On the Hardness of Graph Anonymization

Charu C. Aggarwal
IBM T. J. Watson Research Center
Yorktown Heights, NY, USA
charu@us.ibm.com

Yao Li, Philip S. Yu
University of Illinois at Chicago
Chicago, IL, USA
{*psyu, yli70* }*@uic.edu*

***Abstract*—In this paper, we examine the problem of node re-identification from anonymized graphs. Typical graphs encountered in real applications are *massive* and *sparse*. In this paper, we will show that massive and sparse graphs have certain theoretical properties which make them susceptible to re-identification attacks. We design a systematic way to exploit these theoretical properties in order to construct *re-identification signatures*, which are also known as *characteristic vectors*. These signatures have the property that they are extremely robust to perturbations, especially for massive and sparse graphs. Our results show that even low levels of anonymization require perturbation levels which are significant enough to result in a massive loss of utility. Our experimental results also show that the true anonymization level of graphs is much lower than is implied by measures such as k-anonymity. Thus, the results of this paper establish that the problem of massive graph anonymization has fundamental theoretical barriers which prevent a fully effective solution.**

I. Introduction

In many massive graph and social network applications, it may sometimes be desirable to release a *node de-identified network* for data mining and analysis [2]. This may however not be sufficient protection, if the identities on the nodes can be attacked with the use of background information about network structure. It was observed in [6], that pure de-identification is not sufficient for effective privacy-preservation. A lot of research has recently been devoted to the problem of graph anonymization [3], [7], [11], [9], [10]. These methods use techniques such as duplicating, removing, switching or grouping vertices or edges in input graphs. Some of the techniques [5] rely on completely random edge additions and deletions, whereas others [7] use more carefully designed anonymization methods.

While a number of privacy-attack methods have been proposed for network re-identification [4] or re-construction [8], these methods are *ad-hoc*, and depend upon specific structural characteristics of the graphs. These techniques do not provide a *theoretical understanding* of why a particular attack should be successful. In this paper, our goal is to create such a systematic characterization of the linkage structure of the graph. We provide a theoretical understanding of its effectiveness and use it to create a re-identification attack algorithm. The re-identification algorithm uses the aggregate *covariance behavior* of the network linkage structure. *We will see that this attack measure is far more robust to edge perturbations as compared to typical utility measures such as the distances between nodes.* This is essentially because our attack measure is based on *robust statistical quantifications which depend upon the aggregate network structure*, whereas typical utility measures such as distances are highly sensitive to the behavior of a few edges. Thus, utility is degraded much faster than privacy is achieved. Thus, the results of this paper establish that *the linkage behavior of graphs contains robust statistical information which is hard to hide with the use of structural anonymization techniques.* This establishes the hardness of the problem of massive graph anonymization.

This paper is organized as follows. In section 2, we will show that the linkage behavior of massive networks is robust to perturbations. In section 3, we leverage this statistical robustness in order to create a *characteristic vector or signature* for each node, and use it to create a re-identification attack. Section 4 provides the experimental results. Section 5 contains the conclusions and summary.

II. Network Linkage Signatures

The graph G is denoted by (V, E), where V is the set of vertices, and E is the set of edges. For simplicity, we assume that the edges in the graph are directed, though a similar analysis also applies to the case of undirected networks. We assume that the total number of vertices in the network is denoted by N. We will use a careful analysis of the *correlations in the linkage structure* of the nodes in order to construct our attack algorithm. Unlike degree information, this is *second order correlation information* about the linkage structure, which is particularly robust to random edge perturbations in massive and sparse networks. First, we will define the concept of *linkage covariance* between a given pair of nodes. For a given node i, let $\hat{X}^i$ represent the random 0-1 variable, which takes on the value 1, if node i is linked by a directed edge to any particular (potentially adjacent) node and 0 otherwise. Thus, we have instantiations of this random variable for all possible (potentially) adjacent nodes j, and the corresponding instantiation is denoted by x_{ij}. The value of x_{ij} is 1, if a directed edge does indeed exist from node i to node j. We denote the *Linkage Covariance*

between nodes p and q by $LinkCov(p, q)$ and define it as follows:

