Anonymizing Subsets of Social Networks

by

Jared Glen Gaertner
B.Sc., University of Victoria, 2010
B.Eng., University of Victoria, 2010

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Anonymizing Subsets of Social Networks

by

Jared Glen Gaertner
B.Sc., University of Victoria, 2010
B.Eng., University of Victoria, 2010

Supervisory Committee

Dr. Ulrike Stege, Co-Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Co-Supervisor
(Department of Computer Science)

Supervisory Committee

Dr. Ulrike Stege, Co-Supervisor
(Department of Computer Science)

Dr. Venkatesh Srinivasan, Co-Supervisor
(Department of Computer Science)

ABSTRACT

In recent years, concerns of privacy have become more prominent for social networks. Anonymizing a graph meaningfully is a challenging problem, as the original graph properties must be preserved as well as possible. We introduce a generalization of the degree anonymization problem posed by Liu and Terzi. In this problem, our goal is to anonymize a given subset of vertices in a graph while adding the fewest possible number of edges. We examine different approaches to solving the problem, one of which finds a degree-constrained subgraph to determine which edges to add within the given subset and another that uses a greedy approach that is not optimal, but is more efficient in space and time. The main contribution of this thesis is an efficient algorithm for this problem by exploring its connection with the degree-constrained subgraph problem. Our experimental results show that our algorithms perform very well on many instances of social network data.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

my girlfriend (Laticia), for bringing out the best in me.

my siblings (Tracy, Bryan, and Jason), for showing me the way professionally and personally.

my brother-in-law (Gwynn), for sparking my interest in academia.

my supervisors (Ulrike Stege and Venkatesh Srinivasan), for the opportunity, funding, and counterbalance of enthusiasm and methodicalness without which I would not have been able to complete this.

If you really want to be a first-class scientist you need to know yourself, your weaknesses, your strengths, and your bad faults, like my egotism. How can you convert a fault to an asset? How can you convert a situation where you haven't got enough manpower to move into a direction when that's exactly what you need to do? I say again that I have seen, as I studied the history, the successful scientist changed the viewpoint and what was a defect became an asset.

Richard Wesley Hamming

DEDICATION

To my parents, for their unending support in everything I do.

Chapter 1

Introduction

In this chapter, we begin by introducing background information related to the main problem of this thesis. Then, in Section 1.1, we describe the contributions of this thesis. Lastly, in Section 1.2, we discuss the related work relevant to this thesis.

The idea of informational privacy is a relatively new concept. This can be seen by the lack of a direct translation between many different languages [2]. It is not until the late 1500s that the word "privacy" even entered the English language [18]; in the late 1800s public discussions began taking place in North America [20]. Privacy law has closely followed the development of new technologies, such as the printing press or the Internet, which have brought forth a much larger scope of information being distributed [20]. These new technologies increase the chances of spreading sensitive data about individuals.

In the past, wax to seal an envelope or a key to lock a drawer was sufficient for hiding sensitive information. However, with the advent of electronics and computers, the speed of accessing information has increased, and with it the need to hide electronic data. As the speed of accessing information increases, so does the need to hide the electronic data in a more efficient manner. As well, the increased use of electronic data containing personal information, such as medical data and on social networking sites, brings greater concerns for privacy since the data are much more prevalent. These concerns are exacerbated by the need to have data available for public research to further scientific progress.

In order to alleviate concerns, a way to allow the data to be used within research is required without the possibility of determining the identity of the individual. Without a solution, these data cannot be released to the public, creating a deficiency in data essential for research. One way of solving such a problem for tabular data, given

First	Last	Age	Town	Gender	Married
John	Smith	42	Kelowna	M	Yes
Jack	Smith	15	Kelowna	M	No
Mary	Smith	41	Kelowna	F	Yes
Mary	Johnson	23	Burnaby	F	No
Jessica	Williams	37	Burnaby	F	Yes
Jim	Williams	37	Burnaby	M	Yes

Table 1.1: A dataset.

an integer k where $2 \leq k \leq n$, is by making the dataset *k-anonymous*. In a k-anonymous dataset, the identity of each individual is hidden, as the row containing the information of the individual is identical to at least $(k-1)$ other rows [21]. This is achieved by suppressing entries, which is realised by setting entries to $*$.

