Lost in the Crowd: Are Large Social Graphs Inherently Indistinguishable?

by

Subramanian Viswanathan Vadamalai

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Adriana Iamnitchi, Ph.D.
John Skvoretz, Ph.D.
Paul Rosen, Ph.D.

Date of Approval:
May 10, 2017

Keywords: Graph Anonymization, Privacy Metric, Linkage Covariance

DEDICATION

I would like to dedicate this MS to my family and friends, whose invaluable presence and support helped me through all my travails.

ACKNOWLEDGEMENT

I would like to thank my advisors Dr. Adriana Iamnitchi and Dr. John Skvoretz for supporting me in this endeavor. They provided me with the golden opportunity to understand about social graphs, its dynamics, graph processing, statistics and high performance computing. Their support was monumental in my perseverance to finish my Master's project on inherent indistinguishability of nodes in large social graphs.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Real social graphs datasets are fundamental to understanding a variety of phenomena, such as epidemics, crowd management and political uprisings, yet releasing digital recordings of such datasets exposes the participants to privacy violations. A safer approach to making real social network topologies available is to anonymize them by modifying the graph structure enough as to decouple the node identity from its social ties, yet preserving the graph characteristics in aggregate. At scale, this approach comes with a significant challenge in computational complexity.

This thesis questions the need to structurally anonymize very large graphs. Intuitively, the larger the graph, the easier for an individual to be "lost in the crowd". On the other hand, at scale new topological structures may emerge, and those can expose individual nodes in ways that smaller structures do not.

To answer this problem, this work introduces a set of metrics for measuring the indistinguishability of nodes in large-scale social networks independent of attack models and shows how different graphs have different levels of inherent indistinguishability of nodes. Moreover, we show that when varying the size of a graph, the inherent node indistinguishability decreases with the size of the graph. In other words, the larger a graph of a graph structure, the higher the indistinguishability of its nodes.

CHAPTER 1: INTRODUCTION

Real social network datasets are a fundamental means of research for understanding a variety of phenomena, such as epidemics, crowd management and political uprisings. The sudden increase in the number of online social networks has spurred the interest of computer scientists. In the online social networks that are open for sharing user data, users are not willing to share personal data fearing misuse by unintended parties. In general, data is shared by online social networks to advertising agencies for business purposes, and to scientists for research purposes. Online social networks have strict privacy controls when sharing data and restrict personal information in the data they release for analysis. Information like names and addresses are removed before sharing. This is not sufficient for protecting private data if node identities can be attacked by using background information about network structure. There were several attacks mounted on the releases of such anonymized data that exposed sensitive user information by making use of structural properties, such as neighborhood information of popular nodes.

One such example was of an attack on the data released by Netflix in 2006. The published dataset contained over 100 million movie ratings by over 480 thousand Netflix subscribers between December 1999 and December 2005. Despite removing all personal information from the dataset (such as user name, real name, email addresses, etc.), a team of attackers successfully de-anonymized the dataset and identified 99% of the users correctly [1]

Another example was the case of "Jefferson High" school incident. It is part of the National Longitudinal Study of Adolescent Health, which contains very detailed health information on 100,000 high school students in 140 schools [2]. However, attackers were able to map out the entire sexual network in 12 of the schools [33] by making use of information presented by the authors on the certain nodes and their neighbors in the dataset.

Given these proofs of serious privacy risks due to releasing datasets even without personally identifiable information, much effort has been invested in structurally anonymizing social graphs. Intuitively, the objective is to change the topology of the graph such that individual nodes cannot be recognized based on local information that might be available to the attacker, such as node degree, clustering coefficient or other neighborhood information. At the same time, for the released, anonymized dataset to be useful, the utility must be preserved. Utility is typically defined based on a subset of graph properties of the original dataset that need to be preserved in the anonymized dataset [3, 4].

One significant challenge in structural graph anonymization is the tension between anonymity and utility [10]. Intuitively, the more the structure of the graph is modified, the better the anonymity of the nodes and edges in graph, but the more its topological properties change, the lower the resulting utility. For very large graphs, as today's available datasets are, preserving utility while ensuring graph anonymity is computationally challenging. The algorithmic time complexity in such cases can be in the order of $O(|E_t|)$ and $O(n^2)$ [5, 6].

1.1 Research Objective

A basic question that has not been asked, to the best of our knowledge, is how inherently private large graphs are. Since one measure of anonymity is how much effort it takes to break a secret, computations on large graphs are inherently expensive: the larger the graph, the more

computational effort is needed to de-anonymize the anonymized version of the graph. We thus ask: is it true that large real graphs are more private than small real graphs? If this were true, graph anonymization efforts could be tailored for the size of the graph: smaller graphs would need more sophisticated anonymization procedures, larger graphs would require less significant anonymization effort to provide the same guarantees.

To answer this question, we need a way of defining the inherent indistinguishability of nodes in a graph. In this work, we focus on the indistinguishability of nodes and ignore the issue of edge anonymity. (We note that there is work that considers edge anonymity as the important objective for privacy protection [7]). We refer to the inherent indistinguishability of nodes in a graph as the property of the graph to conceal identifiable topological information for a large set of its nodes. In this respect, a clique, a ring, a star, and a lattice graph are examples of inherently private graphs: based on topological properties, all nodes are identical, thus "lost in the crowd". This intuition is at the core of k-anonymity anonymization techniques [8, 9].

This thesis defines the inherent indistinguishability of nodes in a graph based on the notion of linkage covariance introduced in Chapter 3. Aggarwal et al. [10] introduced linkage covariance as a graph invariant and show that it varies insignificantly in the process of anonymizing graphs by swapping edges. Consequently, they showed that simple edge-swapping anonymization techniques are insufficient for providing anonymity, as the signature of the nodes remains almost intact.

We define the inherent indistinguishability of nodes in a graph as a function of the percentage of nodes in the graph that have unique linkage covariance signatures [10]. Alternatively, we can reason about the inherent indistinguishability of nodes in a graph by

analyzing the percentage of nodes that have at least k other nodes with identical signatures (like k-anonymity).

1.2 Contributions of this Thesis

Existing approaches in the literature do not provide metrics appropriate for measuring the inherent indistinguishability of nodes in a social graph. Instead, they provide a way to measure the extent of anonymity by measuring node re-identification after designing specific attacks on an anonymized network.

We define the inherent indistinguishability of a graph based on two metrics: risk index and safety index. The risk index measures how many nodes have distinct topological properties that might give them away. The safety index measures how many copies of nodes with identical topological properties each node in the graph has.

We follow the work in [10] and investigate the potential of linkage covariance as a metric of quantifying the inherent indistinguishability of nodes in a graph. Linkage covariance considers the structural properties as well as the quality of links that connect a node with other nodes to measure the structural uniqueness of a node in the context of the entire network. To test the efficacy of such an approach, we use a combination of real-life social graphs and synthetic graphs. Graph generating techniques such as Erdős–Rényi [31] and Forest Fire [34], to name a few, generate truly random graph configurations that allows us to validate the approach across the different graph structures that can exist based on a given size and density.

The contribution of this thesis is three-fold:

1. It proposes metrics based on linkage covariance [10] that measures the indistinguishability of nodes in a network;

2. It provides a scalable implementation of linkage covariance for large social graphs; and
3. It shows that large social graphs are inherently more private than smaller graphs of similar type.

1.3 Overview

This thesis has the following format: Chapter 2 overviews related work. Chapter 3 details linkage covariance, metrics that defines the topological relationship between two nodes in the graph. It also introduces a more scalable algorithm than initially provided in [10] for computing this metric. Chapter 4 describes the implementation details for computing linkage covariance on large graphs. Chapter 5 presents our experimental setup and Chapter 6 presents metrics for measuring the inherent anonymity of real graphs. Finally, Chapter 7 concludes with a summary and future work.

CHAPTER 2: RELATED WORK

When measuring the privacy of social graphs, we need to measure the extent to which the privacy of a node is preserved since privacy is known to be inherently personal, which recognizes the sovereignty of the individual [11]. It is shown that the de-coupling of node identities to anonymize social network data is not sufficient for a safe data release [12]. Nodes could compromise the identity of a neighborhood that they are a part of, providing a key to disclosing other nodes in the neighborhood. Hence, the structural properties of a graph should also be anonymized by making use of structural graph anonymization schemes. Broadly speaking these schemes fall under two major categories: clustering based, and perturbation based [13]. Clustering-based anonymization schemes deal with anonymizing a graph by representing a node as a group of nodes in the anonymized version. Perturbation-based anonymization schemes anonymize the graph by adding or deleting nodes or edges and thereby perturb the structure of the graph.

As a perturbation-based scheme most relevant to this work, k-anonymity [9] is used to induce node indistinguishability in a graph such that an individual node is indistinguishable from at least k-1 others. This technique has been extended to methods like k-degree anonymity [24], k-isomorphism [25], and k-symmetry [26], techniques that perturb a graph such that at least k structurally equivalent sub-graph patterns exist. Interested researchers could explore [13] for a deep-dive into a comparison of different anonymization schemes. In our study, however, we focus on the quantification of inherent indistinguishability of nodes in a graph, which is independent of any anonymization mechanisms.