Definition 2.1: The linkage covariance between nodes p and q is denoted by $LinkCov(p, q)$ and is equal to the covariance between the random variables $\hat{X^p}$ and $\hat{X^q}$. Therefore, we have:

$$LinkCov(p, q) = E[\hat{X^p} \cdot \hat{X^q}] - E[\hat{X^p}] \cdot E[\hat{X^q}]$$
$$= \sum_{k=1}^{N} x_{pk} \cdot x_{qk}/N - (\sum_{k=1}^{N} x_{pk}/N) \cdot (\sum_{k=1}^{N} x_{qk}/N)$$

The last equation expands the formula for covariance by using the data instantiations associated with the edges in the underlying network.

Next, we examine the effect of the addition of edges to the link covariance. Let us assume that the node p has n_p outgoing edges, and the node q has n_q outgoing edges. We denote the number of edges from p and q which point to the same node by m_{pq}. Let us also assume that edges are added to the graph with probability f_a. We denote the sparsity level (i.e. the fraction of edges of a completely connected network) of the network by f_s. We note that f_a is typically a small quantity, because most real networks are extremely sparse, and the number of edges which are added are also likely to be smaller than the number of edges in the original network. The latter is usually necessary in order to preserve the utility of the underlying network. Then, we can show the following result for large sparse graphs:

Lemma 2.1: Let L' be the estimated value of the link covariance between nodes p and q after the addition of edges with probability f_a. Then, the expected value of the estimated link covariance L' is related to the true link covariance $LinkCov(p, q)$ by the following relationship:

$$E[L'] = LinkCov(p, q) - 2 \cdot m_{pq} \cdot f_a/N$$

Proof: Omitted. See [1]. ■

We note that the value of f_a is a fraction, which is less than one. Since the link covariance between a pair of nodes changes by only $2 \cdot f_a \cdot m_{pq}/N$, this suggests that the change in link covariance is dependent on the value of m_{pq}/N. The value of m_{pq} is the number of nodes to which both nodes p and q have a connecting edge. *This value is smaller than the degree of both the nodes p and q, and is especially very small for sparse graphs. For very massive and sparse graphs in which the value of N is much larger than the value of m_{pq}, the value of m_{pq}/N becomes extremely small.* This also means that the overall change in the link covariance is extremely small for massive and sparse graphs. This suggests that it is possible to systematically leverage the linkage covariance in the large graph in order to create a node re-identification mechanism which is robust to edge perturbations.

The robustness of the link-covariance measure is because of its aggregate nature. We can further increase the robustness of this measure by using the pairwise information about the link covariances in a more coordinated way. Specifically, for a given node, we can define a *vector* of link covariances to the other nodes in the graph in both the de-identified network and the original network. The key is to define a *mapping* of nodes from one network to the other, such that the ordered vectors created by this mapping are very similar to one another. Since each link covariance value is only slightly affected by the randomization process, it follows that the vector of link covariances is also not affected significantly. We will use this vector in order to create a *characteristic vector* for each node. This characteristic vector is a representation of the relationship of the linkage behavior of a node with all other other nodes in the graph. The small variation in the link covariance will ensure that the characteristic vector is quite robust to the anonymization process. Before discussing the characteristic vector in detail, we will first show that the result discussed above is also true for edge-deletion, though the proof technique is slightly different. This is because anonymization techniques use both edge additions *and* deletions for the transformation process.

Lemma 2.2: Let L' be the estimated value of the link covariance between nodes p and q after the deletion of edges with probability f_d. Then, the expected value of the estimated link covariance L' is related to the true link covariance $LinkCov(p, q)$ as follows:

$$E[L'] = LinkCov(p, q) \cdot (1 - 2 \cdot f_d)$$

Proof: Omitted. See [1]. ■

We note that the deletion probability f_d has a different interpretation than the addition probability f_a, but the value of f_d also needs to be small enough to retain the basic structure of the overall graph. This is because for large values of f_d, the basic structure of the underlying graph may be lost. Furthermore, we note that the final expected value of the link covariance after edge deletion continues to the proportional to the initial value.