Given an integer $e \geq 0$, the problem of making a table k-anonymous in e or less entry suppressions is generally referred to as Data Anonymization. Formally, the Data Anonymization problem is defined as follows: given a dataset D and integers k and e, can D be made k-anonymous in at most e entry suppressions? This problem has been shown to be NP-hard [16].[1]

Solving the problem of Data Anonymization allows, with certainty, for sensitive data to not be linked to the identity of an individual with that dataset alone. This certainty allows the dataset to be provided to the public for research, which would otherwise be unavailable. There is the possibility of discovering sensitive information with auxiliary information, but this is unavoidable [9]. However, with certain alterations to a released dataset, it can be shown that whether a person is included or excluded in the released dataset is inconsequential in sensitive data about the person being discovered [9].

Table 1.1 is a short example of tabular data, which is under consideration to be released for public research. Taking the information from Table 1.1, Table 1.2 is the same table with 2-anonymous data, containing 18 entry suppressions. Note that the top two rows are identical, the middle two rows are identical, and the bottom two rows are identical.

The solution in Table 1.2 is not unique. Table 1.3 is another version of 2-anonymous data of the same dataset (which is also 3-anonymous data). However,

[1]Such an NP-hardness result implies that this problem is highly unlikely to have an efficient (polynomial-time) solution [12].

First	Last	Age	Town	Gender	Married
*	Smith	*	Kelowna	M	*
*	Smith	*	Kelowna	M	*
Mary	*	*	*	F	*
Mary	*	*	*	F	*
*	Williams	37	Burnaby	*	Yes
*	Williams	37	Burnaby	*	Yes

Table 1.2: A dataset that is 2-anonymous with the minimum number of entry suppressions.

First	Last	Age	Town	Gender	Married
*	Smith	*	Kelowna	*	*
*	Smith	*	Kelowna	*	*
*	Smith	*	Kelowna	*	*
*	*	*	Burnaby	*	*
*	*	*	Burnaby	*	*
*	*	*	Burnaby	*	*

Table 1.3: A dataset that is 2-anonymous.

it is not made 2-anonymous in the minimum number of entry suppressions, as it was the case with Table 1.2, and therefore it reveals less information. If we did require the table to be 3-anonymous, Table 1.3 would be the minimum solution.

One naive way to make D k-anonymous is simply to suppress every entry. However, while this solves the problem of making a dataset D k-anonymous, it solves little else, as the data is now unusable. Instead, solving the problem of Data Anonymization relies on the fact that the number of entry suppressions is minimum, in order to keep as much information as possible.

The Data Anonymization problem can be viewed in two possible ways, a *decision* version and an *optimization* version of the problem. In the decision version of the problem, the goal is to determine whether a given dataset D can be made k-anonymous in at most e entry suppressions, with the answer being either yes or no. In the optimization version of the problem, the goal is to make a given dataset D k-anonymous in the minimum number of entry suppressions. It is easy to check that the optimization version of the problem can be used to answer the decision version of the problem, and vice versa, which implies both versions of the problem are equivalent. For example, given a dataset D and an integer k, if we attempt to make D k-anonymous and find the minimum number of entry suppressions is j, we can answer the decision version of the Data Anonymization problem as yes if $j \leq e$, and no otherwise. Similarly, if we can solve the decision version of the Data Anonymization problem for a dataset D and integers k and e, if the answer is yes, we can continually decrement e by one until the answer is no, at which point we know the minimum number of entry suppressions. If the answer was no initially, we continually increment e by one until the answer is yes, at which point we know the minimum number of entry suppressions.