Many existing quantification techniques measure the de-anonymizability of an anonymized network relative to an attack conditioned by the availability of seeds and other auxiliary information.

Pedarsani and Grossglauser [14] discuss the boundaries of anonymity in terms of fundamental network properties regardless of any specific de-anonymization algorithm. It makes use of graph sampling techniques to control the structural similarity of the anonymized and auxiliary graph structures that minimize the cost function of edge-inconsistencies on graph matching. The study measures the indistinguishability of edge pairs as a measure to introduce simple conditions on delineating the boundary of privacy, and show that the mean node degree needs only grow to slightly faster than log n for a network of size n for nodes to be identifiable. However, this quantification is not suitable in practice since most, if not all, observed real-world graphs do not follow the Erdős–Rényi model, which they consider. The study does not present the computational complexity of such techniques.

Later, Korula and Lattanzi [17] have adapted these quantification methods for the problem of social network reconciliation over Preferential Attachment and Erdős–Rényi models. They assumed to have dense seeds (structure-aware mappings) in the attack model, without relying on any specific domain information.

Ji et al. [15] extend the notion of graph matching to find identical edge-wise partners by introducing a novel cost function based on neighborhood difference, and evaluated against real world networks. The used quantification to quantify to quantify the degree of de-anonymizability that is characterized by the graph projection probability. An interesting observation is that the de-anonymization results in two datasets with similar graph densities may be very different in practice. To study the reason for this fact, they consider the degree distribution of respective

graphs. They found that low degree users, especially first degree users, do not have too much distinguishable structural information, which implies that they are difficult to be de-anonymized based on structural information. Consequently, the existence of a large amount of low degree users in one dataset makes it less de-anonymizable. In general, they conclude that when a dataset has a high average degree and a small percentage of low degree users, it is easier to de-anonymize and a large amount of its users are de-anonymizable; otherwise, datasets with a low average degree and a large percentage of low degree users are difficult to de-anonymize based solely on structural information.

Later, Ji et al. [16] introduced the concept of structural-importance aware relative de-anonymization wherein they argue that the higher the degree of a node, the higher its structural importance in the graph. They quantify relative de-anonymization by the availability of seeds. The seed-based version has a condition on the sampling probabilities and the number of seeds. Empirical results conclude that graphs with higher average degree and graph density are better in de-anonymizability which are vulnerable to structure based de-anonymization attacks. The study also finds that sparse datasets have less structural information to compromise, thus they are inherently more resilient to structure-based attack models, an inference that we also make.

Several works use the Bayesian inference model to measure the efficacy of an anonymization scheme [20, 21, 22], where anonymizability is quantified over the number of incorrect guesses made by the attacker. These measures (e.g., Min Entropy [20], Shannon Entropy [21, 22]) gauge the distortion caused by the anonymization algorithm. This set of statistical measures helps in quantifying differentially private algorithms [13].

Berlingerio et al. [19] define NetSimile similarity score to compare k different networks based on structural features supported by social theories. The paper measures the node

overlap between two different graphs without node attribute information but relies on social theories like Social Capital, Structural Hole, Balance and Social Exchange to measure structural distance. The experiments are conducted using two different techniques – FSM, which is the frequent baseline of smaller subgraphs; and EIG, which is the k largest eigenvalues to evaluate the approach. Their study finds that as real networks are perturbed, the similarity score decreases, but it remains unchanged for Erdős–Rényi networks though they are perturbed.

A general attack is modeled to capture the likelihood of nodes being re-identified using machine learning techniques [27]. It is based on a classifier trained on random decision forest to compare perturbation based graph anonymization schemes. The anonymity of a scheme is measured by the de-anonymization success achieved by the trained model. The results are depicted using the performance of the classifier defined by simple statistical methods like Receiver Operating Characteristic and Area Under the Curve. Moreover, the utility is measured over degree distribution, average degree distribution, joint degree distribution, average degree connectivity, degree centrality and the Eigenvector centrality.

Aggarwal et al. [10] present an attack that uses the aggregate covariance behavior of the network linkage structure. To evaluate the effectiveness of their re-identification algorithm, they sample random pairs of nodes and measure pair-wise distance as a utility measure. They also measure deviation of linkage covariance and the node re-identification rate in the pairs as a measure of privacy. The study finds that the characterizations of the linkage structure of the graph are robust to perturbation than distance-based utility measures.

CHAPTER 3: LINKAGE COVARIANCE

This work uses linkage covariance, defined below, as the topological signature of each node in a network. Based on this signature, we later can measure how common or unique a node's signature is, and thus how safe it remains in the "crowd" of a large network.

3.1 Linkage Covariance Definition

We use the definition of linkage covariance as proposed in [10]. Formally, for two nodes p and q, the definition of linkage covariance *LinkCov (p, q)* is equal to the covariance between the two random variables $\hat{X}^p$ *and* $\hat{X}^q$:

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

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

For a given node p, let $\hat{X}^p$ represent the random 0-1 variable, which takes on the value 1, if node p is linked by an edge to any potentially adjacent node q and 0 otherwise. We have instantiations of this random variable for all possible (potentially) adjacent nodes q, and the corresponding instantiation is denoted by x_{pq}. The value of x_{pq} is 1, if an edge does indeed exist from node p to node q. The advantage of linkage covariance is that it is robust to edge additions and deletions for massive and sparse graphs. The linkage covariance for a given node does not change very easily. Hence, they can be used to define a signature or characteristic vector for that node. There are several ways of defining this signature or characteristic vector.

- When the mapping between two different graphs are completely unknown, we can create a vector of linkage covariances, which are sorted in decreasing values. The most natural form of the characteristic vector of node p is defined as the ranked linkage characteristic vector.
- When the mapping between two different graphs are approximately known for all nodes, we can create a sort order of nodes in the two graphs corresponding to this mapping. The sort order is used to define the signature vector. This provides more accurate results, when we match the signatures between the two graphs.

An approximate mapping is known for some of the nodes, but in most cases, it is not. In such cases, if the mapping is known, we use a sort order on the nodes, and to define the remaining part of the signature, we use a sort order on the magnitudes of the linkage covariances.

3.2 Algorithm

The algorithm for the calculation of the linkage covariance function is illustrated below. Using the linkage covariance signatures, we find out how many identical groups are present for all the nodes, and try to evaluate inherent indistinguishability of nodes.

The linkage covariance function begins with the edge list of a graph G. Using the edge list information, for each vertex V_i, we get the neighborhood of V_i with every other node in the graph G. For all the vertices V in the graph G, we then calculate the linkage covariance of the vertices V_i and V_j as the set difference of the neighbors of vertices V_i and V_j respectively. Since the linkage covariance value is associative, the linkage covariance between vertices V_j and V_i is also the same value. This is shown in the Figure 1 describing Algorithm 1 from lines 5 through 9.

In Line 10, the process of ranking the linkage covariance signatures is done to order the signatures in monotonically decreasing order. This process is repeated for all the vertices in the

graph G. We then take the linkage covariance signature of each vertex and compare it with the linkage covariance signature of every other vertex in the graph. If it is unique, we count the number and categorize it appropriately. For all non-unique linkage covariance signatures, we group them based on the size of the group and the number of groups of each size that is identical. This is shown from lines 11 through 27 of the algorithm. We return the identical groups that are found from the algorithm for evaluating the inherent indistinguishability of nodes in the graph.

```
1  linkageCovariances(G):
2  E ← Edge list(G)
3  ∀ V_i ∈ E : Construct N(V_i) | N(V_i) represents neighborhood of node V_i
4  LC ← ø
5  for Vi in E:
6      VS ← V(E) \ Vi
7      for Vj in VS:
8              LC (Vi, Vj) ← |{N (Vi) ∩ N (Vj)}|
9              LC (Vj, Vi) ← LC (Vi, Vj)
10     LC(Vi) ← Sorted vector of {LC (Vi, )} in descending order
11 UniqueLC ← ø
12 IdenticalGroup ← ø
13 for Vi in E:
14     VS ← V(E) \ Vi
15     for Vj in VS:
16             if LC (Vi, ) = LC (Vj, ) then:
17                     if LC[Vi] ∉ UniqueLC then:
18                             UniqueLC ∪ {LC[Vi]}
19                     end
20             end
21 Ngroup ← ø
22 for LC in UniqueLC:
23     N_LC = Σ_{i=1}^{N} [LC[V_i] = LC]
24     N_group ∪ {N_LC}
25 for n in Ngroup:
26     N_LC = Σ_{i=1}^{N} [LC[V_i] = LC]
27     IdenticalGroup[n] = N_l
28 return IdenticalGroup
```

Figure 1: Algorithm describing the proposed approach for identifying the inherent indistinguishability of nodes in a social graph

3.3 Algorithmic Time Complexity

The algorithm for computing the linkage covariance introduced in Figure 1 has the computational complexity of $O(|V|^2)$, where V is the set of nodes in the graph. Although Aggarwal, Li, and Yu reckon that the negative term in Equation 1 can be ignored in the case of large and sparse graphs (which is the case for social networks), this time complexity remains $O(|V|^2)$.

3.4 Example

We have shown a flowchart below that illustrates how the algorithm proposed in Section 3.2 works. For example, let us consider a 10-node graph and run the algorithm described in Section 3.2. The input graph is shown in Figure 3. For this graph, we have shown what the result in each step of the flowchart in Figure 2 will be.