Since the link covariances for a given node do not change very easily, they can be used to define a *signature or characteristic vector* for that node. The characteristic vectors for the nodes in the background knowledge graph and the de-identified graph can be matched with one another in order to create a re-identification attack. There are several ways of defining this signature or characteristic vector:

- When the mapping between the two graphs are completely unknown, we can create a vector of link covariances, which are sorted in decreasing values. This creates a monotonically decreasing vector.
- When the mapping between the two graphs are approximately known for **all** nodes, we can create a sort order of nodes in the two graphs corresponding to this

mapping. The vector is defined with respect to this sort order. This provides more accurate results, when we match the signatures between the two graphs.

- In some cases, an approximate mapping is known for some of the nodes, but not others. In such cases, we use a sort order on the nodes for which the mapping is known, and use a sort order on the magnitudes of the link covariances in order to define the remaining part of the signature.

All of the above definitions are quite useful, because we often start off with no mapping between the nodes, and then partially define the mappings. Correspondingly, we will define different kinds of vectors which will be required during different phases of the matching process. We will use an iterative matching approach, in which the first iteration uses one kind of the characteristic vector with no information about the mapping, and later iterations use other forms of the characteristic vector with partial information about the approximate matching.

When we do not even have any approximate information about the matching between the two graphs, then the most natural form of the characteristic vector of node p is defined as the *ranked link characteristic vector*. This is defined as the characteristic vector obtained by ordering the nodes in decreasing order of the link covariance with respect to node p. This is defined with respect to an *unordered node set* S, and the ranking is imposed based on the behavior of the link-covariance of p to other nodes. This kind of characteristic vector is useful for cases, where we do not have an approximate matching between the pair of graphs. The ability to use an unordered set allows us to create a signature for any node in each of the graphs and match the corresponding signatures.

Definition 2.2: The ranked link characteristic vector $\overline{RC(p, S)}$ for a node p and **unordered** node set $S = \{i_1 \ldots i_s\}$ is defined as the s-component vector $(LinkCov(p, j_1) \ldots LinkCov(p, j_s))$, where the ordering $j_1 \ldots j_s$ is defined in order to ensure that $LinkCov(p, j_i) \geq LinkCov(p, j_{i+1})$. Ties are broken using the lexicographic ordering of the node labels.

In other words, the node set S is *a-priori unordered*, but we impose the rank ordering of the nodes in S in decreasing link-covariance with respect to node p. We use this to construct a smooth monotonically decreasing curve representing the variation in this link covariance. It is evident that the shape of this curve could vary quite significantly with the underlying node. This shape is the *characteristic* of that node, and is used for the matching process. We denote the ranked link covariance by $\overline{RC(p, S)}$. The vector $RC(\cdot)$ can be computed without any knowledge about the mapping, and this is useful in the initial stages of the re-identification process, when no information about the matching of the nodes is known. Therefore, a natural strategy at the initial stages of re-identification is to use an ordering which is specific to each node, and is based on a ranking of the magnitude of the link-covariance. Next, we will define the concept of a characteristic link vector for a given node, with respect to a *known* ordering. The characteristic vector is defined with respect to a node p, and an *ordering* S_o of the node set S. Therefore, S_o is considered a *sequence* of nodes, representing an ordering of the nodes in S. In other words, we have $S_o = \{i_1 \ldots i_s\}$. The characteristic link vector for a given node p and ordered vector S_o is denoted by $\overline{C(p, S_o)}$.

Definition 2.3: The characteristic vector $\overline{C(p, S_o)}$ for a node p and **ordered** set $S_o = \{i_1 \ldots i_s\}$ is defined as the s-component vector $(LinkCov(p, i_1) \ldots LinkCov(p, i_s))$.

