CHALMERS

Security Breach in Anonymized Social networks

Master of Science Thesis in Computer Science

MINA AGHVAMI
HILINA KIDANE

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, December 2012

{An not to long headline describing the content of the report}
[A Subtitle that can be Very Much Longer if Necessary]

MINA AGHVAMI
HILINA KIDANE

Examiner: Philippas Tsigas

Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

ABSTARCT

The purpose of the study was to show a threat of privacy leak for members of social networks. The general way to protect the privacy of these networks is using anonymization methods. On the other hand, deanynimization methods exist against these anonymization algorithms. Those methods were introduced and one of them was explained in details. Improvements of the algorithm were suggested by changing different parameters of it based on the social networks characteristics. These parameters were limiting similarity score, eccentricity, sorting, degree limit, ordering, and degree comparison and running the algorithm in two stages. The result of the experiment showed that by choosing proper values for similarity score, eccentricity, degree and also sorting the nodes, we can improve the performance of the algorithm by preventing it from propagating in the wrong direction. Sorting the nodes will also improve the result with a similar reason.

ACKNOWLEDGMENT

We would like to express our gratitude to all people who helped us to work on this master thesis. We would like to thank our supervisor, Philippas Tsigas who helped and guided us during the time we were working on the thesis and also our families who were encouraging and helping us all the time. We have also to thank our friends who were with us all the time and improved our work with their ideas.

Table of Content

1 INTRODUCTION

This chapter contains the background of social networks and their privacy, related works on this area and the general purpose of the thesis.

1.1 Background

1.1.1 Properties of social networks

Social networks are usually described as structures which contain individuals or organizations, represented as nodes and also their connections which are shown as edges. Nodes and edges are characterized by having different identifiers. Nodes can be identified by a name, an ID or a telephone number. Edges represent types of relationships like friendship, business and common interest. Links between nodes are of two types: directed and undirected. To form an undirected link both nodes have to agree to each other. Depending on what the link represents different social networks have different link types. For example, in Facebook or Twitter, undirected edge will be created if two users become be friends and this will happen if one sends a request and the other accepts. In general, online social networks are constructed in a way to represent the real world networks. A person having a lot of friends is represented as a node having high degree in the network called a hub. As there are people who don't have that much interaction with others, there are also nodes which have very small degree or even single nodes (i.e. nodes with no edge).Singleton or nodes with degree zero are not called social nodes.

The structure of SNs is studied by different researchers and they have shown that these networks have power law, scale free and small world properties [2]. The degree distribution of SNs is power law meaning there are few nodes which have very high degree but the majority is of lower degrees. In Small world networks, wide portion of the nodes are not connected to each other but it is possible to reach them with a small path, in other words there is a shortest path to reach them. The high scale free property is a measure which explains that high degree nodes are more connected to other high degree nodes. In real world social networks, it is like famous people have more connection with other famous people. In addition, network densification, an increase in network density with time, and shrinking diameter, a decrease in the effective diameter of the network as more nodes join the network, are studied properties in recent times.

Social networks have indeed an important role in influencing the online personal and business interactions and also information exchange. These networks are also one means of information for researchers, advertisers and application developers. Studying these network's properties and structures, results in a great change to the future technology. To mention some of these influences: new development of browser plug-ins, web search tools and improvement in other online social activities like viral marketing and political campaign.

1.1.2 Privacy aspects of social networks

As a consequence of the fast growth of social networks, more attackers become interested in this area which demands for further studies about these networks. Different social networks have their own privacy policy which can be set either by default or by the choice of their users. The default privacy policy might be public, in which, the user is able to change it in his

best interest or keep it as it is. The risk of setting the privacy public by itself is another topic. The focus of this thesis is around the topic that, even though the privacy policy is set private, a lot of information is still available for the owners of the networks.
Sometimes the owners of the social networks are allowed to sell the information to researchers, advertisers or application developers. The information would probably be sold out in a way that the identities of the users are hidden. The technique of hiding the identities is called anonymization. Some of these anonymization techniques will be described later.

1.2 Purpose of the thesis

The main purpose of the thesis is to show a practical way of privacy leak for social network users. This is done by implementing deanonymization algorithm which works with only couple of information available as input. Besides, finding strong parameters to make the algorithm more efficient with coverage, execution time and accuracy wise is also part of the milestone.

1.3 Related work

As mentioned earlier security of the social networks has become very important recently. There have been some researches on social networks anonymization algorithms in order to strengthen the security of the information given out. On the other hand de-anonymization algorithms are also improving to breach the security. Both techniques are further explained below.

1.3.1 Anonymization algorithms

Anonymization of social networks, means changing the structure of the data in a way that the identities of the people are kept anonymous. If we consider the social network as a graph, the simplest way of anonymization is replacing the identifiers of the elements with random strings or random integers so the data can be used for other purposes like researching. In these situations researchers don't care about who is connected to who; The important thing is the structure of the graph like its connectivity , node to node distances , frequencies of small sub graphs or the extent to which it can be clustered.

Recently it has been shown that this simple way of replacing identities with random letters is not useful. The reason is that in most of the social networks, nodes represent humans in the world. These people are mostly a member of different social networks. If for example, the edges of a social network represent friendship relationship, people are almost connected to the same people in all of the social networks in which they are a member of. In short, adversaries can take advantage of the structural information such as degree and neighborhood graph, gained from other social networks in order to deanonymize a person.
More advanced anonymization algorithms change the structure of the graph in a way that auxiliary information would not represent a unique node. On the other hand, it should not change the structure of the graph in a way that it becomes useless for research. Furthermore it is hard to know what kind of auxiliary information is available for the adversary so the anonymization algorithm should be resistant under any potential deanonymization algorithm.
Some of the more advanced anonymization algorithms are k-degree, k-neighborhood and k-symmetry algorithms. In the K-degree anonymity, for each node there are at least k-1 other nodes, having the same degree. In K-neighborhood anonymity, for each node there are at

least k-1 other nodes sharing the same neighborhood structure. In the K-symmetry model the network is modified in a way that for each node there exist at least k-1 other nodes which can be an image of it under some automorphism [4] .One problem of this method could be that the statistical properties of the graph might change. In this case sampling methods can be helpful in order to have a rough measurement of the statistical properties.

1.3.2 De-anonymization Algorithm

The counter part of anonymization algorithms are deanonymization algorithms in which the attackers try to find out the identities of the nodes in the social networks with the help of structural information gained from other networks, called auxiliary networks .There are lots of different algorithms for this purpose but they can generally be categorized in two different kinds, active and passive.

1.3.2.1 Active attacks

In this method the attacker makes some changes in the social network in order to deanonymize some nodes later. These changes can be adding some extra nodes to the network. For example in the real social network the attacker can create some fake users to become a member of the network and send friend request to some existing members. Later after anonymization of the network, the attacker starts with finding those added nodes and then with the help of the auxiliary information gained from other networks, continues finding their neighbours and this goes on until a noticeable portion of the network is deanonymized.

1.3.2.2 Passive attacks

In the passive attacks, the attacker does not make any change to the social network but tries to recognize some familiar nodes based on their characteristics. Those characteristics are either so outstanding or they can be found through auxiliary information and then later on with the help of auxiliary information, he will continue anonymizing the rest of the graph. Our deanonymization algorithm is classified under this type of attack.

2 MODEL
2.1 Threat model

The model we use needs three important things to be fulfilled before start. These inputs are: Anonymized or sanitized graph (Gsan), auxiliary graph (Gaux) and some amount of seeds. A short description of what these inputs are and how they can be achieved will be given out as follows.

- **Anonymized Graph**

Anonymized graph is the information in which identity of users is hidden by different anonymization methods explained before. This information can be available to attackers in many ways. One way is that the owners of the social networks themselves share this information with researchers, advertisers, application developers or other people. Information gathered by different third party applications and aggregated together can also be a great asset for the attackers. Besides, the data can be released in different other ways. For example; WellNet is a social network that contains medical record of employees of a company to facilitate different kind of cares. This network allows employers to have access to anonymized data so that they can follow up the medical status of their employees [6]. The other case is that most online social network users publish their photographs. There is a great probability that users who are present on the photograph are neighbors in the network. So even though their profile is hidden, just by taking the photograph, it is possible to get an anonymized graph from the social network [7].

- **Auxiliary Graph**

Auxiliary information is the initial information, in this subject a graph, which helps the main process of deanonymization. Basically the information should have a partial overlap with the anonymized graph in order to be considered useful for the attack. This information can be retrieved from the original graph by using crawling methods or some applications, which can exploit the sub graph by giving out the users who use the application. Aggregating projects, which collect different user's information and their relationships from various sources, are another source of information. Data can also be collected for governmental and law related tasks but used for malicious purposes by the attackers. Depending on the technique used the attacker might get a graph with varies overlap with the Original network. So the final result of the deanonymization algorithm greatly depends on this graph.

- **Seeds**

Seeds are a couple of nodes which exist on both sanitized and auxiliary graph. These seeds are the first members of our map, which will expand every time a node is added to it. So the seeds are used as an input and also driving engine for our propagation algorithm. There are different ways of finding seeds.
The first method is mentioned in section 1.3.2.1 which is based on [8]. Basically it is a de-anonymization algorithm but it is only capable of re-identifying very few amounts of nodes;

so it is more suitable for seed identification. We implement the active attack, which is also a walk based attack, and get some result but we don't use the algorithm for further work. The main reason is that some nodes have to be inserted before the anonymized graph is obtained so that when the graph is released the inserted nodes are used as a base to find the target nodes.

