graphs correspond to social networks in which the nodes have identifiers or attributes. In this section we define the problem of k-degree-anonymizing vertex-labeled graphs and provide a result that the problem is NP-complete.

We start with the definition of a vertex-labeled graph and a label sequence.

Definition 1. *A vertex-labeled graph is a simple, undirected graph $G = (V, E, L, \mathcal{L})$ where V is a set of vertices, $E \subseteq V \times V$ is the set of edges, L is a set of labels and $\mathcal{L}$ is a labeling function, $\mathcal{L} : V \to L$, that assigns a label to every vertex in V.*

Definition 2. *For $v \in V$, we say that $S_v = (l_1, l_2, \ldots, l_m)$ is a label sequence for v if it corresponds to some ordering of the labels of v and the vertices that are adjacent to v. We will consider label sequences of vertices to be equivalent up to reordering.*

Then, k-anonymity for labeled graphs relates to the uniqueness of label sequences:

Definition 3. *Given a vertex-labeled graph $G = (V, E)$, a subset $X \subseteq V$ of vertices is k-anonymous in G if for every vertex $v \in X$, there are at least $k - 1$ other vertices in X whose label sequence is the same as the label sequence of v.*

Thus, the vertex-labeled problem we study in this paper is defined as:

LABELED SUBGRAPH ANONYMIZATION

Input: A vertex-labeled graph $G = (V, E, L, \mathcal{L})$, a set $X \subseteq V$ of vertices, integers t and $k \geq 3$.
Question: Is there a vertex-labeled graph $G' = (V \cup V', E \cup E', L', \mathcal{L}')$ such that $|V'| \leq t$, $E' \subseteq (V \times V') \cup (V' \times V')$, $L'_{|V} = L$, $\mathcal{L}'_{|\mathcal{V}} = \mathcal{L}$ and X is k-anonymous in G'?

That is, can we k-anonymize X by adding at most t new labeled vertices? New edges are allowed between an old vertex and a new vertex or between new vertices.

Theorem 1. *Labeled Subgraph Anonymization is NP-complete.*

The proof of the hardness result depends on building a polynomial time reduction from the k-ATTRIBUTE-ANONYMITY table anonymization problem that was demonstrated by Meyerson and Williams [12] to be NP-hard. However, in the interest of space, we omit the details of the proof. Similar to the previous section, we start with some definitions to clarify the problem and introduce some notation in Table 2.

3 An Efficient Algorithm for Unlabeled Graphs

In the section we consider another type of graphs, those which do not have labels on the vertices, and give an efficient algorithm for anonymizing them.

Definition 4. *Given a graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ and $d(v_i) = |\{u \in V : (u, v_i) \in E\}|$, the degree sequence of G is defined to be the sequence $(d(v_1), d(v_2), \ldots, d(v_n))$.*

Table 2. Notation we use to describe unlabeled graphs

Name	Notation	Definition
Degree of u	d(u)	$\|\|\{v \in \mathrm{V} : (u,v) \in \mathrm{E}\}\|\|$
Start Index of u	$start(u)$	First vertex in the same partition as u
Deficiency of u	$def(u)$	$d(start(u)) - d(u)$
Total Deficiency	N/A	$\sum_{v \in V} def(v)$
Max Deficiency	N/A	$max_{v \in V} def(v)$
i'th Vertex	$v(i)$	Vertex with the $i'th$ highest degree, except for arbitrary tie-breaks

Throughout this paper, we assume degree sequences to be sorted in descending order for the simplicity in describing our algorithm. For unlabeled graphs, k-anonymity is then as defined by Liu and Terzi [10].

Definition 5. *Given a graph $G = (V, E)$, a subset $X \subseteq V$ is k-anonymous in G if for every $v \in X$, there are at least $k - 1$ other vertices in X with the same degree as v.*

Definition 6. *A partitioning of a degree sequence $(d_1, d_2, \ldots, d_n)$ is an ordered collection of $i + 1$ disjoint subsequences of size at least k, given as $((d_1, d_2, \ldots, d_{k+c_1}), (d_{k+c_1+1}, \ldots, d_{2k+c_1+c_2}), \ldots, (d_{ik+c_1+\ldots+c_i+1}, \ldots, c_n))$, with all $c_j \geq 0$.*

The anonymization algorithm, given in Algorithm 1, infers a polynomial time optimal algorithm for the anonymization of just the original graph, but we leave the inference to the reader, given space constraints. The eventual outcome of the entire algorithm is that the newly added vertices are anonymized as well.