The above vector is useful in situations in which a mapping between **all** the nodes of the two graphs is approximately known. We note that the concepts of $\overline{C(p, S_o)}$ and $\overline{RC(p, S)}$ are two extreme ways of creating the characteristic vectors corresponding to whether **no** mapping is known, or (an approximate version of) the **complete** mapping is known. In cases in which the mapping of only a subset S of the nodes is known, it is also possible to create a *partially ranked characteristic vector* $\overline{PRC(p, S_o, V)}$ with respect to the ordered set S_o and the unordered set $V - S_o$, where V is the vertex set of the entire graph. This is essentially a combination of the two cases above in which we use the known portions of the ordering in the subset S_o, and use a sort-order on the link-covariance for the other nodes.

Definition 2.4: Let V denote the vertex set of the entire graph, and S_o be an ordered subset of V. Then we define the *partially ranked characteristic vector* $\overline{PRC(p, S_o, V)}$ by concatenating $\overline{C(p, S_o)}$ and $\overline{RC(p, V - S_o)}$. In other words, we have $PRC(p, S_o, V) = (\overline{C(p, S_o)}, \overline{RC(p, V - S_o)})$.

We will see that the above definition is particularly useful, when the mapping information about a subset of the nodes are known. All of the vectors above can be normalized by dividing by their L_2 modulus. We call such vectors as the *unit* characteristic vectors. Throughout the next section, we will utilize the unit characteristic vectors in order to perform re-identification attacks.

III. Re-Identification Attack

For the base graph G, in which the identities are available, we assume that background information is available in terms of the the links of these nodes. While the entire network is not available, it is reasonable to assume that in many graphs and social networks, small localities around specific nodes are known. This is of course dependent upon the network which is being attacked. For example, a social network such as *Twitter* has an open API, and the entire linkage structure of the network can be known. Therefore, the release of any sensitive information along with the de-anonymized graph can be highly revealing, because a lot of network structure is available in order to determine the node identities. On the other hand, in social networks such as *Facebook*, it is possible to query most nodes of the graph in order to know

the corresponding linkages. However, the site has built-in protections to prevent the automated crawl of the whole network. Therefore, it is reasonable to assume that only a subset of the network can be manually explored.

We assume that for a given set S of nodes, its links, and the links of its neighbors are known. Thus, all nodes up to a distance of 2 from specifically targeted nodes are known. This level of local exploration is often possible in many massive graphs and social networks. The goal of the re-identification algorithm is to determine a node set S' from the *de-identified graph* so that a one-to-one correspondence can be created from nodes in S to nodes in S'. This leads to disclosure of the identities of the nodes in S', and therefore the sensitive information associated with nodes or edges can be breached. This one-to-one matching can be defined by picking an ordering of nodes in S', which matches the corresponding ordered nodes in S. We define the *goodness of the re-identification* as the *link similarity* between S and S'. The link similarity between S and S' is defined as the dot product between the corresponding characteristic vectors:

Definition 3.1: The link similarity between the ordered set of (labeled) nodes S from the publicly available graph, and the set of nodes S' in the de-identified graph is defined as the dot products between their *normalized partially ranked characteristic vectors*.

We note that this dot product always lies in the range $(0, 1)$ because of the normalization process. The higher the dot product between the characteristic vectors, the greater the link similarity between the corresponding pair of nodes. We make the following observation:

Observation 3.1: If we have a pair of identical graphs G and G' (without edge randomization), and we correctly match a set of nodes S in G to a set of nodes S', then the dot product between the corresponding characteristic vectors is 1.

We define the matching problem from the publicly available portions of the graph to the anonymized graph as follows:

Problem 3.1: Determine an ordered set S' in the de-identified and perturbed graph G', so that the dot product of its characteristic vector with the characteristic vector of the ordered (identified) set S from the base (publicly available) graph G is maximized.

The above problem is NP-hard even in the case when the edges are not perturbed. Therefore, we will design an effective heuristic for the problem. In order to design the re-identification algorithm, we will make a number of approximation assumptions based on the behavior of the algorithm. First, we note that most massive graphs are *extremely sparse* and a very small fraction of the nodes have large degrees. For example, a typical social network such as *Facebook* may have millions of nodes, but most nodes have links to less than a hundred nodes. In general, we make the assumption that the average degree of a node is significantly less than $\sqrt{N}$, where N is the total number of nodes in the social network. This makes it easy to perform the following approximation for the link co-variance. As before, we assume that $\hat{X}^i$ represents the random 0-1 variable, which characterizes the linkage of node i to node j. The instantiations of this random variable to the different adjacent nodes j are denoted by x_{ij}.