The second method is the one Arvind and Vitaly used for their algorithm in [1]. In this method the algorithm first finds a clique of n nodes in the auxiliary graph. Keeping in mind that our sanitized graph is anonymized; it takes only the *degree value* of these n nodes and the *common neighbor count* between pairs of nodes. The algorithm then continues finding the corresponding clique in the sanitized graph. It starts from a random node and check whether it has the same degree as the first node in the n clique, then it finds a neighbor of the node which has the appropriate neighbor count corresponding to the node in the clique. If such node is found, it will be added to the list and continues until it gets n number of nodes which are also clique in the target graph. Every time a random node is picked, it will check if it is a neighbor of those nodes which are added to the list so far. This algorithm works best when the graphs have dense edge and the overlap between them is high.

The third and the last method of getting seeds, is by selecting nodes from the ground truth file. This file contains the map between sanitized nodes and auxiliary nodes. According to the high scale free property of social networks, nodes with high degree are highly connected to other nodes of high degree. These high degree nodes are of great importance for our deanonymization algorithm to propagate.

The amount of seeds needed for a successful deanonymization depends on the type of the graph and the overlap between sanitized graph and auxiliary graph. The more the graph is connected and the overlap increases, the number of seeds needed decrease.
We have done some experiments with the methods mentioned above. We started with the seed finding algorithm based on a clique of n nodes in both graphs and the maximum size of clique we were able to find was 3. As a result, we decided not to use this technique for further experiment:

- Due to the nature of most of our graphs it was difficult to find a clique which exists in both graphs.
- Even though we found a clique the nodes in the clique are not with high degree so they are not good enough to keep the algorithm from dying out on early stages.
- The result obtained with these seeds is with a lot of false positive

Finally, based on the results we obtained we decide to give more focus on using the seeds by just selecting them from the ground truth file.

2.2 Success of the attack

Sometimes even though is not possible to clearly identify a node in the anonymized graph, it might be enough to reveal that there is a set. This set contains nodes which have similar properties as the target node. By property we mean similar degree or neighbourhood structure. Revealing this information can also be taken as an attack. By calculating how much information is revealed it is possible to measure the success of the attack. Besides, two other

metrics called coverage and accuracy are used to measure the success of an algorithm as mentioned in [7].Coverage represents the total percentage of nodes re-identified and accuracy refers to the percentage of nodes which are re-identified correctly. These metrics together with a parameter called degree limit (section 4.2) are used to effectively calculate the success of our algorithm.

Coverage = $\frac{Total\ number\ of\ nodes\ Re-identified}{Number\ of\ nodes\ in\ the\ target\ graph} * 100$ …….…… (Eq 1)

Note: Number of nodes in the target graph considered for this formula depends on the degree limit used for the specific case. For example if the graph has x number of nodes and y number of its nodes are with degree less than 2, then when we run the algorithm with degree limit 2. For the calculation of coverage *'Number of nodes in the target graph'* should be taken as x-y.

Accuracy=100–Falsepositive …….. (Eq 2)

False positive= $\frac{Number\ of\ nodes\ re-identified\ incorreclty}{total\ number\ of\ nodes\ re-identified} * 100$ …… (Eq 3)

3 METHODOLOGY

The first moves on this thesis work were to make the overall all draft of the algorithm, which is developed step by step throughout the journey, and also finding the necessary inputs for the work. Basically we took the base of the algorithm from the work of Arvind and Vitaly [1], and divided it into different functions in order to make it simple and be able to play with different parameters. About the implementation of the algorithm a pseudo code and description of each parameter is given below. When it comes to inputs, our algorithm is based on three important inputs called Sanitized (anonymized) graph, auxiliary graph and seeds. These input data are taken from different datasets which have properties of social networks.

3.1 Data sets

The data sets used for this thesis work are either given to us or we do crawling from the database. Basically we tried to use three different types of datasets .The need for having more than one dataset was to see how much the algorithm is effective in different datasets. The following subsections have brief explanations about each of the datasets.

3.1.1 Unknown social network

The graph we used from this dataset as a target graph G3 has 153711 nodes and 158116 edges. The degree distribution of the graph is shown in figure 3.1.1.1. Two other graphs, G1 and G2, from the same dataset are used as auxiliary information to deanonymize the target graph. G1 has 81830 nodes and 53367 edges and G2 with 61586 nodes and 90362 edges. These graphs have 33.7% and 57.14% edge overlap with G3 respectively. Figure 3.1.1.2 and Figure 3.1.1.3 shows the degree distribution of G1 and G2.