3.1 Partitioning the Original Graph's Vertices

The first step of our algorithm is to identify which vertices of the original graph should be anonymized into the same equivalence class by partitioning the degree sequence into subsequences of length at least k. This recurrence differs from the similar degree sequence anonymization of Liu and Terzi [10], because our problem requires producing a resultant partitioning that minimizes the max deficiency rather than the total deficiency. Two simple propositions are thus very useful:

Proposition 1 *The deficiency of a subsequence containing a highest degree of d_i and a smallest degree of d_j will be less or equal to the deficiency of any subsequence containing d_i and any d_{j+c} or containing d_j and any d_{i-c}, $\forall c \in \mathbb{N}$.*

Proposition 2 *For any subsequence $(d_i, \ldots, d_{2k+c})$, the max deficiency is greater or equal to the subsequences $(d_i, \ldots, d_{k+c'})$, $(d_{k+c'+1}, \ldots, d_{2k+c})$. That is to say, it never produces a higher max deficiency when one splits a subsequence.*

Algorithm 1 k-Degree-Anonymization of Unlabeled Graphs

1: Optimally partition degree sequence of G (§3.1)
2: Augment graph with m (or so) dummy vertices (§3.2)
3: Connect original graph vertices to new dummy vertices (§3.2)
4: Insert inter-dummy-vertex edges to anonymize dummies, too (§3.3)

Both propositions follow trivially from the fact that the degree sequences are assumed to be sorted and that max deficiency (i.e., difference) is transitive. We use them to produce an incremental algorithm described by the following recursion. Let

$$\mathrm{A} = \max(\mathcal{C}(1, x-k), \Delta(x-k+1, x))$$
$$\mathrm{B} = \max(\mathcal{C}(1, \mathcal{S}(x-1)), \Delta(\mathcal{S}(x-1)+1, x))$$

Then

$$\begin{aligned}
\Delta(x,y) &= d_x - d_y \\
\mathcal{S}(x) &= 1, \text{ if } x < 2k. \\
\mathcal{S}(x) &= \mathcal{S}(x-1), \text{ if } x \geq 2k \text{ and } \mathrm{A} < \mathrm{B}. \\
\mathcal{S}(x) &= x-k+1, \text{ if } x \geq 2k \text{ and } \mathrm{A} \geq \mathrm{B}. \\
\mathcal{C}(1,x) &= \Delta(1,x), \text{ if } x < 2k. \\
\mathcal{C}(1,x) &= \min(\max(\mathrm{A},\mathrm{B})), \text{ if } x \geq 2k.
\end{aligned}$$

Stated more intuitively, the algorithm computes the ideal partitioning from the left by incrementally adding the next degree to the right of the sequence. The Δ function computes the deficiency of a particular partition; the $\mathcal{S}$ function keeps track of where x's partition starts, should x be the smallest degree in it; and the $\mathcal{C}$ function computes the overall max deficiency (cost) of the best possible partitioning up to the $x'th$ element.

If there are fewer than $2k$ degrees in the sequence, there is not any choice but to group them all together. If, however, there are more than $2k$ degrees, then there are two ways in which the partitioning can be extended. Either the new rightmost, smallest degree can be added to the rightmost partition, or a new partition can be formed with the $k-1$ rightmost degrees and the new degree.

It is a result of Proposition 1 that, in general, the new degree will clearly be added to the right. Adding the new degree to any other partition would produce a greater or equal max deficiency.

For the example graph with a degree sequence of $(5,3,3,2,1,1,1)$, the best partitioning of this degree sequence is $((5,3,3),(2,1,1,1))$, as given when the recursion is evaluated. This algorithm yields the following lemma:

Lemma 1. *The degree sequence partitioning of an unlabeled graph can be solved optimally for vertex addition in $\mathcal{O}(n)$ time and $\mathcal{O}(n)$ space.*

It is important to note that the $total\ deficiency$ indicates precisely how many edges need to be added in order to anonymize the original graph vertices:

Lemma 2. *The $total\ deficiency$ of an optimal degree sequence partitioning into subsequences of length $\geq k$ and $< 2k$ is upper-bounded by $(n-1)(2k-1)$.*

3.2 Anonymizing the Original Graph's Vertices

To anonymize the original graph's vertices, we must address both the $max\ deficiency$ and the $total\ deficiency$. To balance out a $max\ deficiency$ of m requires adding at least m dummy vertices. Ergo, we typically add exactly m additional vertices; but

occasionally, as we detail later, we augment the graph with up to $m'+1$ vertices instead, where m' is the larger of m and k.