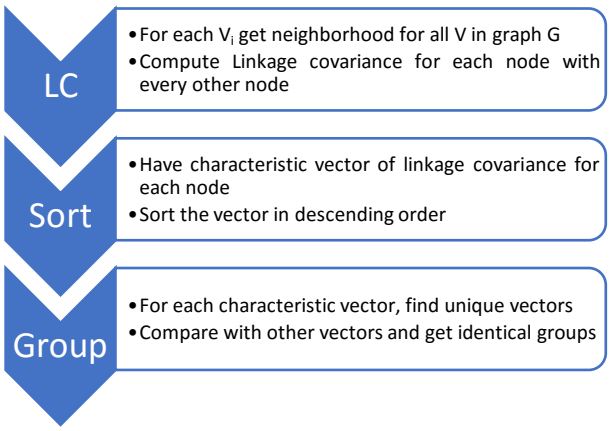

Figure 2: Flowchart of linkageCovariances Algorithm

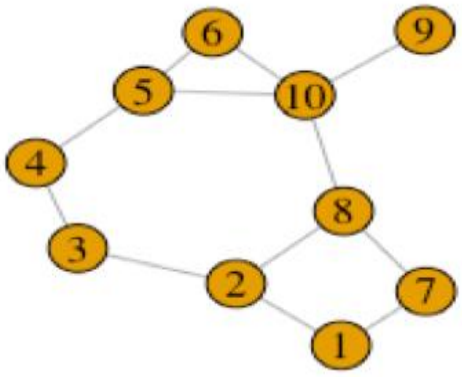

Figure 3: An example 10-node graph

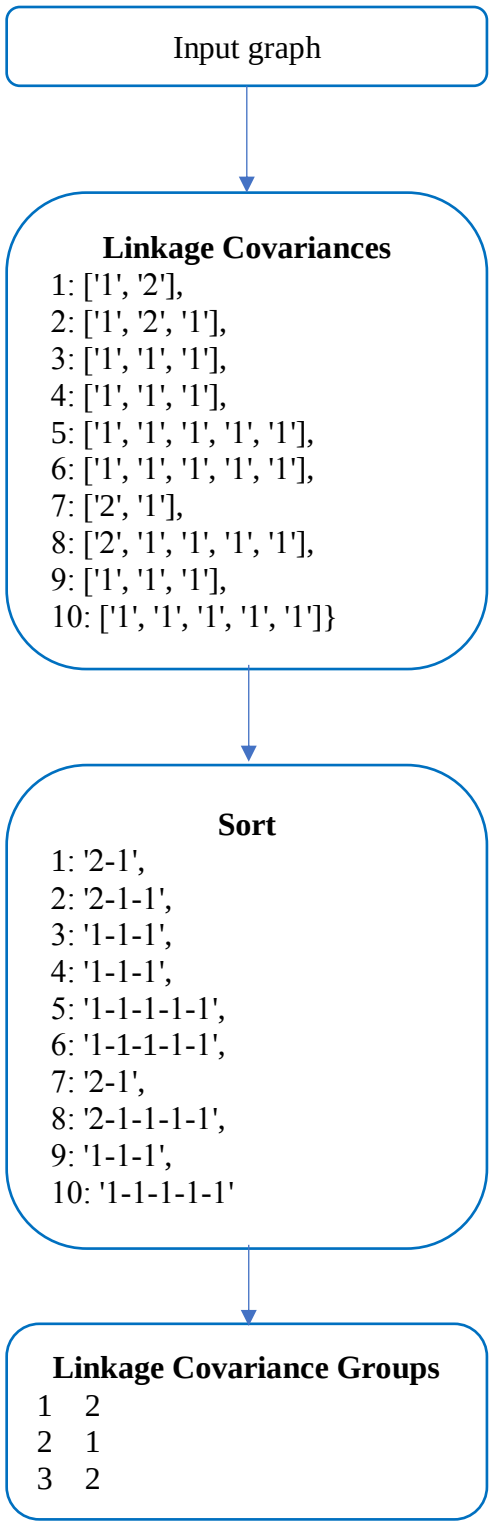

Figure 4: LinkageCovariances Algorithm on the example shown in Figure 3.

CHAPTER 4: IMPLEMENTATION FOR SCALE

Social graphs have millions of nodes, but are sparsely connected. To explore the viability of applying linkage covariance on real-life social graphs, it is necessary to leverage scaling methodologies and parallel processing techniques to alleviate the computational burden required to process graphs of this order.

4.1 Proof of Concept Implementation in R

Initial implementation of linkage covariance as a proof-of-concept model was done in R for ease of prototyping. However, due to R's inherent issues with handling large data and its data structures with high memory costs, scaling the model to process graphs beyond 10,000 nodes was not possible. The data structures available in R allowed processing of graphs in the form of matrices which requires processing memory to be of the order $|V|^2$ for a graph with V set of nodes. When scaling beyond 10,000 nodes, this translates to more than a million data points for computing linkage covariance. Additionally, when considering the computational complexity of the algorithm at $O(|V|^2)$, scaling beyond 10,000 nodes was not viable. For a 5,000-node graph, the program took approximately 6 hours to complete. When the program was tested for a 10,000-node graph, the program caused out-of-memory errors.

To enable the computation of linkage covariance on graphs of more than 10000 nodes, we introduced two techniques. First, we rewrote the code to reduce the memory footprint, which required a different programming language, for which we chose Python. Second, we re-wrote the code using multi-threading. In the following section, we present the techniques for reducing memory footprint and multi-threading.

4.2 Reducing Memory Footprint

Aggarwal and Liu have formulated the value of linkage covariance [10] as mentioned in Chapter 3. In graph parlance, the first term in the equation represents the common neighbors between nodes p and q. The second term translates to the product of the degrees of nodes p and q respectively. Considering large social graphs, Aggarwal and Liu mention in the study [10] that the second term in the Linkage Covariance calculation can be omitted since it will give rise to a very small value. Thus, the equation for calculating linkage covariance reduces to

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

The above equation reduces the linkage covariance formula to just consider the common neighbors between nodes p and q, over the number of nodes N as shown below.

$$LinkCov(p,q) = E[X\grave{a}p \cdot X\grave{a}q] = \sum_{k=1}^{N} x_{pk} \cdot x_{qk}/N \quad (3)$$

This contributes to the reduction in memory footprint. In addition to this, the value of the number of nodes N for large social graphs will be very high. But large social graphs are sparse, so the common neighbors between a pair of nodes will be a small number. Hence, the above equation 3 will give rise to a floating-point value. To make computation easier, since the linkage covariance value is going to be uniform for all nodes, we get rid of the division by N, the number of nodes. This reduces memory footprint considerably since there will be $|V|^2$ different values of linkage covariances. After this level of optimization, the equation for computing linkage covariance becomes as shown below.

$$LinkCov(p,q) = E[X\grave{a}p \cdot X\grave{a}q] = \sum_{k=1}^{N} x_{pk} \cdot x_{qk} \quad (4)$$

Additionally, we improve the time complexity of the algorithm by exploiting the symmetric nature of the adjacency matrix of a graph. This means that the linkage covariance of node (p, q) is equal to the linkage covariance of node (q, p). Hence this associative property

allows us to process only one half of the adjacency lists and obtain the linkage covariance of all nodes in the graph. When we employ this method to get the linkage covariance values for all nodes in a graph against all the other nodes in the graph, we found that due to large social graphs being sparse in nature, most of the node pairs did not have common neighbors, thereby creating a lot of zeroes as linkage covariance values. Since the linkage covariance characteristic vector, as described in Chapter 3 is a signature of each node, it is an overhead in the context of the data structure as well as unnecessary computation to proceed to the next step where each characteristic vector is sorted in monotonically decreasing order. Hence, after calculating the linkage covariance between two nodes, we write the value to the signature only if it is not zero, thereby stripping the signature vector of all zeroes. This significantly reduces the memory footprint of the implementation.

Finally, we made use of the dictionary data structure available in Python to make look-up and sort efficient and quick. Since dictionaries offer a significant speedup and are a memory efficient data structure, this vastly decreases the memory footprint of the implementation as against making use of matrices.

4.3 Parallelizing Linkage Covariance Calculation

At its core, the algorithm's complexity remains $O(|V|^2)$. However, we can reduce the time required to complete one iteration of the linkage covariance calculation by parallelizing its processing steps.

We use threads to parallel process the algorithm. We ensure that the shared resources, in this case, the social graph object available as an adjacency list, and the resultant linkage covariance signature vectors are limited to only the pre-computed neighbor sets for each node and the final linkage covariance value.

We have designed the thread in such a way that the algorithm would split the linkage covariance computation process for the graph equally over *k* threads given *m* cores of a processor where k<=m. Additionally, we sort the linkage covariance vector in monotonically decreasing order to ensure that the linkage covariance vector would be more characteristic of the node's quality of links towards its neighbors. It would also account for the random nature of thread execution. The sorting algorithm also contributes significantly towards the computational complexity of the algorithm and was improved by using an indirect sorting algorithm with quicksort at its core.