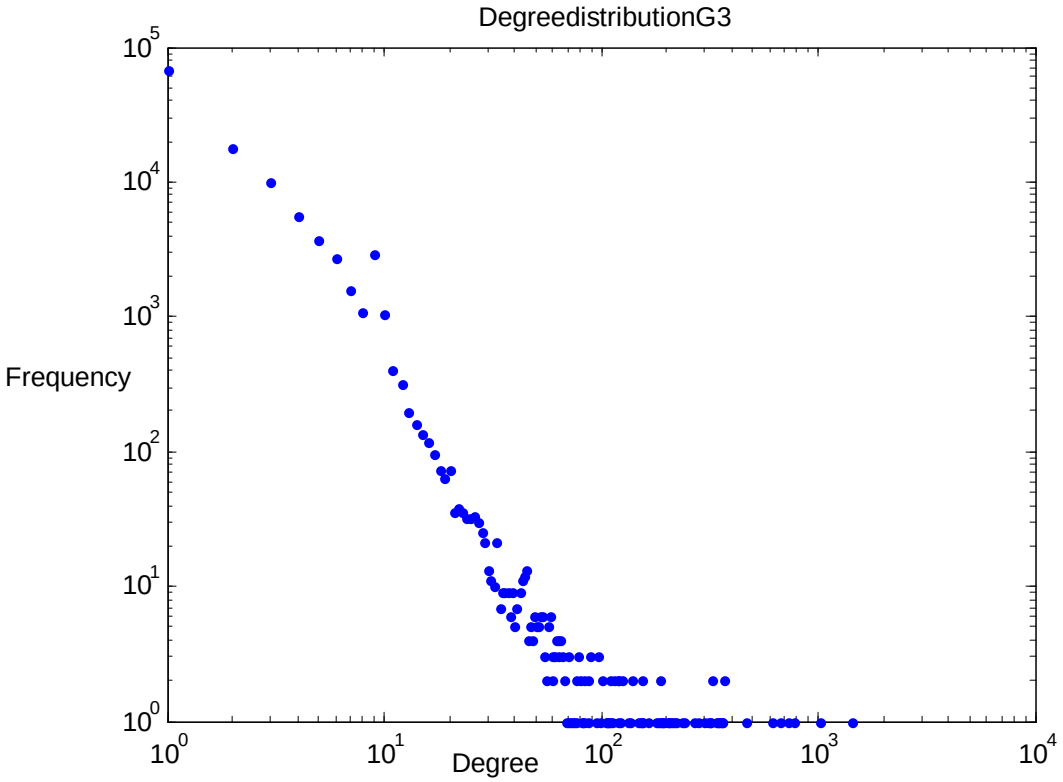

Figure 3.1.1.1 Degree distribution of G3

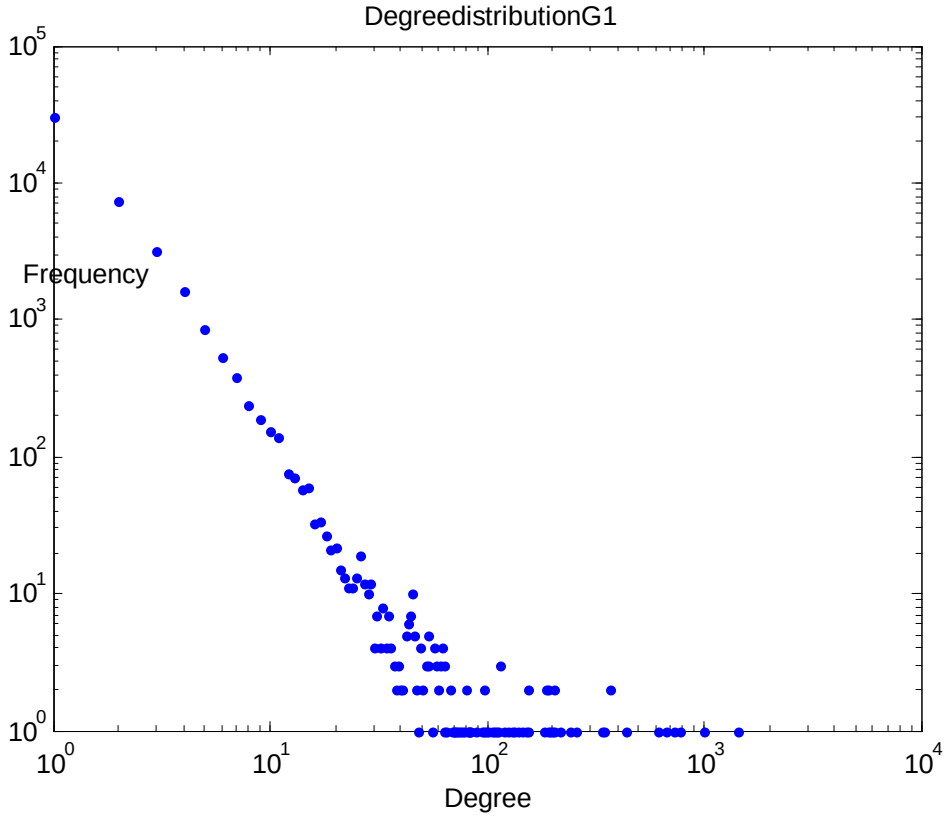

Figure 3.1.1.2 Degree distribution of G1

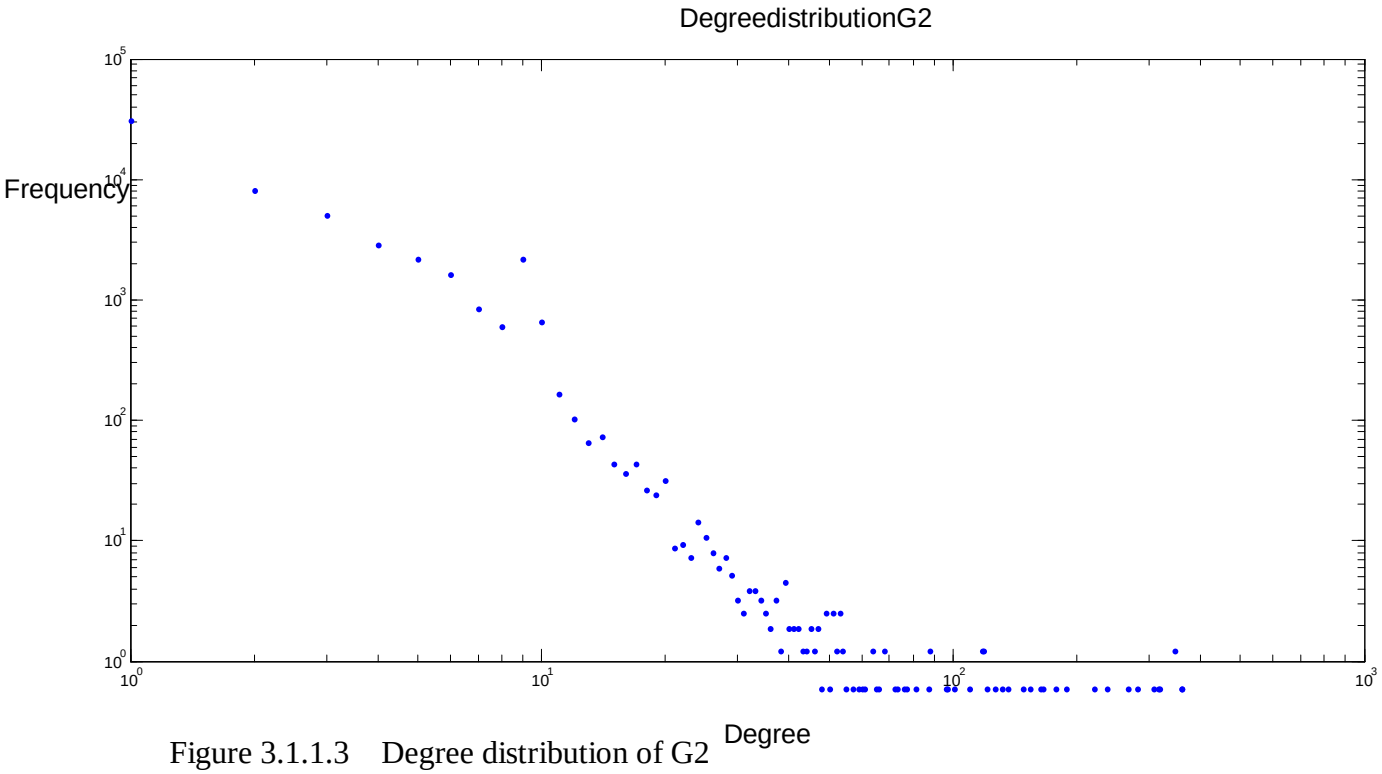

Figure 3.1.1.3 Degree distribution of G2

3.1.2 Slashdot

Slashdot is a website in which users share science and technology news with others. The reader uses the comment section for giving feedback or some additional information about the post. The graph is taken from [9], as it is stated on the paper "the network contains friend or foe links between users in which it is collected by allowing the users to tag their friends or foes in November 2006".The original graph was obtained in a text format and it has 77360 nodes and 9054680 edges. In order to simplify the experiment, we use a perturbation method to create two smaller graphs. By perturbation, we mean randomly deleting nodes with degrees below some limit.Then two graphs, one with 40993 nodes and 173227 edges and the other with 30474 nodes and 100000 edges, are obtained. We named them as slashdot1 and slashdot2 respectively. These graphs have 57.45% node overlap and 36.6% edge overlap with each other. Their degree distribution is power law as most other social networks. Refer figure 3.1.2.1 and figure 3.1.2.2.