There is a nice characteristic of our interim solution after the dynamic programming. Effectively, we need to come up with a bipartite graph between original nodes and additional nodes, starting from scratch. We proceed as follows. We order the m additional vertices. We then connect the first $def(v(1))$ additional vertices to the original vertex $v(1)$, the next $def(v(2))$ additional vertices to $v(2)$, and so on until all m additional vertices have an edge. This process ends at some $v(i)$.

We continue with another iteration, this time starting at $v(i)$ and with subsequent iterations until we have satisfied the deficiency of every node in the original graph.

Because of the nature of this *cycling* procedure, always adding an edge to an additional vertex that has not yet been visited on that particular iteration, we can guarantee that, if d iterations are required, some $m - x$ additional vertices will have degree d and the remaining x will have degree $d - 1$, for some $0 \leq x < m$. Because we serviced each original vertex in turn, no edge can be accidentally added twice.

3.3 Ensuring the k-Anonymity of the Additional Vertices

The last detail is to ensure that all the new vertices are themselves k-anonymous. Recall that $m-x$ of the new vertices have degree d and the other x have degree $d-1$, for some $0 \leq x < m$. Furthermore, d and x can both be computed in advance of the edge addition step, since $d = floor(\frac{total\ deficiency}{m})$ and $m - x = total\ deficiency\ (mod\ m)$. As a first recourse, if groups of vertices of degree d and $d - 1$ are both present in the set of anonymized original nodes, nothing needs be done.

If either d or $d - 1$ is not present in the anonymized degree sequence, we explicitly anonymize the new vertices. This scenario can be detected after partitioning the degree sequence, and we then choose to instead augment the graph with $m' = min(m, k)$ additional vertices. This ensures that if we construct H such that all additional vertices have the same degree, they will themselves form a k-anonymous group.

For the x vertices with degree $d - 1$, we randomly pair them and add an edge between each pair. If x is even, this is sufficient: all m' additional vertices have degree d.

If, instead, x is odd, then this pairing will leave out one last vertex, call it r. If $m'-1$ is even (and at least 2), then we can add an edge from r to each of two other additional vertices so that all three have degree $d+1$. The remaining $m'-3$ vertices with degree d can then all be paired off again (since $m'-3$ is even) and all additional vertices will be anonymized with degree $d+1$. If, on the other hand $m'-1$ is odd, then we simply use an extra additional vertex ($m'+1$) at the beginning of the second phase of the algorithm.

Consequently, we have the following theorem:

Theorem 2. *Our algorithm can k-anonymize a graph by introducing a number of additional vertices optimal modulo k in $\mathcal{O}(nk)$ time and $\mathcal{O}(n)$ space.*

4 Experimental Results

In the previous section, we presented an algorithm and proved its asymptotic efficiency and worst-case optimality. In the introduction, we mentioned that certain structural

Table 3. Structural Properties of the Datasets

Graph	Nodes	Edges	APL	CC
Enron	36692	183831	3.39	0.09
Power Grid	4941	6594	18.99	0.10
Net Science	1589	2742	0.35	0.69

properties of graphs are either unaffected, or controllably affected, by a vertex-addition approach. Here we illustrate that on standard benchmark datasets, other commonly studied aggregate structural properties are quite minorly distorted by anonymization, even as k grows quite large. This is important because preserving privacy (through k-anonymity) is not especially useful if it results in tremendously misleading output.

4.1 Metrics and Setup

Datasets We select three diverse datasets, ranging from a power grid network which models the connectivity of generators and substations to an email communication network in a company (namely, Enron). The dataset properties are shown in Table 3.

Metrics We measure the distortion introduced by the algorithm via metrics, defined below, which are commonly studied in the social network literature [1].