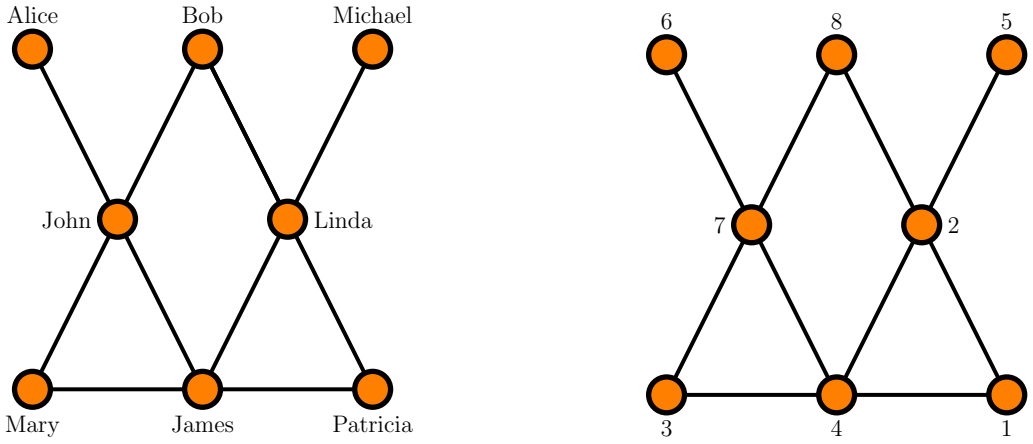
While the use of tabular data has historically been more prevalent, as relational databases are the most common form of databases and use a tabular structure, the use of graphs, defined in Section 2.1, to represent data is becoming increasingly popular, especially due to the advent of social networking. The representation of social network data as a graph is considered by many a more intuitive notion than representing social network data as tabular data. This is due to the many interconnections of "friends" on social networks, where the users are the entities (or more formally, vertices) and the friendships directly correlate to links (or more formally, edges). This is one such interpretation of a graph, but due to their versatility graphs can be structured based on different types of information, all of which can yield useful research data depending

on the situation. An example of this would be email communications between two or more people, where each person is represented by individual vertices and edges could be formed when 1) an email is sent between two people or 2) when five emails are sent between two people. Both would yield interesting, yet completely different, graphs. Another example of a graph is where each city is represented by a vertex and an edge connects any cities that have a direct flight between them. This could also be made more specific by making a graph for a specific airline only, for different times of the day, etc.

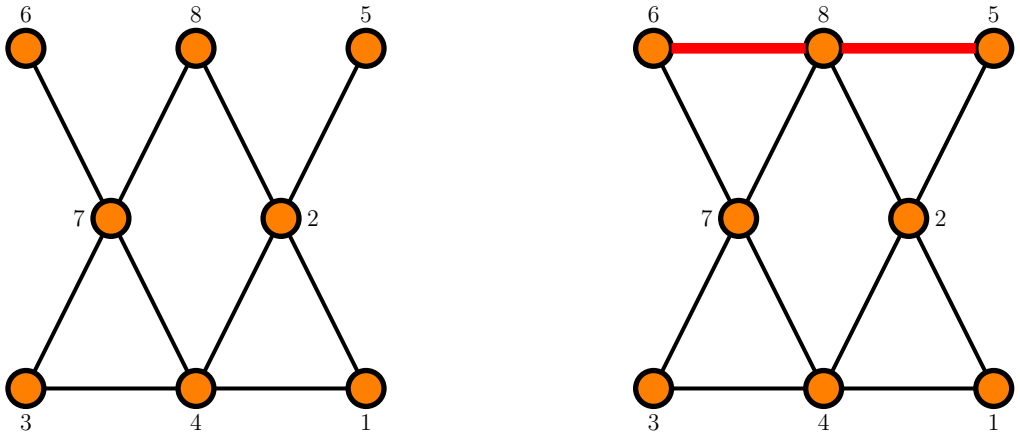
Similar to the problem of making tabular data k-anonymous, one can ask to make a graph anonymous. This can mean one of many things, as a graph reveals data in more ways than tabular data; not only can the vertices hold information, the structure of the graph itself can reveal information. What it means for a graph to be anonymous depends on the knowledge of the adversary and the context of the data. Several assumptions have been introduced in literature (see Section 1.2 for a review), a natural of which is that the degree of a vertex is possibly known. Given this assumption, one way of making a graph anonymous is by making the graph *k-degree anonymous*; the identity of the individual is hidden. A graph is k-degree anonymous, if, given any vertex in the graph, its degree is the same as the degree of at least $(k-1)$ other vertices [14]. This is realised by adding edges to the graph.

Figure 1.1a is a small example, which shows graph data (left) that have been naively anonymized (right) by simply removing the name of the individual. This has been shown to be insufficient to stopping adversaries with knowledge of the degrees of the vertices [3]. Figure 1.1b contains the naively anonymized graph (left) that is made 4-degree anonymous (right). The graph on the right is 4-degree anonymous because there are four vertices of degree 2 (vertices 1, 3, 5, and 6), four vertices of degree 4 (vertices 2, 4, 7, and 8), and no vertices of any other degree. It can be shown that this graph contains the minimum number of edge additions needed, though more can be added and it could still be 4-degree anonymous. In general, observe that any graph can be made k-degree anonymous, as we can add all possible edges, making a complete graph. However, such a graph would not reveal any usable structural data.