Approximation 3.1: The linkage covariance between nodes p and q is denoted by $LinkCov(p, q)$ can be approximated as the expectation of the product of random variables variables $\hat{X}^p$ and $\hat{X}^q$. Therefore, we have:

$$LinkCov(p, q) \approx E[\hat{X}^p \cdot \hat{X}^q] \quad (1)$$

The important point to show here is that the characteristic vector encodes the *relative behavior* of the link co-variances, and this relative behavior is unaffected by this approximation. In order to elaborate further, we express the link covariance as follows:

$$LinkCov(p, q) = E[\hat{X}^p \cdot \hat{X}^q] - E[\hat{X}^p] \cdot E[\hat{X}^q] \quad (2)$$

In order to show that the relative behavior of the link-covariances are unaffected by this approximation, we have to show that non-zero values of the second term are significantly smaller in magnitude than non-zero values of the first term. Note that the *least* non-zero value of the first term is $1/N$, which corresponds to the case that p and q share exactly one link. The "typical" value of the second term is approximately d^2/N^2, where d is the average degree of the nodes. Since $d << \sqrt{N}$, it follows that $d^2/N^2 << 1/N$. Therefore, the shape of the characteristic vector is likely to be well estimated with the use of this first-order approximation.

A second property of this approximation is that *it allows the computation of the characteristic vector for a node with the use of a small amount of local information around that node*. This holds true for all forms of the characteristic vector (ranked with respect to a full or partial ordering of nodes). Specifically, we make the following claim:

Observation 3.2: Any of the characteristic vectors for a node p can be approximated with the use of only the link information of node p and its immediate neighbors.

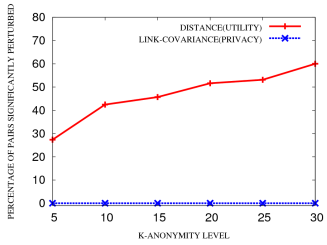
This is because any other node q which has a non-zero value of $E[\hat{X}_p \cdot \hat{X}_q]$ must link to at least one node which is a neighbor of node p. Therefore, by examining all the inlinks of the neighbors of p, and determining the number m_{pq} of neighbors of p which contain an inlink from node q, it is possible to estimate the value of $E[\hat{X}_p \cdot \hat{X}_q]$ as m_{pq}/N. The link covariance of all nodes which do not share a common neighbor with node p are set to 0.

Thus, if the local link structure of a subset of nodes S is known, then it can be used in order to construct the (ordered, partially ordered or unordered) characteristic vector. Of course, in order to construct the re-identification, we need to determine two things:

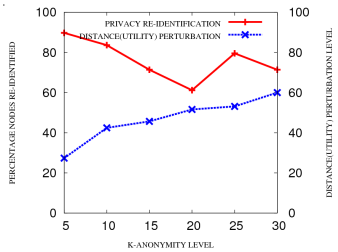
- For the node set $S \subseteq V$ in G, we need to determine the corresponding set $S' \subseteq V'$ in G'.
- For the sets S and S', we need to construct a one-to-one matching of the nodes, so that the characteristic vectors can be created with the corresponding partial orderings.

We will transform the problem to a bipartite graph matching problem. Let $G = (V, E)$ and $G' = (V', E')$ be the original and de-identified graphs respectively. It is assumed that we have local information about a subset S of nodes of the original graph G. We create a bipartite graph $H = (V \cup V', D)$, where one partition of the bipartite graph contains nodes corresponding to V of G, whereas the other partition contains nodes corresponding to V' of G'. The edge set D contains an edge from each node of $S \subseteq V$ to every node of V'. The weight of an edge between $i \in V$ and $j \in V'$ is (initially) defined as the dot product of the normalized characteristic vectors for nodes i and j with respect to the entire vertex sets in the two graphs. In other words, we use the dot product of $RC(i, V)$ and $RC(j, V')$. In the event that the graphs do not have the same number of vertices, we append zeros at the end of the smaller of the two characteristic vectors, so that a dot product can be performed. For a given set $S \subseteq V$, a *bipartite matching* is defined as a set of node-disjoint edges, such that there is one edge from a node in S to a node in V'. These defines the one-to-one mapping between the nodes of the two graphs. Clearly, the determination of the maximum weight one-to-one matching from the subset of nodes S to the nodes in V' determines a matching of the nodes which maximizes the similarity between the corresponding nodes.