1. Clustering Coefficient (CC): A measure of *triadic closure*. Social networks are known to have significant triadic closure (friends of a person are also likely to know each other). More formally, for all ordered triples $u, v, w \in V$,

$$CC = \frac{||\{u, v, w \in V : (u, v) \in E \wedge (u, w) \in E \wedge (v, w) \in E\}||}{||\{(u, v, w \in V : (u, v) \in E \wedge (u, w) \in E\}||}$$

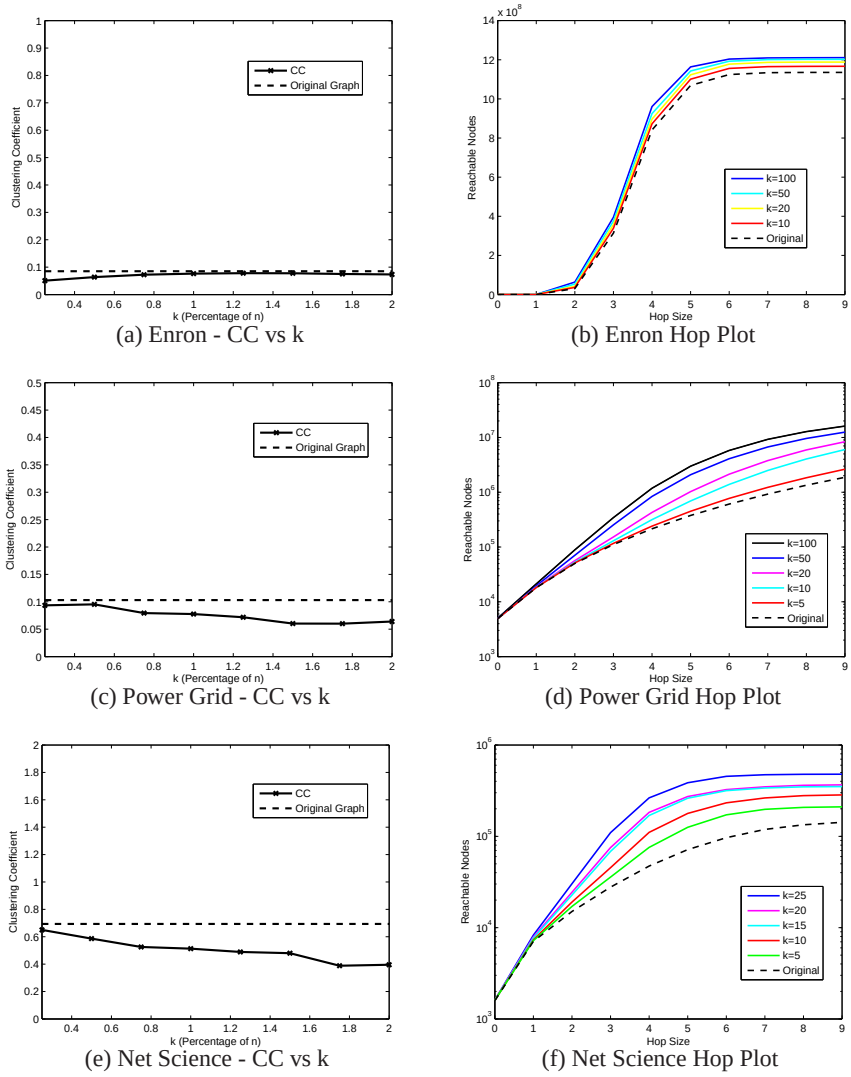
2. Hop Plot: The connectivity of a graph can be modeled using a *hop plot*. A hop plot studies reachability for each path length k. For k, the hop plot displays, summed over all vertices, the number of nodes reachable using paths of length at most k. The maximum value for any value of k is n^2 where n is the number of vertices in the graph. The smallest value of k for which the maximum value of n^2 is reached is the *diameter* of the social network, the path length using which any two nodes in the graph can reach each other. Changing or distorting the connectivity of a graph drastically changes the hop plot shape. This is the main motivation behind studying these plots.

3. Average Path Length (APL): The expected path length between any two randomly chosen connected vertices. This metric is highly relevant as it is directly related to the *six degrees of separation* in social networks [13]. It can be read from a hop plot, but is interesting in its own right. Define a predicate $C(u, v)$ to be *true* if u and v are connected in the graph and $false$ if they are not connected. Define $CP = \{(u, v) : C(u, v) = true\}$ to be the set of all the pairs of vertices that are connected. Then:

$$APL = \frac{\sum_{(u,v) \in CP} PathLength(u, v)}{\mid CP \mid}$$

Setup A java implementation [3] was used to measure the distortion based on the metrics for the five chosen datasets defined earlier in this section. The resulting graphs were

[3] Available at `http://webhome.csc.uvic.ca/~schester/`



(a) Enron - CC vs k (b) Enron Hop Plot

(c) Power Grid - CC vs k (d) Power Grid Hop Plot

(e) Net Science - CC vs k (f) Net Science Hop Plot

Fig. 1. Results - All Datasets

manually verified to be k-anonymous. All experiments were performed on a quad-core Intel Xeon 5140 $2.33GHz$ processor with $4MB$ of L2 cache and $6GB$ of RAM.