CHAPTER 5: EXPERIMENTAL SETUP

The objective of this thesis is an empirical evaluation of the inherent indistinguishability of nodes in graphs at different sizes. Our hypothesis is that larger graphs will have a larger percentage of their nodes with similar or identical structural signatures, which will make them more protected in the event of a re-identification attack. Thus, for the empirical component of this work:

1. We select a set of representative social graphs (described in Section 5.2.1);
2. We generate a number of synthetic graphs similar to the real datasets, which enable us to vary graph size while maintaining some of the original graph properties (Section 5.2.2);
3. We introduce two measures for inherent indistinguishability of nodes based on the linkage covariance presented before (Section 5.3);

We describe the computation platform in which we ran the experiments below.

5.1 Computing Platform

The experiments were conducted on the XSEDE supercomputing systems maintained by the University of Texas at Austin. The XSEDE system comprises of three different subsystems, namely, Stampede, Maverick and Wrangler. The experiments were primarily conducted on the Wrangler Data and Analytics system, whose technical specification is described below:

1. A 10 PB disk based storage system
2. A cluster of 96 Intel Haswell based analytics servers
3. A 0.5 PB shared flash storage system able to support data I/O at unprecedented rates across the analytics system

Each analytics node has 24 cores and 128 GB of volatile memory with both Infniband FDR and 40 Gb/s Ethernet connectivity. Wrangler has a maximum potential network throughput of 200 Gb/s for ingesting and accessing data.

We are grateful to the XSEDE supercomputing cluster for their allocation of the TACC Data Analytics System (Wrangler): 1,000.0 Node Hours and TACC Long-term Storage (Wrangler Storage): 500.0 GB. Wrangler has provided the computing cluster that has supported our research on the inherent indistinguishability of nodes in large social graphs.

5.1.1 Performance of Scalable Implementation

The implementation in Python was more memory efficient and computationally less intensive than the R implementation. This is highlighted by the time taken to execute the algorithm. The original R implementation was unable to scale for large social graphs beyond 5000 nodes in some cases. But the Python implementation with threads completed execution on an average of 7,530 seconds for a graph with 75,879 nodes.

To check correctness of implementation across R and Python versions, we used a smaller test dataset to verify the results. We considered the “terrorist network” graph dataset with 63 nodes and 154 edges. Both the R and Python implementations produced the same result. We also tested the implementations on “Sweden 5000” graph, a bigger graph dataset. The results were a match again. This ensured that the implementation in Python was correct and was hence used for computing the linkage covariances of larger social graph datasets.

5.2 Datasets

From the many public repositories of graph datasets [28, 29, 30], we selected three datasets of varying characteristics.

5.2.1 Real Datasets

We used *soc-Epinions1* [28], *soc-SlashDot0811* [29], and *BlogCatalog3* [30]. The properties of these datasets are provided in Tables 1 and 2. Each of these graphs have different graph characteristics, such as density, clustering coefficient and average path length.

Table 1: Size of the real social graph datasets.

Graph	Description	Nodes	Edges
BlogCatalog3	A social blog directory	10,312	333,983
soc-Epinions1	A who-trust-whom online social network	75,879	508,837
soc-SlashDot0811	A technology-related news website	77,360	905,468

Table 2: Properties of the real social graphs.

Graph	Graph Density	Average Path Length	Global Clustering Coefficient
BlogCatalog3	0.006282	2.382352	0.091392
soc-Epinions1	0.000177	4.307860	0.065679
soc-SlashDot0811	0.000303	4.024371	0.024157

5.2.2 Synthetic Datasets

Synthetic datasets are important for what-if scenarios, extrapolations, and simulations when real life social graphs are impossible to collect (e.g., a very large friendship graph between people). In our case, we want to be able to vary the size of the graph while preserving, as much as possible, the properties of *real* graphs. To this end, we selected two models for graph generation, based on the Erdős–Rényi model and the Forest Fire graph dynamics algorithm. The parameters of these models were chosen such that the synthetic graphs have similar graph properties to those of the real-life graphs. While generating graphs of different sizes using the

Erdős–Rényi model, we maintained the average degree constant and equal to that of the real graph. In the case of the Forest Fire model, we first calibrate the model to generate a graph with the same number of nodes and edges (as much as possible) as the real graph. We then use the same calibrated model parameters to generate graphs of various numbers of nodes.

5.2.2.1 Erdős–Rényi Model

The Erdős–Rényi model in graph theory is a model for random graph generation. As per the model, all graphs on a fixed vertex set with a fixed number of edges are equally likely. The Erdős–Rényi model of graph generation has traditionally been used in probabilistic methods to prove the existence of graphs satisfying various properties. The model has also been used to provide a solid theory of what it means for a property to hold good for almost all different types of graphs.

The Erdős–Rényi model can be generated using different ways:

1. Specifying the number of nodes and the number of edges. In this case, from the collection of all graphs that have the specified number of nodes and edges, a graph is chosen uniformly at random. This is generally called the G (V, E) model, where V is the number of vertices and E, the number of edges.
2. Specifying the number of nodes and an edge probability along with the number of nodes, a graph is generated at random. An edge is included in the graph with probability p that is independent from all the other edges. This is generally called the G (V, p) model, where V denotes the number of vertices and p, the edge probability.

$$p^E\ (1-p)^{\binom{V}{2}-E}\ (5)$$

Since p is a probabilistic value, it ranges from 0 to 1. As p tends from 0 to 1, the graph model becomes more likely to generate graphs with more edges and vice versa. The G (V, p)

model is commonly used as it allows for freedom with choosing edges using the probability value p.

For the Erdős–Rényi model, the parameters were tuned for fair comparison by using the edge probability value given by Equation 5.

$$Edge\ Probability\ p = \frac{Number\ of\ real\ edges}{Number\ of\ possible\ edges} \quad (6)$$

where the number of real edges corresponds to the number of edges of the real-life graph which we are trying to simulate; and, the number of possible edges is the number of edges in a fully connected graph.

5.2.2.2 Forest Fire Model

The Forest Fire Model [31] is based on having new nodes attach to a network. It is a rich enough class of model that was specifically designed to model densification power law with shrinking effective diameters, and nodes with high out-degrees, as is observed in most real social graphs.

Let us consider a node V_1 joining the network at time t > 1, and let G_t be the graph constructed until now. V_1 forms out-links to the already existing nodes in G_t by first choosing an ambassador node V_2 uniformly at random. Now a link is formed between V_1 and V_2. Then, a random number x is generated binomially distributed with mean $(1 - p)^{-1}$. V_1 then selects x links incident to V_2, choosing from both in and out links. The model selects in-links with probability r times less than out-links. Let V_{21}, V_{22}, ..., V_{2x} denote the other ends of these selected links. Using these, the node V_1 forms out-links to V_{21}, V_{22}, ..., V_{2x} and then goes through the process of selecting links recursively to each of V_{21}, V_{22}, ..., V_{2x}. Graph construction is acyclic, since the nodes with its edges that is already burnt are not visited another time. Thus, the

"burning" of links in Forest Fire model begins at V_2, spreads to V_{21}, V_{22}, ..., V_{2x}, and continues recursively until it dies out.

The Forest Fire model requires three parameters for graph generation - forward spreading probability, backward factor and number of ambassadors. In every time step, a new vertex is added to the graph. The new vertex chooses an ambassador (or more than one) and starts a simulated forest fire at its ambassador(s). The fire spreads through the edges with a spreading probability. The fire may also spread backwards on an edge. When the fire ends, the newly added vertex is connected to the vertices "burned" in the previous fire. Unlike Erdős–Rényi, the Forest Fire model is a growing graph model, as such the nature of the graph generation is purely random depending solely on the probabilistic values provided.

5.2.2.3 Synthetic Graphs

Using the algorithms described above, we generated synthetic graphs with the number of nodes reduced by factors of 2, 5 and 10 with respect to the number of nodes of the real graphs. Table 3 provides the parameters that are used to generate the synthetic graphs as in our description above.

Table 3: Values for the probabilistic parameters to be passed to the synthetic graph generators of the real datasets.

Dataset	Erdős–Rényi	Forest Fire		
	p	fw_prob	bw_factor	ambassadors
BlogCatalog3	0.006282	0.400	0.496	5
soc-Epinions1	0.000177	0.350	0.560	2
soc-Slashdot0811	0.000303	0.354	0.560	3

Tables 4, 5 and 6 below provide the graph characteristics of the real graphs BlogCatalog3, soc-Epinions1 and soc-Slashdot0811 datasets and their synthetic equivalents respectively.

Table 4: Characteristics of the Original and Synthetic Datasets derived from the BlogCatalog3 graph dataset.

Graph	Nodes	Edges	Graph Density	Average Path Length	Global Clustering Coefficient
BlogCatalog3	10312	333983	0.006282	2.382352	0.091392
Erdos-Renyi	10,312	334,892	0.006299	2.653722	0.006287
Erdos-Renyi/2	5,156	166,991	0.012566	2.425060	0.012570
Erdos-Renyi/5	2,062	66,783	0.031429	2.094773	0.031363
Erdos-Renyi/10	1,031	33,391	0.062887	1.952818	0.062689
Forest-Fire	10,312	331,710	0.006239	2.422267	0.054213
Forest-Fire/2	5,156	158,719	0.011943	2.350246	0.086814
Forest-Fire/5	2,062	56,412	0.026548	2.245819	0.137532
Forest-Fire/10	1,031	25,641	0.048291	2.153811	0.198094

Table 5: Characteristics of the Original and Synthetic Datasets derived from the soc-Epinions1 graph dataset.