The first matching provides a correspondence of nodes in S and S'. This can be used to further refine the weights of the edges. The first iteration uses a characteristic vector $RC(\cdot, \cdot)$ which is ranked by link covariance size, since we have no a-priori information about the mapping. We can then use the matching generated from this first phase in order to further improve the quality of the assignment by using information about the approximate mapping which is generated. We achieve this by using partially ranked characteristic vectors for the set S and S' which have already been matched. Then, we create partially ranked characteristic vectors with this ordering on S and S' using the first matching and re-define the weights as the dot-products on the (normalized) partially ranked characteristic vectors. As in the previous case, we handle inequality in the sizes of the two graphs by adding zeros to the ranked part of the characteristic vector. We solve the matching problem again with re-defined weights, in order to redefine the matching between S and S'. We repeat the iterative process of re-defining weights from matchings, and vice-versa. In practice, it was found that convergence was achieved in a small number of iterations.



(a) Utility-Privacy:Covariance vs. Anonymity



(b) Utility-Privacy:Re-Identification vs. Anonymity

Figure 1. Privacy and Utility Tradeoffs (Power Grid Graph)

IV. Experimental Results

The effectiveness of a privacy-preservation technique depends upon it ability to retain utility after the privacy-transformation process. We used the following privacy and utility measures:

(1) Distance perturbation (Utility Measure): We sample k pairs of nodes, and determine the distances between these pairs both in the original graph and the perturbed graph. We compute the standard deviation of these k different distance values. We compute the number of node pairs for which the change in distances between node pairs from the original to the perturbed graph is greater than one standard-deviation of the original distances between the node pairs. We note that lower values of the distance perturbation are more desirable because it implies a better preservation of utility.

(2) Link Covariance Perturbation (Privacy Measure): We sample the *same* pairs of nodes as above and calculate the link-covariances between them. As in the previous case, we compute the number of node pairs for which this change is greater than one standard deviation. Thus, in this case, *lower values of the link-covariance perturbation imply that privacy can be attacked, because it means that the link covariance based attack signatures are robust to the underlying anonymization process.*

(3) Node Re-identification Rate (Privacy Measure): An explicit measure of privacy is the *node re-identification rate*. This is the percentage of nodes which are accurately identified with the use of the matching algorithm. This reflects the *true effectiveness* of a k-anonymization algorithm, for which the re-identification rate should be no larger than $1/k$, but turns out to be significantly greater in reality.

A. Results

In this paper, we present results on the Power Grid graph[1], which consists of 4940 vertices and 6594 edges in a power-grid network. Each vertex stands for a generator/transformer and substation and the edge between pairs of them represent the power line in the network. This data set was also used in [5]. In addition results on experiments on a *co-author graph* and *protein* graph are available in [1]. We assumed background knowledge about the linkage structure of only 1% of the nodes.

We tested two different algorithms for its resistance to privacy attacks: a random edge addition-deletion algorithm [5], and a degree-based k-anonymization algorithm with the use of the greedy swap method [7]. The results for the random edge algorithm are available in [1], whereas we present only the results for the k-anonymization method here. The results are presented in Figure 1(a). The X-axis illustrates the value of k for the k-anonymity level, whereas the Y-axis illustrates the distance-based utility as well as the co-variance based privacy measure. The anonymity level was varied between 5 and 30. The percentage of distance pairs significantly perturbed increase monotonically with increasing anonymity. In the case of the *Power Grid* data set illustrated in Figure 1(a), about 60% of the distances between node pairs were perturbed by more than a standard deviation at an anonymity level of 30. Thus, *distances between a majority of the node pairs* are affected *significantly* by the perturbation process. This implies that distance-based utility may be seriously compromised at this level of anonymity. On the other hand, the link-covariance measures for privacy are not affected too significantly, because of their aggregative robustness.