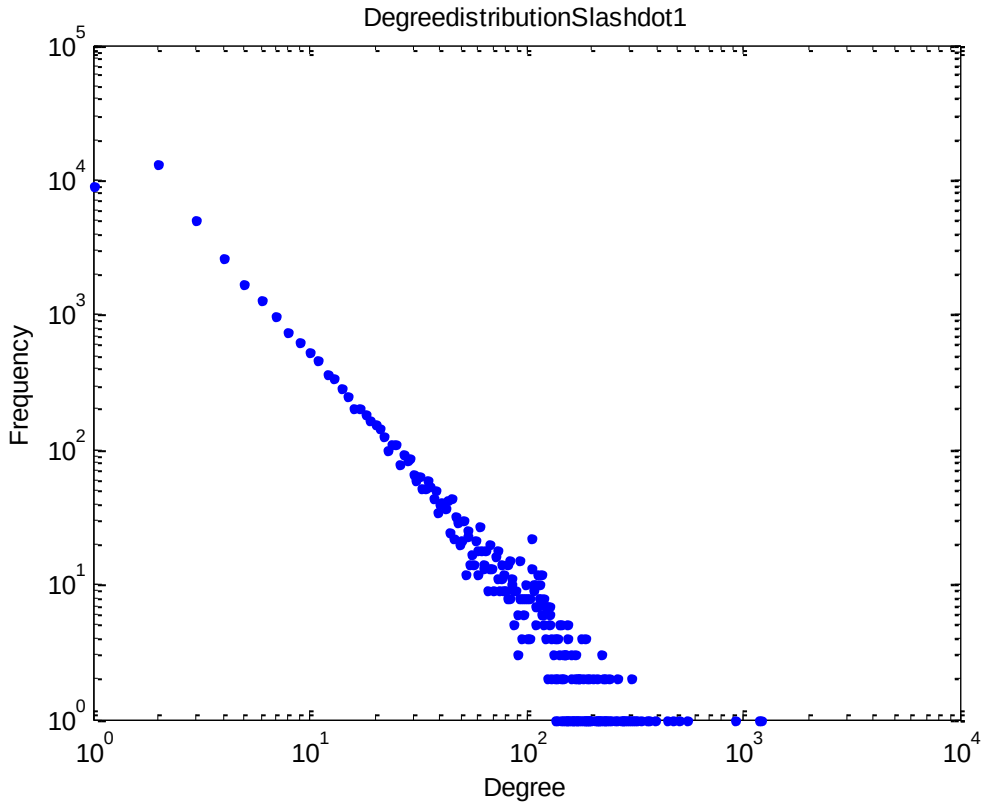

Figure 3.1.2.1 Degree distribution of slashdot1

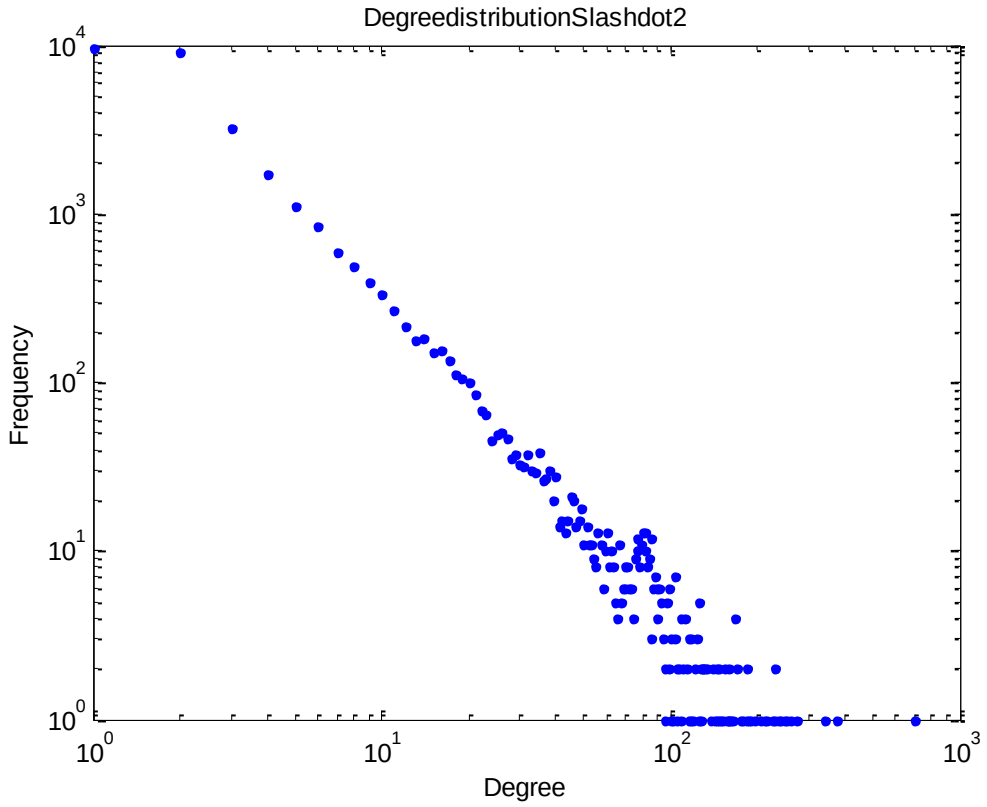

Figure 3.1.2.2 Degree distribution of slashdot2

3.1.3 DBLP

DBLP is a computer science bibliography in which conference papers and other journals authors are published. Nodes are people who have publication in this database and edges are formed if two people have published a paper or a journal together. Unlike the other two data sets this graph is obtained by crawling the DBLP database.
By the time, we did the parsing the database had 986331 nodes and 6872951 edges. Of all the graphs we used this is the largest in size. The graph has all the properties of social networks that are mentioned in previous sections. The following figure shows the degree distribution of this graph.

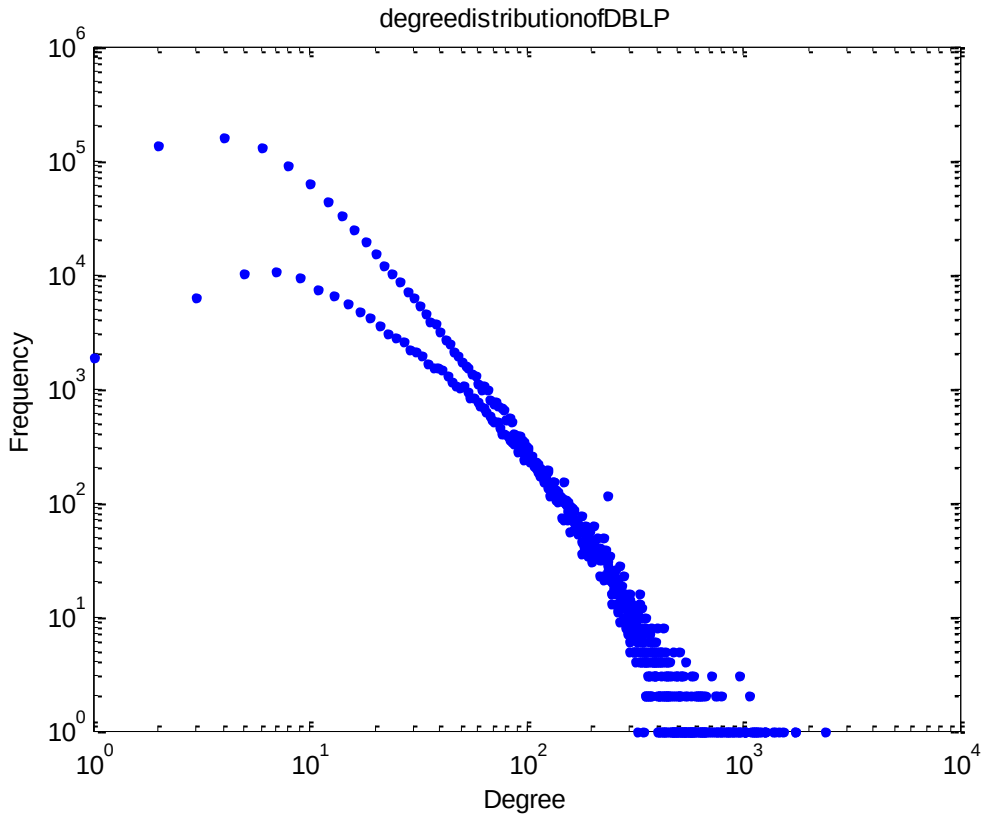

Figure 3.1.3 Degree distribution of DBLP

For parsing we modified the simple crawler given in the official DBLP website. According to the detailed description of the crawler in [5], it is used to find the shortest path between two DBLP authors in the co-author graph .We took this crawler and modified it in a way to gain co -author relationship between all authors .The dblp.xml file, which contains the record of authors, their publication and also other related information, is too costly to download [5].The crawler simplifies the process by using the URL and get all the information directly from the server. A simple API for XML (SAX) parser is used for this purpose. The simple parser also uses a co-author function that returns the co-authors of a user by taking the urlpt of the user. The urlpt is a parameter which contains the mapping between the user and the URL. Using the authors as a node, a graph is constructed by making an edge based on the result obtained from the co-authors function.
Due to the size of this graph, our algorithm was not able to deanonymize the whole graph so we haven't included any result using this graph. May be by using a high processing computers, it could be possible to deanonymize the graph.

3.2 Performance improvement of the propagation algorithm

Our propagation algorithm is completely based on the structure of the graphs. By structure we mean the degree of the nodes and their neighborhood without considering their identity. There are some parameters which played a great role in the algorithm; Similarity score, eccentricity, sorting, and degree limit are the basic ones. Besides, ordering, degree comparison and running the algorithm in two stages are also some of the ways to increase the performance of the algorithm.

Similarity score: is calculated based on cosine similarity between nodes of the sanitized and auxiliary graph. Nodes are added to the map if they are identical, so each pair of mapped nodes are considered as identical. There are two graphs G1 & G2 and a map m, the similarity score between node X of G1 and nodes Y of G2 is calculated as follows.

$$\text{Similarity Score} = \frac{(\textbf{Number of common neighbours of } X \textbf{ and } Y \textbf{ in } m)}{\sqrt{(\textbf{Number of neighbours of } X \textbf{ in } m) * (\textbf{Number of neighbours of } Y \textbf{ in } m)}}$$

The maximum score that can be achieved is 1 and it happens when the two nodes are exactly the same or both have only one neighbor in the map (m).Every time the similarity score was calculated for a node in the auxiliary graph, there was a chance of finding more than one node in the target graph which could be a match; All These nodes will be added to a candidate set(C), and later only one of the nodes will be chosen. Adding every node to the set increases the size of the candidate set which in turn increases the chance of choosing the wrong node from the set .To overcome this problem we limited the score in a way that a node is chosen as a candidate if its score value is at least 0.5. This is the same as saying that the target node should have at least 50% similarity to the auxiliary node.

Eccentricity (theta): is a method stated in [10] and it is used to measure the importance of an item in a set. As we said above, our candidate set (C) contains the candidate nodes which are running for the selection to be a match. By calculating the eccentricity a node with a maximum score can be selected if it stands out from all the nodes in the set.

$$\text{Eccentricity (theta)} = \frac{max(C) - max2(C)}{\sigma(C)}$$

Where: C is the candidate set ; max(C) and Max2(C) are the highest and second highest scores in the set and $\sigma(C)$ is the standard deviation.

The result of this formula is used in a way suitable for the algorithm to propagate. We have used different threshold values aiming to increase the total number of nodes identified and also minimize the execution time of the algorithm. The threshold can also be considered as a filtering mechanism while choosing a match for a node.

Degree limit

This parameter is used to filter out nodes which have degree more than a certain value so the algorithm exclude all the nodes with less than the limit and only consider those fulfilling the

filtering value .The main reason of using this parameter is to minimize the execution time of the algorithm and increase performance.

Degree Comparison

Another parameter to care was choosing a candidate which has a degree closer to the degree of the node in the auxiliary graph. The reason is that while anonymizing most of the time the degree of the nodes is kept almost the same. For example if the anonymized graph is assumed to be changed in way that degree of the nodes is increased by 2, that means the algorithm will use degree comparison 2. It will first calculate the similarity score and then add the node to the candidate set if and only if the node has degree equal or 2 more than the degree of the auxiliary node.

Two stage process

As mentioned in previous chapter, seeds have a great influence on the performance of the algorithm. Rather than just increasing the number of seeds, we run the algorithm in two stages. In the first stage, the algorithm will run with a large value of theta, for example theta greater than 2 and some amount of seeds. The result of this stage will be a collection of nodes deanonymized. The algorithm then takes this result as a seed and a smaller value of theta to start the second stage. The reason for choosing two different values of theta is that with greater theta it is possible to filter more accurate result in the first stage which can be used as seeds; on the other hand with a smaller theta in the second stage, the algorithm becomes more faster. In this way nodes which are ignored in the first stage will get a second chance.

Ordering

In different iterations of our java program, neighbors of a node are returned as a collection and the order of the elements of the collection varies in different iterations. Whenever the program is executed nodes are surfed with a different order and a node which is picked as having the maximum score might not be chosen in another execution. The solution to overcome this problem was ordering the nodes in the list based on integer sorting or string sorting. The reason is that when the nodes are ordered both in the auxiliary and sanitized graphs, they will have the same order in both graphs. For example if node A precedes node B in the auxiliary graph, the corresponding anonymized node of A would also precedes the corresponding anonymized node of B. This happens if anonymization is done after ordering the nodes. The technique is used in some anonymization algorithms and we did one of these algorithms.
The anonymization algorithm works as follows: it takes the collection of the nodes from the graph, order them using string sorting, and then change the identity of each node in a sequence. A ground truth with a match between the old and new identity of the node will be kept for future calculation of performance of the algorithm.