Graph	Nodes	Edges	Graph Density	Average Path Length	Global Clustering Coefficient
soc-Epinions1	75879	508837	0.000177	4.30786	0.065679
Erdos-Renyi	75,879	508,965	0.000177	4.619275	0.000180
Erdos-Renyi/2	37,939	254,415	0.000354	4.354577	0.000369
Erdos-Renyi/5	15,175	101,762	0.000884	3.972669	0.000899
Erdos-Renyi/10	7,587	50,877	0.001768	3.721367	0.001678
Forest-Fire	75,879	506,502	0.000176	4.040610	0.018941
Forest-Fire/2	37,939	252,780	0.000351	3.924914	0.028309
Forest-Fire/5	15,175	99,559	0.000865	3.760569	0.045383
Forest-Fire/10	7,587	49,660	0.001726	3.622492	0.065230

Table 6: Characteristics of the Original and Synthetic Datasets derived from the soc-Slashdot0811 graph dataset.

Graph	Nodes	Edges	Graph Density	Average Path Length	Global Clustering Coefficient
soc-Slashdot0811	77,360	905,468	0.000303	4.024371	0.024157
Erdos-Renyi	77,360	904,816	0.000303	3.865760	0.000294
Erdos-Renyi/2	38,680	452,734	0.000605	3.697908	0.000612
Erdos-Renyi/5	15,472	181,093	0.001513	3.395790	0.001484
Erdos-Renyi/10	7,736	90,546	0.003026	3.124380	0.003003
Forest-Fire	77,360	910,673	0.000306	3.359442	0.011430
Forest-Fire/2	38,680	451,978	0.000604	3.302968	0.019180

Table 6 (Continued)

Graph	Nodes	Edges	Graph Density	Average Path Length	Global Clustering Coefficient
Forest-Fire/5	15,472	179,552	0.001500	3.156973	0.033705
Forest-Fire/10	7,736	87,341	0.002919	3.048142	0.051937

5.3 Metrics for Computing Inherent Indistinguishability of Nodes in a Graph

We measure how "lost in the crowd" each node of a graph is in terms of its underlying structural similarity with other nodes in the graph. We do this by examining the linkage covariance signature of each node of the graph and comparing it with the linkage covariance signature of every other node in the graph.

We report the percentage of nodes that have unique linkage covariance signatures and refer to it as a *risk*. Intuitively, the higher the percentage of nodes with unique linkage covariance, the more vulnerable they are to attack.

As a measure of *safety*, we report the average number of replicas in terms of linkage covariance signature that the nodes with non-unique signatures have in the graph. The larger the mean number of replicant nodes are, the harder should be for an attacker to properly identify the node. This safety measure is related to the concept if k-anonymity in a graph. K-anonymity is one objective of graph anonymization, in which each node in the anonymized graph belongs to a group of at least k nodes with identical properties. While previous literature defined such node properties as degree, graph density, and structural properties such as average path length and neighborhood characteristics, we considered the linkage covariance. Because linkage covariance captures more topological information about the node's neighborhood than degree, graph density or the structural properties such as average path length and neighborhood characteristics, we believe it is a stronger measure of indistinguishability of nodes in a graph.

These two metrics allow us not only to quantify the inherent indistinguishability of nodes in a family of graphs of various sizes, but also to compare different graphs.

CHAPTER 6: EMPIRICAL RESULTS

To measure node indistinguishability, we plot the cumulative distribution function of the number of nodes in groups with the same linkage covariance signatures against the natural logarithm of the size of the groups. This allows comparing graphs of different sizes, and the real graphs and their synthetic equivalents.

In each case, we generated 5 synthetic graphs for every set of parameters for every model. This chapter reports results based on the average of these runs.

6.1 BlogCatalog3

Figure 5 shows that 99.4% of the nodes in the graph have unique linkage covariance signatures. This means that the risk index is high, as also shown in Figure 9. Moreover, due to the small size of the graph, the synthetic graphs generated by both the Erdős–Rényi and Forest Fire models have at most two groups of nodes each based on linkage covariance signatures. Specifically, all nodes in all the FF graphs have unique linkage covariance vectors, thus any node is uniquely identifiable by an attacker if given sufficient information about its 2-hop neighborhood. The ER graphs have two groups of nodes each based on linkage covariance signatures. This is reflected by the single data point for FF graphs in the Figure 5.

6.2 Soc-Epinions1

The soc-Epinions1 graph is inherently more private than the BlogCatalog3 graph: 33.8% of the nodes in soc-Epinions1 are uniquely identifiable (as compared to 99.9% in BlogCatalog).

Unlike in the previous example, the Erdős–Rényi graphs of all sizes generated (shown in red in Figure 6) are more inherently private than the real dataset: a very small percentage of nodes are uniquely identifiable (0.6% – 6.4% for graphs of 37,940, 15,176, 7,587 nodes, respectively), and a larger percentage of nodes have more replicas than the nodes in the real dataset. Thus, the ER graphs have both a low risk index and a high safety index.

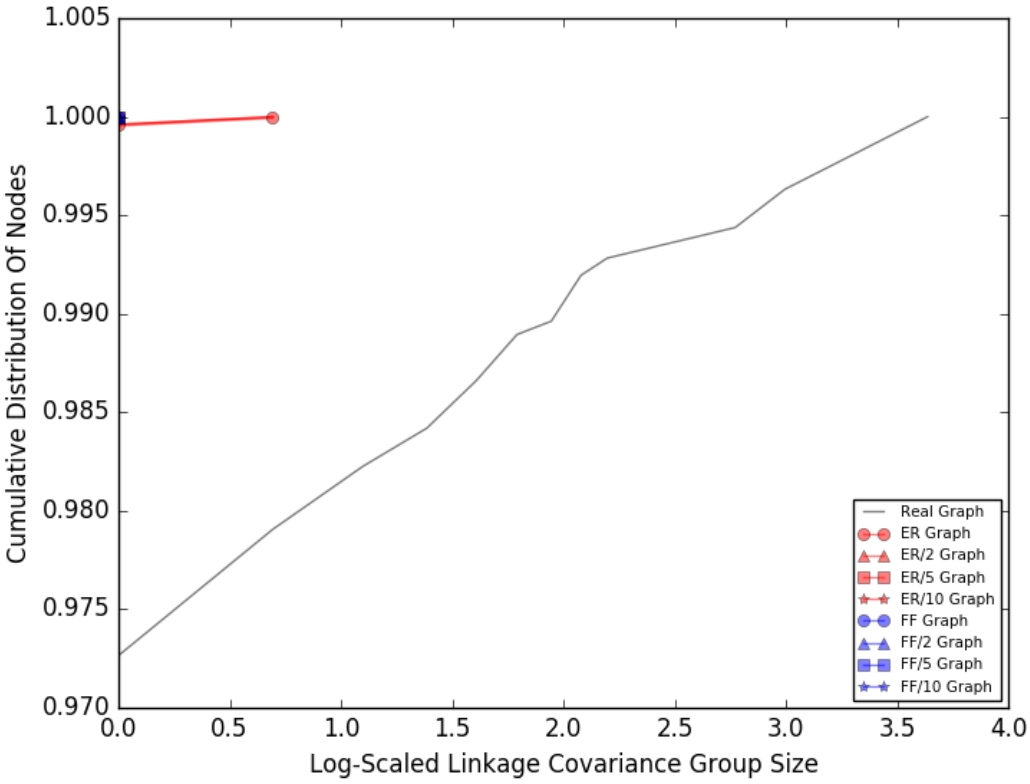

Figure 5: Cumulative distribution function plot of the BlogCatalog3 dataset showing the identical groups of the real and the synthetic datasets.

Forest Fire graphs, on the other hand, appear less private than the real graph: 59.79% of their respective nodes are uniquely identifiable (thus, a high risk index) and of the remaining 40.21% nodes with replicas, 65.4% have fewer than 50 replicas.

In terms of how inherent indistinguishability of nodes in a graph fares with scale, we observe that the larger the graph, the higher the indistinguishability of nodes in a graph. For

example, for the ER family of graphs, the largest graph (of 75,879 nodes) has 0.25% of uniquely identifiable nodes while the smaller (of 7,587 nodes) has 6.44%. At the same time, the number of replicas for the nodes with similar linkage covariance signatures with others varies between 99.4% to 97.7%. The same pattern is evident for the FF family of graphs. The nodes with uniquely identifiable linkage covariance signatures vary from 59.79% to 70.36%. The number of replicas for the nodes with the similar linkage covariance signatures with others varies between 40.21% to 29.64%.

In this case, the real dataset is in between ER and FF graph families and follows more accurately the FF slopes.