4.2 Results and Discussion

Figure 1 show how the metrics of the resulting graphs over varying k compare to those of the original graphs for the five datasets in the experimental study. We vary k from $k = 0.25$ up to 2.00% of n for our experiments, maintaining that $k \ll d$. For Enron, 2% of the number of nodes still translates to k values of 720, which is substantial for the context, providing a reidentification probability of 0.14%.

Distortion Performance From the plots in Figure 1, it can be observed that the values of clustering coefficient [Figures 1 (a),(c),(e)] in the distorted graphs are *close* to the corresponding values before distortion. This holds for APL, too, but we omit the figures for space. For the largest dataset (Enron), even for an extreme value of k (at 2%), these values are very close to the ones before distortion. The Net Science dataset nicely depicts pathological behaviour, still quite good, wherein the densest nodes in the network are not connected to each other. By necessity, an optimal partitioning of the graph into groups each containing at least k vertices will group together these dense and disconnected vertices and create much shorter paths through the graph.

By observing the hop plots [Figures 1 (b),(d),(f)], it can be seen that the shape is very similar for all values of k. Of course, due to the addition of new nodes, the diameter and the maximum y-axis value necessarily increase.

Running Time For the largest dataset (Enron), the running time over five trials was at worst $70s$ to anonymize the graph. The naive computation of the metrics took 20 to 30 minutes each. The times on smaller graphs were much lower and had the same trend where the computation of the metrics dominated the running time.

References

1. Chakrabarti, D., Faloutsos, C.: Graph mining: Laws, generators, and algorithms. ACM Computing Surveys 38(1), 2 (2006)
2. Cheng, J., Fu, A.W.C., Liu, J.: K-isomorphism: privacy preserving network publication against structural attacks. In: SIGMOD 2010. pp. 459–470 (2010)
3. Dwork, C.: Differential privacy. In: ICALP (2). pp. 1–12 (2006)
4. Dwork, C.: Differential privacy: A survey of results. In: TAMC. pp. 1–19 (2008)
5. Dwork, C.: Differential privacy in new settings. In: SODA. pp. 174–183 (2010)
6. Erdős, P., Kelly, P.: The minimal regular graph containing a given graph. In: American Mathematics Monthly v70. pp. 1074–1075 (1967)
7. Hay, M., Miklau, G., Jensen, D., Towsley, D.F., Weis, P.: Resisting structural re-identification in anonymized social networks. PVLDB 1(1), 102–114 (2008)
8. Kapron, B., et al.: Social network anonymization via edge addition. In: ASONAM (2011)
9. König, D.: Akademische verlagsgesellschaft. In: Leipzig (1936)
10. Liu, K., et al.: Towards identity anonymization on graphs. In: SIGMOD. pp. 93–106 (2008)
11. McSherry, F., Mironov, I.: Differentially private recommender systems: Building privacy into the netflix prize contenders. In: KDD. pp. 627–636 (2009)
12. Meyerson, A., et al.: General k-anonymization is hard. In: Principles of Database Systems (2004)
13. Milgram, S.: The small world problem. In: Psychology Today. vol. 2, pp. 60–67 (1967)
14. Thompson, B., Yao, D.: The union-split algorithm and cluster-based anonymization of social networks. In: ASIACCS 2009. pp. 218–227 (2009)
15. Tripathy, B.K., Panda, G.K.: A new approach to manage security against neighborhood attacks in social networks. In: ASONAM. pp. 264–269 (2010)
16. Wu, W., Xiao, Y., Wang, W., He, Z., Wang, Z.: k-symmetry model for identity anonymization in social networks. In: EDBT 2010. pp. 111–122 (2010)
17. Zheleva, E., Getoor, L.: Preserving the privacy of sensitive relationships in graph data. In: KDD 2007. pp. 153–171 (2007)
18. Zhou, B., Pei, J.: Preserving privacy in social networks against neighborhood attacks. In: ICDE 2008. pp. 506–515 (2008)