3.3 Propagation algorithm

As the name implies the propagation algorithm starts from a few nodes called seeds in the map and gradually propagate. This process is affected by all the parameters that we mentioned in the above section.
The diagram below shows the procedure of how one node is added to the map. The algorithm starts from a node in the map and searches for neighbors of the node, which are outside the map but in the auxiliary graph. For each iteration the algorithm picks an unmapped node from these neighbors and checks the degree of that node; If the node has degree above the limit, the score for corresponding unmapped nodes in the sanitized graph will be calculated and stored in an array and then continues to calculate the eccentricity and if the value is above the threshold the node with the maximum score will be chosen as a good match but still not yet good enough to be added to the map. Hence the same process will be repeated in the reverse direction; that is from the chosen node in the sanitized graph to the auxiliary graph. This time, if the node with the maximum score is the auxiliary node itself, this pair of nodes will be added to the map.

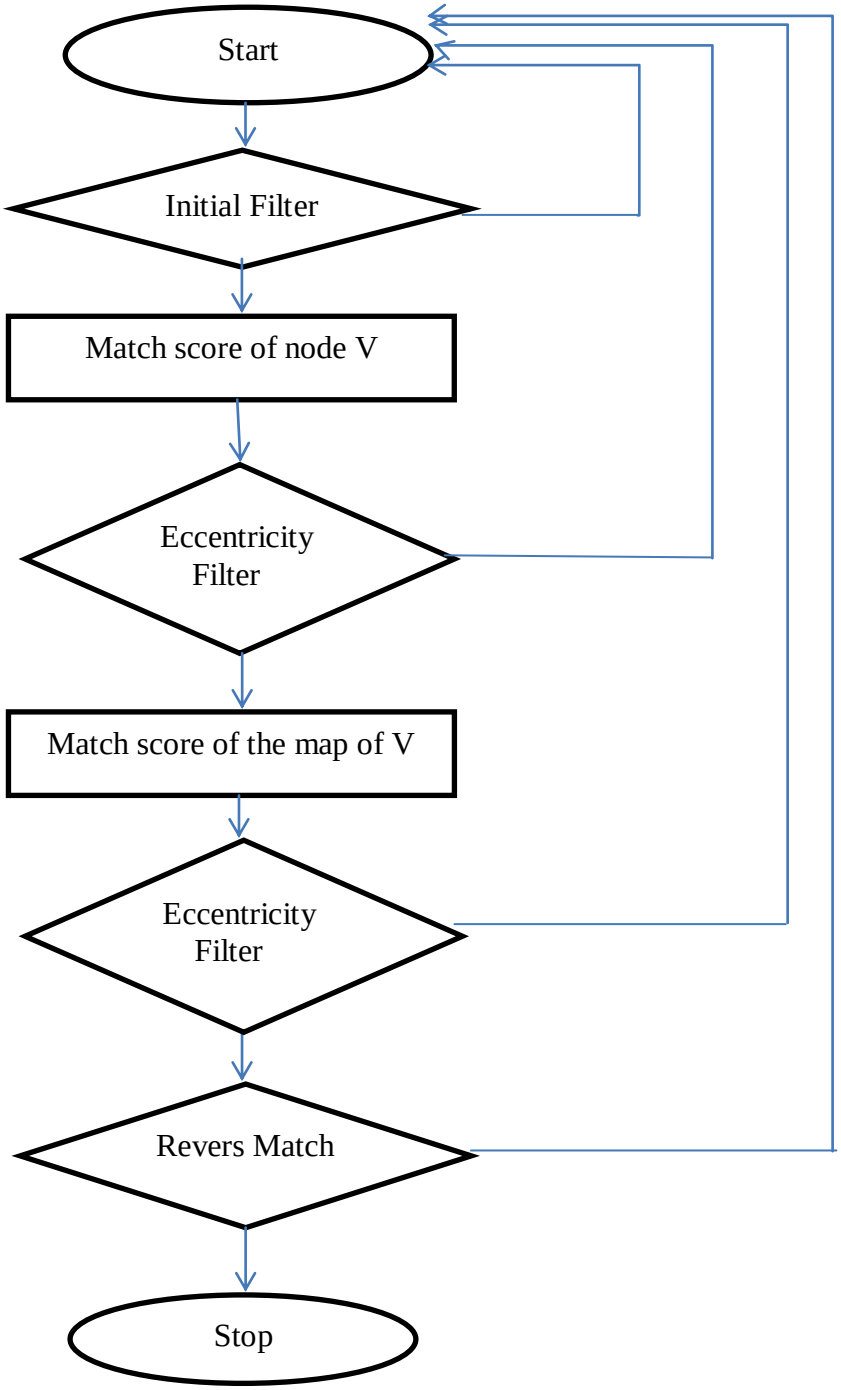

Figure 3.3 procedures for mapping one node

The pseudo code for the whole algorithm is given below. It was originally taken from [1] and it was modified and improved based on the parameters that have been discussed previously.

```
function propagation(auxilarygraph, sanitizedgraph, mapping)

    for mappednode in mapping:
        auxilaryNodes=sort neighbours of mappednodes

         for lnode in auxiliaryNodes  having degree greater than d :
                scores[lnode] = matchScores(auxilarygraph, sanitizedgraph, mapping, lnode)
              if eccentricity(scores[lnode]) < theta: continue
               rnode = (pick node from sanitizedgraphs.nodes where
                scores[lnode][node] = max(scores[lnode]))

        scores[rnode] = matchScores(sanitizegraph, auxilarygraph, invert(mapping), rnode)
                        if eccentricity(scores[rnode]) < theta: continue
                                reverse_match = (pick node from auxilarygraph.nodes where
                        scores[rnode][node] = max(scores[rnode]))
                if reverse_match != lnode:
                continue

                mapping[lnode] = rnode

        function matchScores(auxilarygraph, sanitizedgraph,mapping, lnode)

                initialize scores = [0 for rnode in sanitizedgraph.nodes]

                for (lnbr, lnode) in auxilarygraph.edges:
                        if lnbr not in mapping: continue
                                rnbr = mapping[lnbr]
                for (rnbr, rnode) in sanitizedgraph.edges:
                 if rnode in mapping.image: continue
                scores[rnode] =  SimilarityScore

               return scores
        function eccentricity(items)
                return (max(items) - max2(items)) / std_dev(items)

        until convergence do:
        propagationStep(auxilarygraph, sanitizedgraph, seed_mapping)
```

4 RESULT AND DISCUSSION

4.1 Evaluation of each performance improvement method

Each performance improvement method has its own pros and cons. To see these effects a couple of experiments are performed. For these experimental processes two graphs from the unknown social network dataset are used. The two graphs G1 and G3 have 81831 and 153712 nodes respectively; and 33.75% edge overlap between them. G3 is used as a target graph and G1 as auxiliary graph. We started our experiments with a small number of seeds but that was not enough to propagate the algorithm. By gradually increasing the number of seeds the result started to get better. For the first couple of experimental result that is going to be discussed in this section, 17 seeds were chosen based on their degree value.

Two-stage process

To test how effective is the two stage process, the algorithm will run twice with different theta value and number of seeds in each stage. In the first stage it runs with theta greater than or equal to 1 and outputs a map of deanonymized nodes. The algorithm then takes these mapped nodes as seeds and another theta value greater than or equal to zero. The final result of the two stage process in comparison with one stage process is shown in table 4.1.2. This process increases both the total coverage and the false positive. The bigger false positive result is because of the highly dependence of the parameter with the theta value chosen during the first and the second stages. The result obtained from the first stage also plays a great role in the final value; if the first stage result, has a lot of false positives, the second stage will continue using those false results as seed and start to propagate. As a result, there is a high chance of ending up with a false result. Besides the increase in false positives, this technique also doubles the execution time of the algorithm. Due to these reasons we prefered to search for other parameters rather than trying to make more experiment to overcome the problems.

Degree limit	#of nodes with specified degree limit	Theta	Degree comparison	Two stage	Total #of nodes deanonymized	False +ve	%total	%false +ve
>2	8059	>=1	+2	Without	2738	669	33.9	24.4
>2	8059	>=1 &>=0	+2	with	3323	995	41.2	29.9
>1	15406	>=1	+2	Without	4438	1239	28.8	27.9
>1	15406	>=1 &>=0	+2	with	5960	2042	38.68	34.2

Table 4.1.2 Effect of running the algorithm in two stages

Degree comparison

At first it was hard to get more than few nodes deanonymized, but after we started using the degree comparison parameter, changes were clearly seen. Varying this parameter to get a better output had an effect but it was random as shown in table 4.1.3. From the properties of the two graphs it seemed like when the graph evolved the nodes changed their degree in a random way, however majority of the nodes changed their degree by 2. The parameter highly depended on the property of the graphs and above all it made the algorithm act randomly. Hence degree comparison was not an efficient parameter to evaluate how well the algorithm performed.