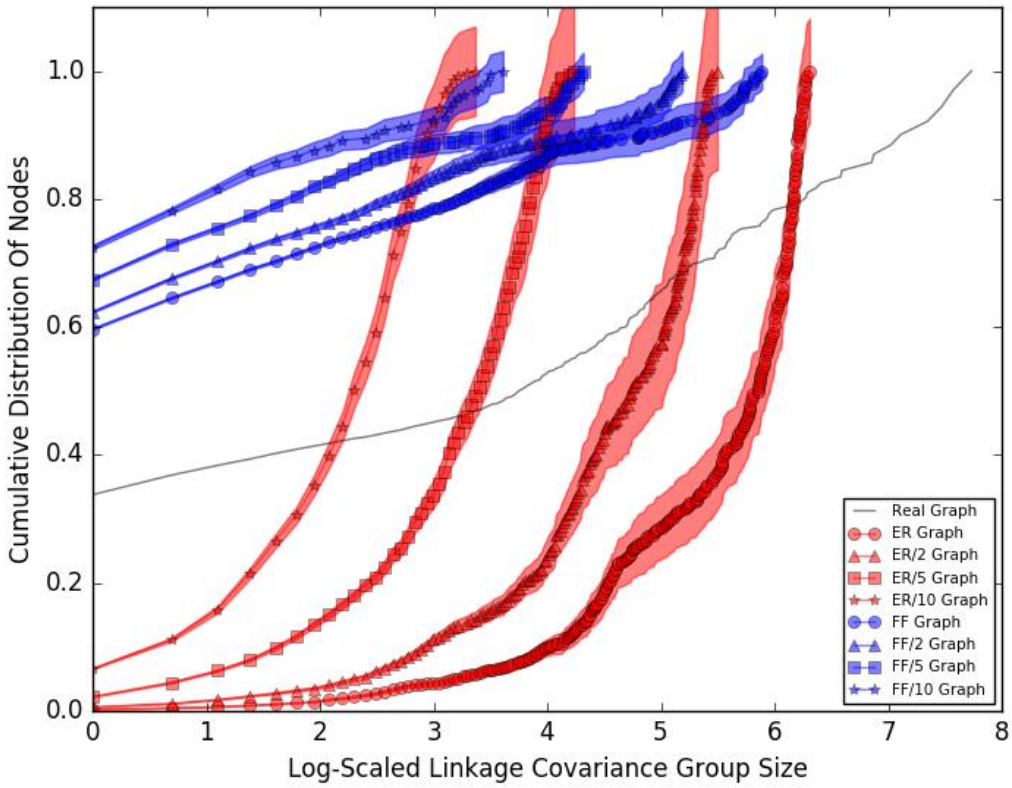

Figure 6: Cumulative distribution function plot of the soc-Epinions1 dataset showing the identical groups of the real and the synthetic datasets.

6.3 Soc-Slashdot0811

The soc-Slashdot0811 graph is inherently more private than the BlogCatalog3 graph, but less private than the soc-Epinions1 graph: 42.03% of the nodes in soc-Slashdot0811 are uniquely identifiable (as compared to 99.9% in BlogCatalog3 and 33.8% in soc-Epinions1). The Erdős–Rényi graphs of all sizes generated (shown in red in Figure 7) are more inherently private than the real dataset: a very small percentage of nodes are uniquely identifiable (1.5% – 53.24% for graphs of 38,930, 15,572, 7,786 nodes, respectively), and a larger percentage of nodes have more replicas than the nodes in the real dataset. Thus, the ER graphs have both a low risk index and a high safety index.

Forest Fire graphs, on the other hand, appear less private than the real graph: 81.6% of their respective nodes are uniquely identifiable (thus, a very high risk index) and of the remaining 18.4% nodes with replicas, 94.22% have fewer than 50 replicas.

In terms of how inherent indistinguishability of nodes in a graph fares with scale, we observe that the larger the graph, the higher the indistinguishability of nodes in a graph in this case too. For example, for the ER family of graphs, the largest graph (of 77,860 nodes) has 1.56% of uniquely identifiable nodes while the smaller (of 7,587 nodes) has 53.25%. At the same time, the number of replicas for the nodes with similar linkage covariance signatures with others varies between 98.5% to 46.76%. This pattern is evident for the FF family of graphs. The nodes with uniquely identifiable linkage covariance signatures vary from 1.5% – 53.24%. The number of replicas for the nodes with the similar linkage covariance signatures with others varies between 40.21% to 29.64%.

In this case, the real dataset is in between ER and FF graph families and follows more accurately the FF slopes.

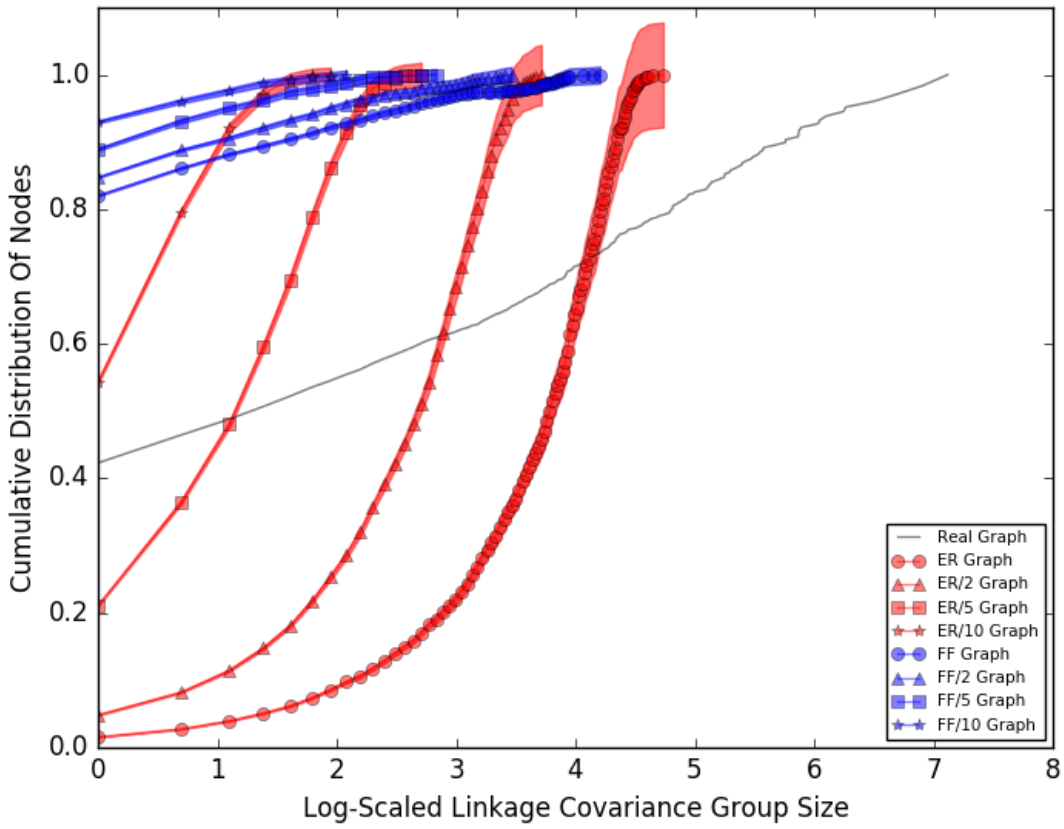

Figure 7: Cumulative distribution function plot of the soc-Slashdot0811 dataset showing the identical groups of the real and the synthetic datasets.

6.4 Comparing the Inherent Indistinguishability of Nodes of Different Graphs

A different way to look at the risk and safety indices of different graphs is shown in Figures 8, 9, 10, 11. The X axis shows the risk metric (the percentage of nodes with uniquely identifiable linkage covariance signatures) and the Y axis shows the safety metric (the average number of nodes with replicas in the linkage covariance signatures in the graph). The closer a graph is to the top left corner of the plot, the more private it is; conversely, the closer to the bottom right corner of the plot, the less private it is. This is an extreme scenario.

Figure 8 represents the comparison of the soc-Epinions1 real and synthetic datasets. As already inferred from the previous plots, the ER family of graphs is significantly more private than the FF family: both their risk index is lower and their safety index is higher. Again, the real dataset is shown in between, with a much higher safety index but a moderate risk index. When comparing graphs of the same time but at different scales, the larger the graph, the more private it is in both metrics.