It is also interesting to explore *more direct measures* of the privacy, which test the percentage of nodes which can be re-identified with the use of an attack algorithm. As in the previous case, we also plot the distance-based utility measures (replicated from Figure 1(a)) along with the re-identification level. One issue with showing utility and privacy on the same plot is that the two quantities correspond to different units of reference. The privacy corresponds to the percentage of nodes re-identified, whereas the utility corresponds to the percentage of node *pairs* for which the distance changes by more than a standard deviation. Thus, we have *used two different Y-axes*, a left Y-axis, which corresponds to the re-identification rate, and a right Y-axis which corresponds to the utility. The results are illustrated in Figure 1(b). One of the key observations was that the re-identification rate was *much higher*, than is implied by the expected re-identification rate of $1/k$ as is implied by k-anonymity techniques. For example, for an anonymity level of 30, the expected re-identification rate should be $1/30 = 3.33\%$. However, as is evident from the re-identification chart (left Y-axis) of Figures 1(b), the actual re-identification rates for the *Power Grid* data set at an anonymity level of 30 was 71%. Thus, *the true re-identification rates could be as much or more than an order of magnitude greater than is implied by the k-anonymity model*. Furthermore, even at such high re-identification rates, a significant percentage of the nodes exhibited changes in distance values. At this anonymization level, the percentage of node pairs which showed significant changes in distance values for the *Power Grid* graph was 60%. This essentially implies that a significant compromise of utility is still not sufficient to retain the privacy implied by the k-anonymization approach.

[1]Available at http://www.cs.helsinki.fi/u/tsaparas/MACN2006

V. Conclusions

In this paper, we examined the problem of massive graph anonymization and proposed a technique for mounting re-identification attacks on the nodes. We presented theoretical results which suggest that such attacks are likely to be particularly robust to perturbation in the case of massive graphs. We also present experimental results which confirm that such characterizations of the graph linkage structure are much more robust to perturbation than distance-based utility measures. Thus, the results of this paper establish the fundamental hardness of the problem of graph anonymization.

References

[1] C. C. Aggarwal, Y. Li, P. Yu. On the hardness of graph anonymization. *IBM Research Report*, 2012.

[2] C. C. Aggarwal, P. S. Yu. *Privacy-Preserving Data Mining: Models and Algorithms*, Springer, 2008.

[3] G. Cormode, D. Srivastava, T. Yu, Q. Zhang. Anonymizing Bipartite Graph Data using Safe Groupings. *VLDB*, 2008.

[4] M. Hay, G. Miklau, D. Jensen, D. Towsley, P. Weis. Resisting Structural Re-identification in Social Networks, *VLDB*, 2008.

[5] M. Hay, G. Miklau, D. Jensen, P. Weis, S. Srivastava. Anonymizing social networks. *Technical Report 07-19, University of Massachusetts*, Amherst, 2007.

[6] L. Backstrom, C. Dwork, J. Kleinberg. Wherefore Art Thou R3579X? Anonymized Social Networks, Hidden Patterns, and Structural Steganography. *WWW Conference*, 2007.

[7] K. Liu, E. Terzi. Towards identity anonymization on graphs. *ACM SIGMOD Conference*, 2008.

[8] L. Wu, X. Ying X. Wu. Reconstruction from Randomized Graph via Low Rank Approximation, *SDM Conf.*, 2010.

[9] X. Ying, X. Wu. Randomizing Social Networks: a Spectrum Preserving Approach. *SDM Conf.*, 2008.

[10] X. Ying, K. Pan, X. Wu, L. Guo. Comparisons of randomization and k-degree anonymization schemes for privacy-preserving social network publishing. *KDD Conf.*, 2009.

[11] B. Zhou, J. Pei. Preserving Privacy in Social Networks Against Neighborhood Attacks. *ICDE Conference*, 2008.