Degree limit	#of nodes with specified degree limit	Theta	Degree comparison	Total # of nodes deanonymized	False +ve	%total	%false +ve
>2	8059	>=1	+1	2656	556	32.9	20.9
>2	8059	>=1	+2	2738	669	33.9	24.4
>2	8059	>=1	+3	2698	626	33.4	23.2
>2	8059	>=1	+4	2530	687	31.3	27.4

Table 4.1.3 effect of degree comparison

Ordering

Parameters like degree comparison or running the algorithm in two stages were ways to increase the coverage of the algorithm but after we came up with the idea of ordering those parameters, they couldn't have that much effect rather than increasing the execution time. For the experiment with ordering we took G3, which is target graph, sorted the entire node using string sorting and then anonymized the ordered nodes to hide their identity.
The algorithm ran with different combinations of parameters as shown in table 4.1.4. The main goal of using ordering is to overcome the random results that are obtained from different iteration of the same parameter setup. The table shows that not only a consistent result is achieved but also there is a noticeable improvement in the overall performance. Before ordering the output for the same parameter, the result was sometimes with large coverage and sometimes very small coverage with different numbers of false positives. However after ordering is done, the algorithm keeps deanonymizing large portion of the graph with average false positives.

# seeds	Degree limit	# nodes with specified degree limit	Theta	Total # nodes deanonymized	False positive	%total	%false positive
17	>2	8059	>=1	1886	97	23.4	5.14
17	>1	15406	>=1	5514	467	35.7	8.4
17	>0	45228	>=1	29097	9181	64.3	31.5

Table 4.1.4 Overall Effect of ordering the graph before anonymization is done

Seeds

One way of getting a high coverage was by increasing the number of seeds. We used 17 seeds which were chosen based on their degree values. Even though having a large number of seeds brings some change on the final result, our algorithm had to work efficiently with a very limited input scenario so we decided to focus on how important those seeds are than just the quantity. We knew that the degree and neighborhood structure of seeds were important. The experiment was done once with 8 seeds and the other with 17 seeds. In the table below it is shown that there is a slight difference in the total coverage and also the false positive. When the amount of seeds was almost double we did not get a better result. The main cause for this is, in the 17-seed case the connectivity between the nodes was very weak. This might in turn affect the similarity score calculations which influenced the final result.

# seeds	Degree limit	#nodes with specific degree limit	Theta	Total # of nodes deanonymized	False +ve	%total	%false positive
17	>0	45228	>=1	29097	9181	64.3	31.5
8	>0	45228	>=1	29218	11031	64. 6	37.7

Table 4.1.5 Difference between using 17 and 8 seeds

Similarity score

Putting a limit on similarity score is one way to decrease the execution time of the algorithm and so to improve the performance. The algorithm used had a large execution time because for each node, there was a big candidate set to be chosen as the map of the node. By putting some limit on the similarity score, the size of the candidate set was decreased. For our experiment we chose similarity score limit value to be at least 0.5.This means a node was a candidate, if it had at least 50% similarity with the auxiliary node. Besides the execution time, limiting the similarity score also affected the false positive as shown in Table below.

#seeds	Degree limit	#nodes with specific degree limit	Theta	Score limit	Total # of nodes deanonymized	False +ve	%total	%false+ ve
8	>0	45228	>=1	None	29218	11031	64. 6	37.7
8	>0	45228	>=1	>=0.5	29210	10181	64. 58	34.85

Table 4.1.6 limiting similarity score

Degree limit

This experiment was done by different degree values to filter the nodes which are going to be deanonymized. For each result obtained after using a certain value of degree limit, the coverage was calculated by considering only the number of nodes which had a degree above the limit. With no degree limit the coverage was 65.4% but high false positive result compared to the ones with degree limit. For example filtering the nodes with degree greater than 1 resulted in 38.3 % coverage with only 11.1% false positive; based on this for 65.4%

coverage we should have obtained a false positive result of around 18.95%. Sometimes high coverage by itself doesn't show the effectiveness of the algorithm, so false positive results should be considered as well. Hence we can conclude that limiting the degree value had a better effect on performance. The other advantage of this parameter is minimizing the execution time. As shown in the table below for every degree limit value the algorithm excluded all nodes which had degree lower than that value. Even though the table doesn't include the time it takes to execute one iteration, it was observed that there was a difference in execution times between each result. One obvious reason behind is that the algorithm focused only on those nodes that were filtered, which was a less figure compared to the whole graph. For example with degree limit 1, the number of nodes filtered was only 34% of the total number of nodes in the graph and with degree limit 2 it was 17.8% from the total.
Generally this parameter had a positive effect on the overall performance of the algorithm together with ordering and theta.

# seeds	Degree limit	# nodes with specified degree limit	Theta	Total # nodes deanonymized	False positive	%total	%false positive
8	>0	45228	>=0	29592	10333	65.4	34.9
8	>1	15406	>=0	5909	656	38.3	11.1
8	>2	8059	>=0	2368	97	29.3	4.0
8	>3	4837	>=0	729	31	15.0	4.2

Table 4.1.7 degree limit

Theta (eccentricity)

Theta is another filtering parameter that was used to choose a node which stood out from a candidate set. Experimental work with theta value varying from 0 to 2 was performed and the result is shown in table 4.1.8.With higher theta value it was possible to minimize the execution time; On the other hand with theta value 0, it was possible to cover a larger portion of the graph with average false positives. From most of our experimental results, we can conclude that theta is a useful parameter when it is used together with degree limit, score limit and ordering.

# seeds	Degree limit	# nodes with specified degree limit	Theta	Total # nodes deanonymized	False positive	%total	%false positive
8	>1	15406	>=2	1919	230	12.4	11.9
8	>1	15406	>=1	5114	601	33.2	11.7
8	>1	15406	>=0.5	5815	646	37.7	11.1
8	>1	15406	>=0	5909	656	38.3	11.1

Table 4.1.8 theta

4.2 Results of the algorithm on different graphs

The goal of our work was to write an algorithm which has high coverage, low error and small execution time. However making it effective on a single type of dataset is not sufficient to evaluate its performance. Hence it was tested on some other graphs with different edge and node overlap as well as on graphs of different datasets. For this reason two datasets and around five different graphs were used .The same experiment was performed by taking one graph as auxiliary one other as target.

After evaluating the importance of each parameter for the performance of the algorithm, only some of them were chosen for future experiments. Hence Ordering, Degree limit, Similarity score limit and Eccentricity or theta were the only parameters used for the next experiments.

4.2.1 The result of using G1 to deanonymize G3

G1 and G3 are from the same social network, but G3 evolves from G1 and they have 53.23% node overlap and 33.75% edge overlap. G1 was used as an auxiliary graph to deanonymize G3, which is the target. Since these two graphs were used in the experiments explained in previous section, only the overall result is given in the table below.

Degree limit	#nodes with specific degree limit	Theta	Total # of nodes deanonymized	False +ve	%total	%false +ve
>0	45228	>=1	29210	10181	64.5	34.8
>0	45228	>=0.5	29416	10220	65.0	34.7
>0	45228	>=0	29592	10333	65.4	34.9
>1	15406	>=2	1919	230	12.4	11.9
>1	15406	>=1	5114	601	33.2	11.7
>1	15406	>=0.5	5815	646	37.7	11.1
>1	15406	>=0	5909	656	38.3	11.1
>2	8059	>=2	385	29	4.7	7.5
>2	8059	>=1	2148	87	26.6	4.0
>2	8059	>=0.5	2285	102	28.35	4.5
>2	8059	>=0	2368	97	29.3	4.0

Table 4.2.1 the result of using G1 to deanonymize G3

4.2.2 The result of using G2 to deanonymize G3

G2 and G3 are also graphs of the same social network. The only difference between the graphs G1 and G2 is that G2 has 40% node overlap and 57.1% edge overlap with G3. The algorithm was run to deanonymize G3 by using G2 as auxiliary graph and different parameter setup. The following table contains the overall result and a comparison of the results is done on later sub sections. One thing to notice is that unlike G1, G2 has more number of nodes with degree above 3 so it was possible to use the degree limit parameter with higher values.

Degree limit	#nodes with specific degree limit	Theta	Total # of nodes deanonymized	False +ve	%total	%false +ve
>0	61586	>=1	43799	8567	71.1	19.5
>1	29120	>=2	13889	288	47.6	2.07
>1	29120	>=1	21127	790	72.5	3.7
>1	29120	>=0.5	21817	741	74.9	3.3
>1	29120	>=0	21876	747	75.1	3.4

>2	20005	>=2	6775	287	33.8	4.2
>2	20005	>=1	13395	462	66.9	3.4
>2	20005	>=0.5	14303	449	71.4	3.1
>2	20005	>=0	14255	442	71.2	3.1
>3	14167	>=2	4208	141	29.7	3.3
>3	14167	>=1	8487	451	59.9	5.3
>3	14167	>=0.5	9258	427	65.3	4.6
>3	14167	>=0	9313	434	65.7	4.6
>4	10815	>=0	6540	336	60.4	5.1
>5	8242	>=0	3417	170	41.4	4.9
>6	6291	>=0	1800	73	28.6	4.0
>7	5246	>=0	990	23	18.8	2.3
>8	4497	>=0	643	16	14.2	2.4
>9	1907	>=0	201	3	10.5	1.4

Table 4.2.2 the result of using G2 to deanonymize G3

4.2.3. Graphs from Slashdot social network

A brief description about Slashdot social network was given in section 3.1.2. Similar experiment was done on target graph Slashdot1 by using slashdot2 as auxiliary and the result is presented below. As mentioned previously, these two graphs have 57.45% node overlap and 36.6% edge overlap.