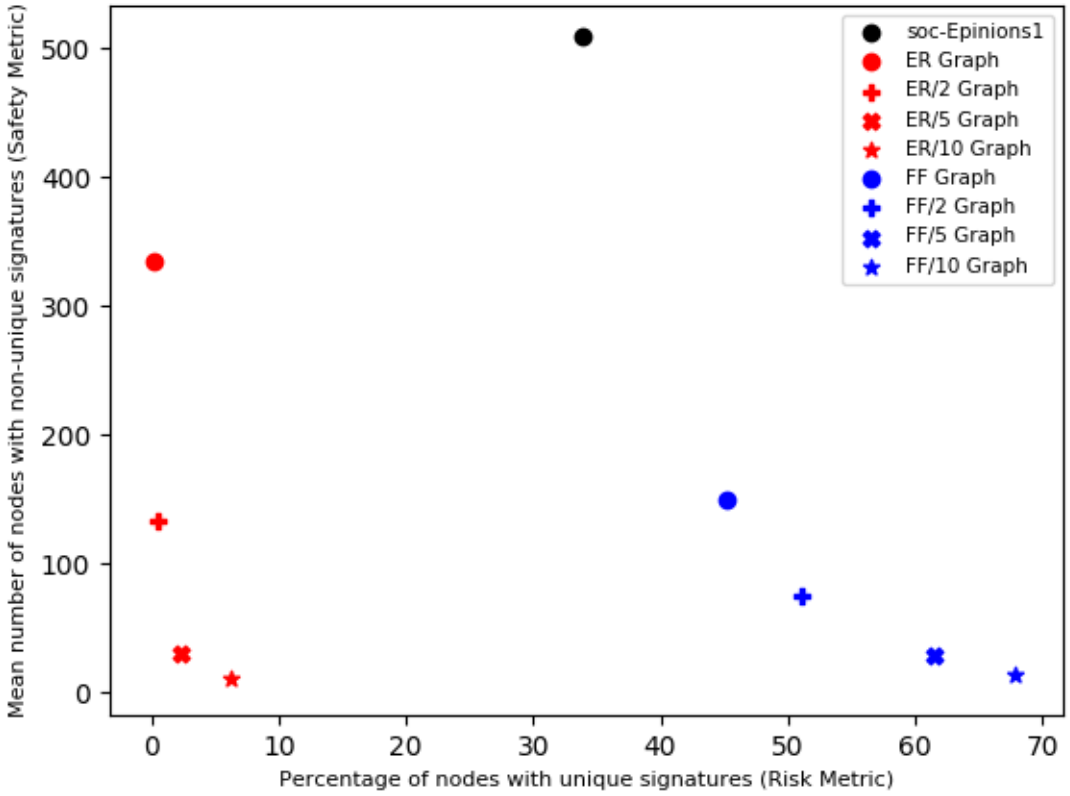

Figure 8: Indistinguishability comparison plot of the soc-Epinions1 dataset and its synthetic equivalents.

The extreme scenario is best shown by the BlogCatalog1 plot in Figure 9, where the real graph is inherently private while the synthetic graphs are at the extreme).

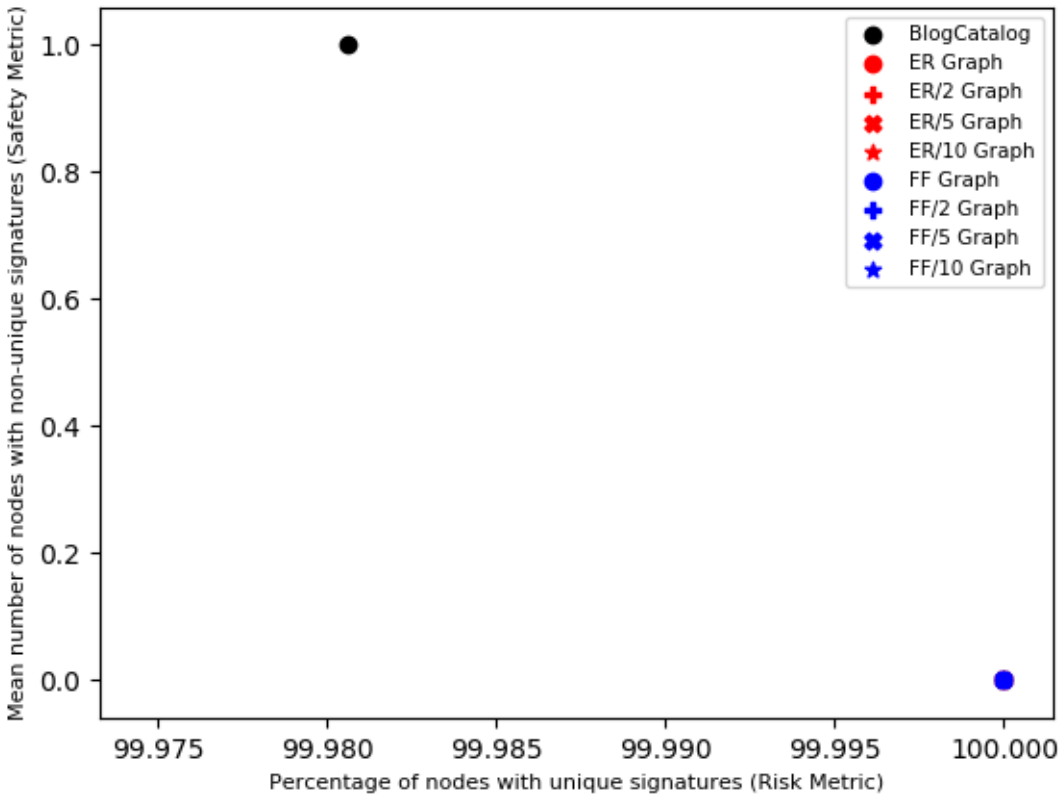

Figure 9: Indistinguishability comparison plot of the BlogCatalog1 dataset and its synthetic equivalents.

Figure 10 shows the comparison of the soc-Slashdot0811 real and synthetic datasets. As already inferred from the plot in Figure 8 for the soc-Epinions1 graph, the ER family of graphs is significantly more private than the FF family in this case: both their risk index is lower and their safety index is higher. The real dataset is shown in between, with a much higher safety index but a moderate risk index.

Figure 11 places all real datasets in the same normalized space and shows how we can compare the inherent indistinguishability of nodes of different graph datasets. The soc-Epinions is the clear winner, with lowest risk and the highest safety indices of all.

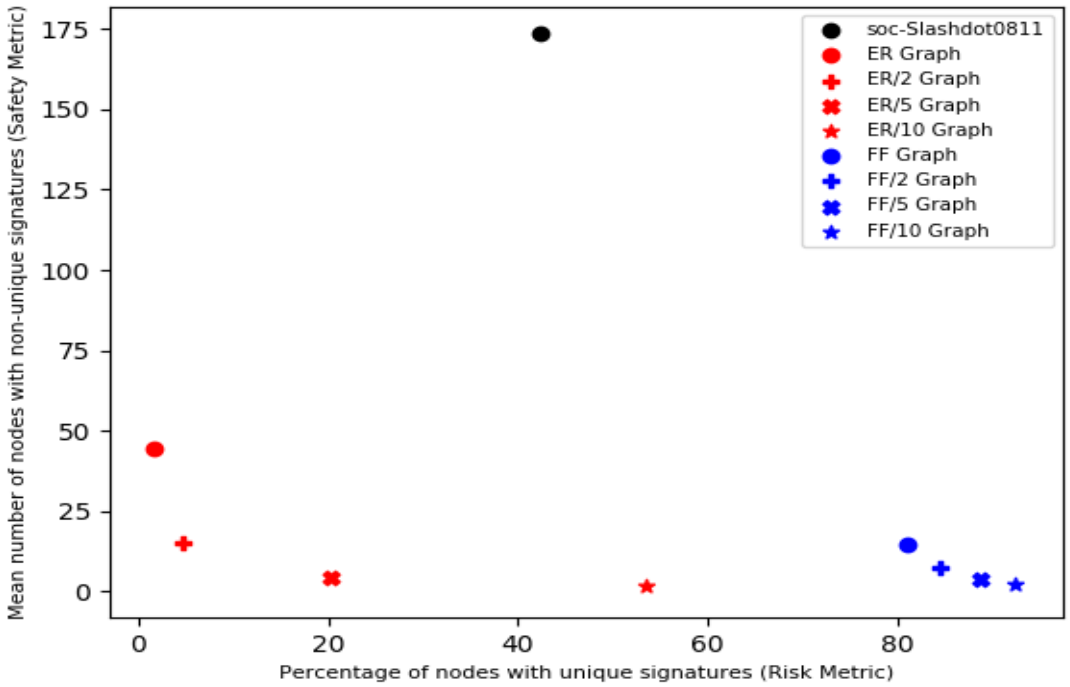

Figure 10: Indistinguishability comparison plot of the soc-Slashdot0811 dataset and its synthetic equivalents.

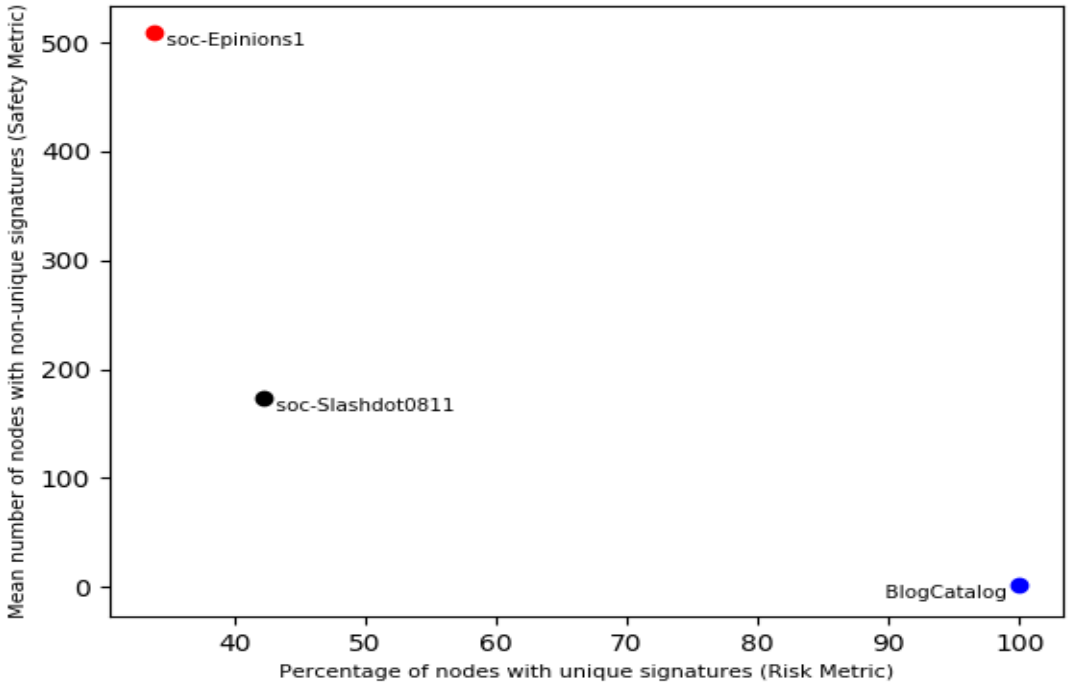

Figure 11: Comparison of indistinguishability of nodes of all the real social graph datasets.

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

In this thesis, we asked whether large social graphs are inherently more private than their smaller counterparts. To this end, we used the linkage covariance metric to represent the local position of individual nodes of a given graph. Empirical results show that large graphs are inherently more private than smaller graphs from the same family. In the process, we proposed a scalable implementation for the computation of linkage covariance and a set of metrics for quantifying inherent indistinguishability of nodes in a graph.

We also learned that the Erdős–Rényi graph generation model produces more private graphs than the Forest-Fire model: Erdős–Rényi graphs have fewer uniquely identifiably nodes and their nodes can be grouped in larger groups of the same linkage covariance.

This effort can be applied in the space of graph data anonymity in various ways. For example, one can set a desired maximum risk index and a desired minimum safety index and selectively apply anonymization techniques to the graph datasets that do not meet the desired criteria. Alternatively, the anonymization of a graph can selectively target the at-risk nodes (i.e., those with unique linkage covariance signatures or with small safety index) and perturb only their local neighborhoods.

This work also opens a new set of research directions.

- Improving linkage covariance methodology to k-hop neighborhood. While the current definition of linkage covariance includes the number of direct common neighbors

of two nodes in the graph, we could extend this definition to consider the more remote common neighbors (for example, a 2-hop neighbor of A and a direct neighbor of B will contribute to the value of the new linkage covariance definition). This approach would capture structural information about a larger local neighborhood and be used to measure the resilience against a stronger attacker.

- Testing the claims of anonymity made by measuring the inherent indistinguishability of nodes in a graph using the metrics proposed in this thesis by creating an attack model that uses linkage covariance information to de-anonymize a dataset.
- Creating an anonymization technique based on node indistinguishability and linkage covariance. Linkage covariance vectors can be used as the underlying metric for quantifying structural difference between two graphs.

REFERENCES

[1] Narayanan, A., & Shmatikov, V. (2008, May). Robust de-anonymization of large sparse datasets. In Security and Privacy, 2008. SP 2008. IEEE Symposium on (pp. 111-125). IEEE.

[2] Bearman, P. S., Moody, J., & Stovel, K. (2004). Chains of affection: The structure of adolescent romantic and sexual networks 1. American journal of sociology, 110(1), 44-91.

[3] Sala, A., Zhao, X., Wilson, C., Zheng, H., & Zhao, B. Y. (2011, November). Sharing graphs using differentially private graph models. In Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference (pp. 81-98). ACM.

[4] Zheleva, E., & Getoor, L. (2008). Preserving the privacy of sensitive relationships in graph data. In Privacy, security, and trust in KDD (pp. 153-171). Springer Berlin Heidelberg.

[5] Hay, M., Miklau, G., Jensen, D., Towsley, D., & Weis, P. (2008). Resisting structural re-identification in anonymized social networks. Proceedings of the VLDB Endowment, 1(1), 102-114.

[6] Liu, C., & Mittal, P. (2016, February). LinkMirage: Enabling privacy-preserving analytics on social relationships. In 23nd Annual Network and Distributed System Security Symposium, NDSS (pp. 21-24).

[7] Ji, S., Li, W., Mittal, P., Hu, X., & Beyah, R. A. (2015, August). SecGraph: A Uniform and Open-source Evaluation System for Graph Data Anonymization and De-anonymization. In Usenix Security (pp. 303-318).

[8] Leskovec, J., Lang, K. J., Dasgupta, A., & Mahoney, M. W. (2009). Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Internet Mathematics, 6(1), 29-123.

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

[10] Aggarwal, C. C., Li, Y., & Philip, S. Y. (2011, December). On the hardness of graph anonymization. In Data Mining (ICDM), 2011 IEEE 11th International Conference on (pp. 1002-1007). IEEE.

[11] Solove, D. J. (2007). I've got nothing to hide and other misunderstandings of privacy. San Diego L. Rev., 44, 745.

[12] Backstrom, L., Dwork, C., & Kleinberg, J. (2011). Wherefore art thou R3579X?: anonymized social networks, hidden patterns, and structural steganography. Communications of the ACM, 54(12), 133-141.

[13] Ji, S., Mittal, P., & Beyah, R. (2016). Graph Data Anonymization, De-anonymization Attacks, and De-anonymizability Quantification: A Survey. IEEE Communications Surveys & Tutorials.

[14] Pedarsani, P., & Grossglauser, M. (2011, August). On the privacy of anonymized networks. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1235-1243). ACM.

[15] Ji, S., Li, W., Srivatsa, M., & Beyah, R. (2014, November). Structural data de-anonymization: Quantification, practice, and implications. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (pp. 1040-1053). ACM.

[16] Ji, S., Li, W., Yang, S., Mittal, P., & Beyah, R. (2016, April). On the relative de-anonymizability of graph data: Quantification and evaluation. In Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on (pp. 1-9). IEEE.

[17] Korula, N., & Lattanzi, S. (2014). An efficient reconciliation algorithm for social networks. Proceedings of the VLDB Endowment, 7(5), 377-388.

[18] Butts, C. T., & Carley, K. M. (2005). Some simple algorithms for structural comparison. Computational & Mathematical Organization Theory, 11(4), 291-305.

[19] Berlingerio, M., Koutra, D., Eliassi-Rad, T., & Faloutsos, C. (2013, August). Network similarity via multiple social theories. In Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on (pp. 1439-1440). IEEE.

[20] Boldi, P., Bonchi, F., Gionis, A., & Tassa, T. (2012). Injecting uncertainty in graphs for identity obfuscation. Proceedings of the VLDB Endowment, 5(11), 1376-1387.

[21] Bonchi, F., Gionis, A., & Tassa, T. (2014). Identity obfuscation in graphs through the information theoretic lens. Information Sciences, 275, 232-256.

[22] Nguyen, H. H., Imine, A., & Rusinowitch, M. (2015, August). Differentially private publication of social graphs at linear cost. In Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on (pp. 596-599). IEEE.

[23] Smith, G. (2009, March). On the foundations of quantitative information flow. In International Conference on Foundations of Software Science and Computational Structures (pp. 288-302). Springer Berlin Heidelberg.

[24] Xue, M., Karras, P., Chedy, R., Kalnis, P., & Pung, H. K. (2012, October). Delineating social network data anonymization via random edge perturbation. In Proceedings of the 21st ACM international conference on Information and knowledge management (pp. 475-484). ACM.

[25] Cheng, J., Fu, A. W. C., & Liu, J. (2010, June). K-isomorphism: privacy preserving network publication against structural attacks. In Proceedings of the 2010 ACM SIGMOD International Conference on Management of data (pp. 459-470). ACM.

[26] Wu, W., Xiao, Y., Wang, W., He, Z., & Wang, Z. (2010, March). K-symmetry model for identity anonymization in social networks. In Proceedings of the 13th international conference on extending database technology (pp. 111-122). ACM.

[27] Sharad, K. (2016, October). True friends let you down: Benchmarking social graph anonymization schemes. In Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security (pp. 93-104). ACM.

[28] Richardson, M., Agrawal, R., & Domingos, P. (2003, October). Trust management for the semantic web. In International semantic Web conference (pp. 351-368). Springer Berlin Heidelberg.

[29] Leskovec, J., Lang, K. J., Dasgupta, A., & Mahoney, M. W. (2009). Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Internet Mathematics, 6(1), 29-123.

[30] Zafarani, R., & Liu, H. (2009). Social computing data repository at ASU. [http://socialcomputing.asu.edu]. Tempe, AZ: Arizona State University, School of Computing, Informatics and Decision Systems Engineering.

[31] Leskovec, J., Kleinberg, J., & Faloutsos, C. (2007). Graph evolution: Densification and shrinking diameters. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(1), 2.

[32] Narayanan, A., & Shmatikov, V. (2009, May). De-anonymizing social networks. In Security and Privacy, 2009 30th IEEE Symposium on (pp. 173-187). IEEE.

[33] Narayanan, A. (2008, December 15). The Fallacy of Anonymous Institutions. Retrieved May 8, 2017, from https://33bits.org/2008/12/15/the-fallacy-of-anonymous-institutions/

[34] Erdös, P., & Rényi, A. (1959). On random graphs, I. Publicationes Mathematicae (Debrecen), 6, 290-297.