Degree limit	#nodes with specific degree limit	Theta	Total # of nodes deanonymized	False +ve	%total	%false +ve
>2	11787	>=1	10377	299	88.0	2.88
>2	11787	>=2	22	10	0.19	45.50
>3	8558	>=0	8417	248	98.35	2.95
>3	8558	>=1	8124	162	94.93	1.99
>3	8558	>=2	22	10	0.25	45.50
>4	6845	>=0	6796	148	99.28	2.17
>4	6845	>=1	6290	4	91.89	0.64
>4	6845	>=2	6294	4	91.95	0.63
>5	5730	>=0	5707	119	99.59	2.08
>5	5730	>=1	5485	4	95.70	0.07
>5	5730	>=2	5489	4	95.79	0.07
>6	4880	>=2	4740	4	97.13	8.00
>7	4291	>=2	4225	4	98.40	0.09
>8	3798	>=2	3759	4	98.90	0.10
>9	3405	>=2	3376	1	99.10	0.29

>10	3074	>=1	3065	42	99.70	0.13
>10	3074	>=2	3052	1	99.20	0.32

Table 4.2.3 result from Slashdot social network

4.3 Comparison and Discussion

The common property that was observed in both types of datasets is that the coverage increases as the eccentricity (theta) value decreases as shown in the figure 4.3.1. The maximum coverage for a specific degree limit is achieved while using eccentricity value zero. The negative effects of using theta value zero are longer execution time of the algorithm and sometimes increased number of false positives. This explanation might be applicable depending on the density of the graph. For example if we take Slashdot data set, with theta value zero it was possible to deanonymize 99.5% of the nodes with only 2.08% false positives. For G1 and G3 when theta value is zero the maximum coverage was 65.4% and the false positive result is around 34.9%. Since the Slashdot graph has high connectivity, the coverage can be more when theta value more than zero is used together with degree limit value.

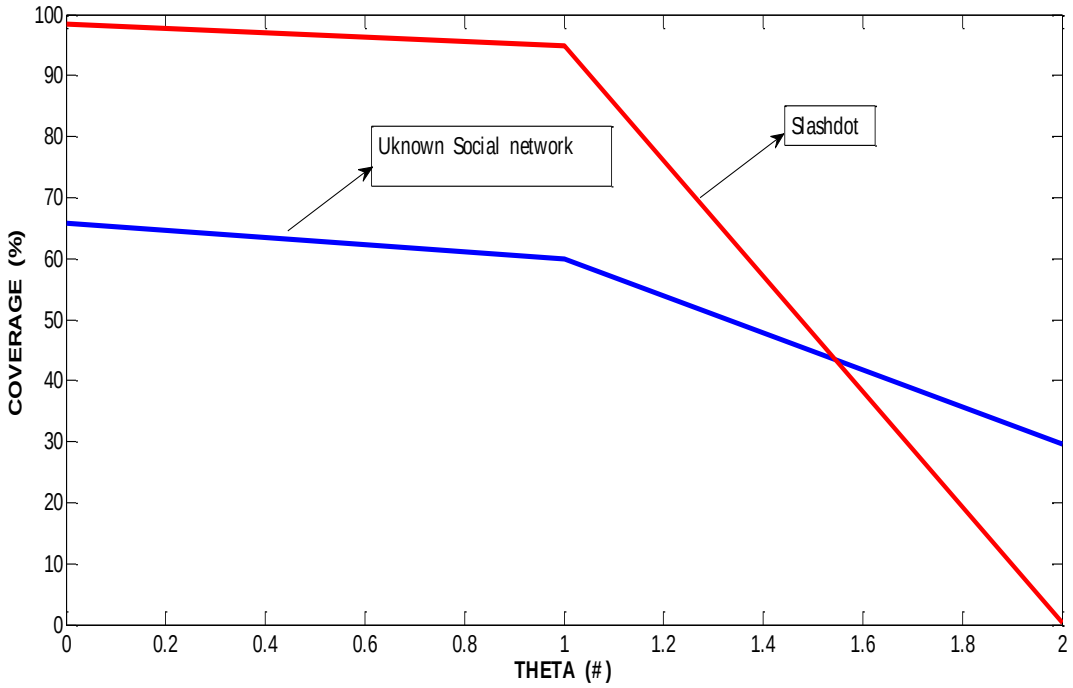

Figure 4.3.1 Effect of eccentricity (theta) on different datasets

One special property of the Slashdot graph is its density. Compared to the other social networks, in this graph the average degree of a node varies from 3 to 5. This has a positive effect on the performance of the algorithm on the graph. As shown in table 4.2.3, not only it was possible to use large values for degree limit, but also a high percentage of coverage with less error was obtained**.** On the contrary, the unknown social network has totally different reaction towards the increase of degree limit value. As it is shown in the figure 4.3.2**,** coverage decreases as degree limit increases. The main reason for this is that in these graphs more than 40% of the nodes have degree one and they are connected to higher degree nodes.

Hence whenever we use certain degree limit we are excluding that many nodes from the graph which in turn makes the graph less connected than it was before. The algorithm then will face difficulty in getting a node which fulfills the minimum requirement of similarity score.

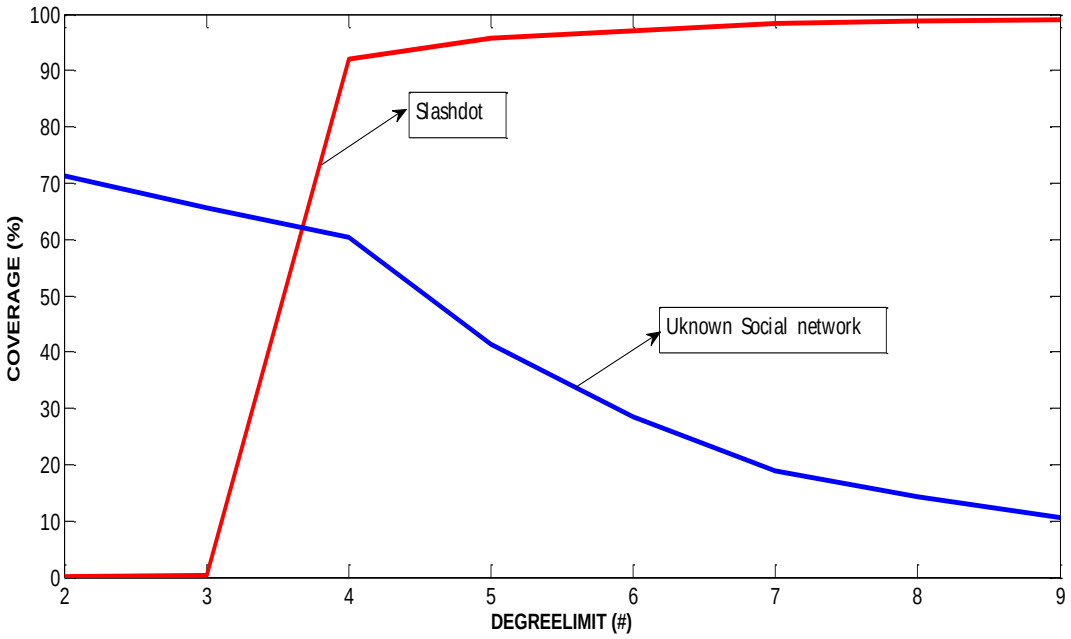

Figure 4.3.2 Effect of degree limit on different datasets

On the other hand it was clearly shown that G3 was deanonymized better when G2 is used as auxiliary graph because G2 has greater edge overlap with G3 than G1 does. Basically G1 has more number of common nodes with G3 but most of them are with degree less than 2; this makes the algorithm dies out on early stage and also increases the false positive. Even though G1 and G2 has the above mentioned differences, their overall property is similar. Due to this reason both degree limit and theta have the same effect on each case. From figures 4.3.3 and 4.3.4 we can see that the coverage in both cases increases when minimum theta and degree limit is used.

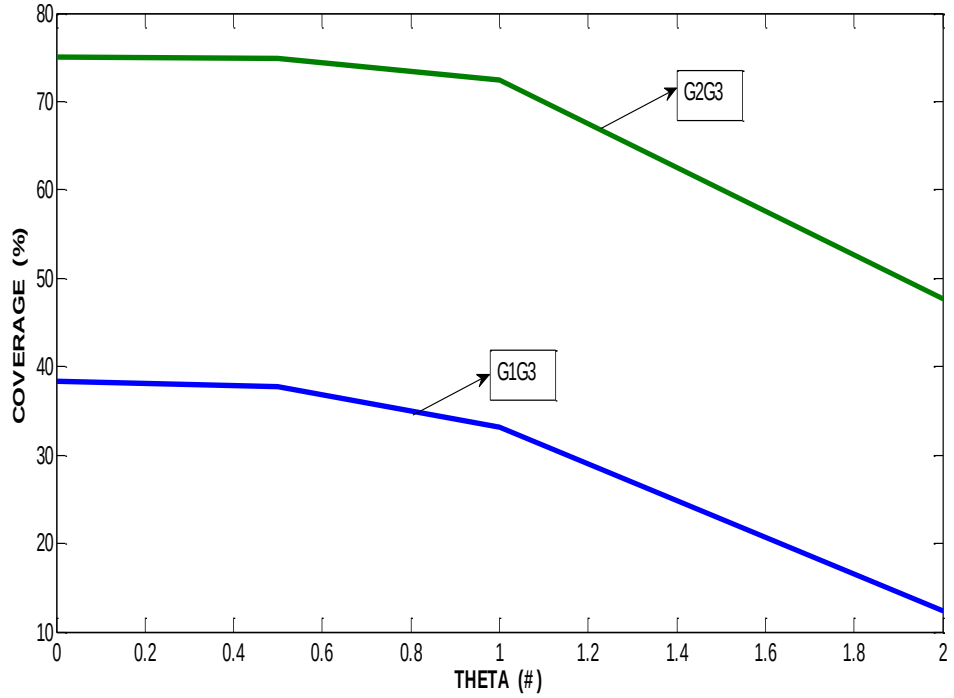

Figure 4.3.3 Comparison of effect of theta on G1G3 and G2G3

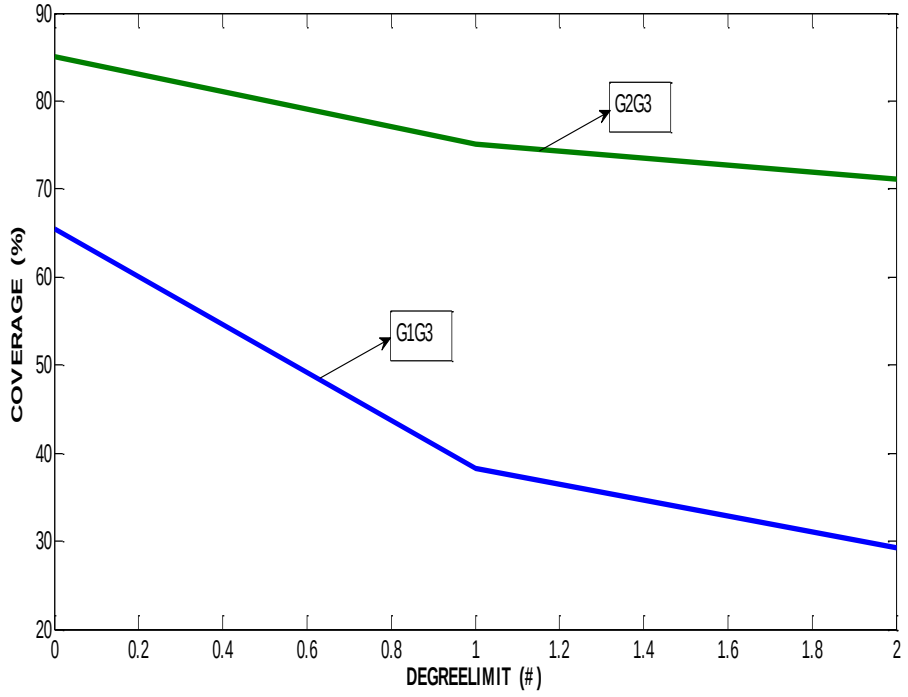

Figure 4.3.4 Comparison of effect of degree limit on G1G3 and G2G3

5 CONCLUSION

As mentioned in previous sections, the main purpose of the thesis work was not only to make deanonymization algorithm but also to find strong parameters to improve the performance of such algorithm. So with the proposed parameters the algorithm we developed was getting closer to perform and give a result with high coverage, low error and small execution time. In chapter 5 we evaluated each performance improvement parameters, based on that degree comparison and two stage processes has an inconsistent result and negative effect toward the performance of the algorithm by doubling the execution time. On the other hand the algorithm performs best when we use parameters like ordering, degree limit, score limit, eccentricity (theta) and high degree seeds. Ordering was one of the very important ways to obtain high coverage and consistency. Eccentricity, limiting similarity score and degree limit makes the algorithm's execution time very short by filtering out only couple of nodes at different iteration. Generally, by using the appropriate values for each parameter a high coverage with minimized false positive is achieved. From the discussion in the above chapter we can also conclude that the more the graph is dense and the more the overlap between the two graphs is increased, the more the algorithm covers large portion of the graph.
Finally, It can be obtained from this experiment that more attention is needed to be considered about the privacy of the social networks members. The algorithm presented, shows the possibility of security breaches in these networks and so more attention and research in order to develop better anonymization algorithm seems to be required.

6 FUTURE WORKS

Future Work can be categorized in three different parts:

- Trying to do the experiments on larger social networks and compare the result with our result. As sorting was a good improvement in performance for us, it would be good to try it for other social networks as well.
- Improving anonymization algorithms by making them in a way to overcome the parameters which increase the performance of deanonymization algorithms.
- More studies can be done about special characteristics of social networks and how they change in order to improve the algorithm.

REFERENCES

[1] Arvind Narayanan, Vitaly Shmatikov: De-anonymizing Social Networks CoRR abs/0903.3276 (2009)

[2] Alan Mislove, Massimiliano Marcon, P. Krishna Gummadi, Peter Druschel, Bobby Bhattacharjee: Measurement and analysis of online social networks. Internet Measurement Comference 2007:29-42

[3] A. Felt and D. Evans: Privacy protection for social networking APIs. In W2SP, 2008.

[4] James Cheng, Ada Wai-Chee Fu, Jia Liu: K-isomorphism: privacy preserving network publication against structural attacks. SIGMOD 2010:459-470

[5] Michael Ley: DBLP XML Requests Appendix to the paper "DBLP — Some Lessons Learned 2009

[6] Medical News Today. WellNet launches online social networking program for health care coordination. http://ww1w.medicalnewstoday.com/articles/118628.php, 2008.

[7] Z. Stone, T. Zickler, and T. Darrell: Auto tagging Facebook: Social network context improves photo annotation. In *Workshop on Internet Vision*, 2008.

[8] L. Backstrom, C. Dwork, and J. Kleinberg: Whereforeart thou R3579X? :Anonymized social networks, hidden patterns, and structural steganography. In *WWW*,2007.

[9] http://snap.stanford.edu/data/

[10] A. Narayanan and V. Shmatikov : Robust deanonymization of large sparse datasets. In S&P, 2008.

[11] B. Popescu, B. Crispo, and A. Tanenbaum: Safe and private data sharing with Turtle: Friends team-up and beat the system. In *Cambridge Workshop on Security Protocols*, 2004.

[12] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan: Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54, 2006.

[13] A. Campan and T. M. Truta: A clustering approach for data and structural anonymity in social networks. In *PinKDD*, 2008.

[14] J. Cheng, Y. Ke, A. W.-C. Fu, J. X. Yu, and L. Zhu: Finding maximal cliques in massive networks by h*-graph. In *To appear in SIGMOD*, 2010.

[15] M. Hay, G. Miklau, D. Jensen, D. F. Towsley, and P. Weis: Resisting structural re-identification in anonymized social networks. *PVLDB*,1(1):102–114, 2008.

[16] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava: Anonymizing social networks. *Technical report No. 07-19, ComputerScience Department, University of Massachusetts Amherst*, 2007.

[17] R. Kumar, J. Novak, and A. Tomkins: Structure and evolution of online social networks. In *KDD*, pages 611–617, 2006.

[18] K. Liu and E. Terzi: Towards identity anonymization on graphs. In *SIGMOD conference*, pages 93–106, 2008.

[19] L. Zou, L. Chen, and M. T. Ozsu. K-automorphism: A general framework for privacy preserving network publication. In *VLDB*,2009.

[20] L. Adamic, E. Adar: How to search a social network. Social Networks 27(2005).

[21] R. Agrawal, R. Srikant: Privacy-preserving data mining. Proc. SIGMOD, 2000.

[22] I. Dinur, K. Nissim: Revealing information while preserving privacy.Proc. PODC,2003.

[23] A Narayanan, V. Shmatikov : How to Break Anonymity of the Netflix Prize Dataset. arxiv cs/0610105, Oct. 2006.

[24] M. Chew, D. Balfanz, and B. Laurie :(Under) mining privacy in social networks. In *W2SP*, 2008.

[25] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri: Feedback effects between similarity and social influence in online communities. In *KDD*,2008.
[26] A. Felt and D. Evans: Privacy protection for social networking APIs. In *W2SP*, 2008.
[27] B. Fitzpatrick and D. Recordon: Thoughts on the social graph. http://bradfitz.com/social-graph-problem/,2007.
[28] R. Gross, A. Acquisti, and H. Heinz: Information revelation and privacy in online social networks. In *WPES*, 2005.
[29] S. Guha, K. Tang, and P. Francis. NOYB: Privacy in online social networks. In *WOSN*, 2008.
[30] G. Kossinets, J. Kleinberg, and D. Watts: The structure of information pathways in a social communication network. In *KDD*, 2008.
[31] F. Kerschbaum and A. Schaad: Privacy-preserving social network analysis for criminal investigations. In *WPES*, 2008.
[32] B. Krishnamurthy and C. Willis: Characterizing privacy in online social networks. In *WOSN*, 2008.
[33] M. Lucas and N. Borisov. flyByNight: Mitigating the privacy risks of social networking. In *WPES*, 2008.
[34] D. Liben-Nowell and J. Kleinberg: The link prediction problem for social networks. In *CIKM*, 2003.
[35] M. Richardson and P. Domingos: Mining knowledgesharing html, 2008.