The problem of making a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ k-degree anonymous in e or less edge additions is generally referred to as the problem of Graph Anonymization [14]. Formally, Graph Anonymization is defined as follows: given a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ and integers k and e, can G be made k-degree anonymous with at most e edge additions? Similar to the problem of Data Anonymization, and for similar reasons,



(a) Graph data that have been naively anonymized



(b) Graph data that have been made 4-degree anonymous with a minimum number of edge additions. The bold edges are edges that have been added to make the graph 4-degree anonymous.

Figure 1.1: Example of anonymizing a graph.

solving the problem of Graph Anonymization relies on adding a minimum number of edge additions.

As another example, in British Columbia (BC), Canada, utilities are provided by BC Hydro. This year, an initiative of BC Hydro is to outfit all homes with smart meters that would permit adjusting home owner's utility costs in closer correspondence with peak power consumption periods [19]. Such an initiative could additionally provide BC Hydro with an extensive database of microdata on home owner power consumption levels.

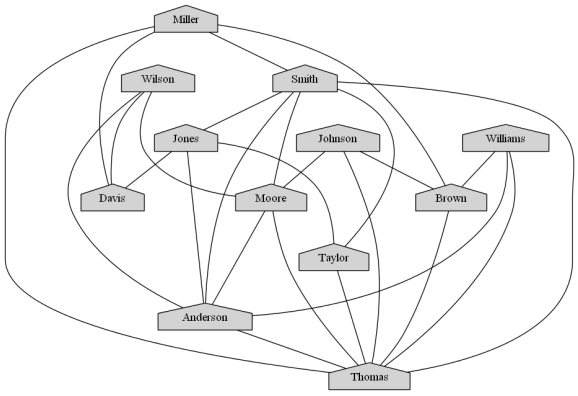
Figure 1.2a is an example of a graph G that could be obtained from BC Hydro data, where each vertex is a household, the names on each vertex are the last names

of the households, and an edge is included between households that are friends. Note that the edges could represent other relationships, such as being within a certain proximity of another household, but we consider the direct social relationship instead. Figure 1.2b is the same graph made naively anonymous, where the numbers on the vertices are arbitrary values. Looking at this graph, it becomes apparent that the identity of the Thomas household is not hidden by this method alone, namely in the case when the adversary knows the degrees of the vertices. This is due to the fact that it is the only vertex with degree eight. Similarly, the Moore, Anderson, and Smith household can be identified with a 50% probability. This problem can be solved by making the graph 3-degree anonymous, as is shown in Figure 1.2c. The bold edges are edges that have been added to G to make it 3-degree-anonymous. The graph is 3-degree anonymous because there are five vertices of degree 4 (vertices 0, 4, 7, 8, and 9), four vertices of degree 5 (vertices 2, 3, 5, and 6), three vertices of degree 8 (vertices 1, 10, and 11), and no vertices of any other degree. This allows an adversary with knowledge of the degrees of a given individual to only be able to guess the identity with at best a 1/3 probability. This decreases with larger values of k.

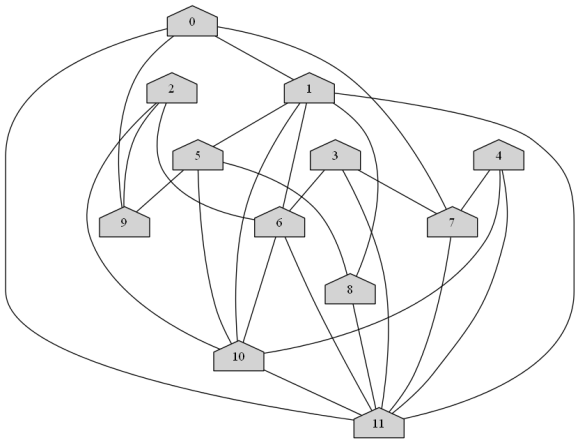
Conceivably, this could lead to fantastic opportunities for climate studies research through social network analysis. Consider BC Hydro, perhaps with a view to improving public opinion of the smart meters, soliciting volunteers to link their social network accounts with their power consumption microdata, yielding a richly annotated social network. If a snapshot of the social network were then released for research purposes, it would provide climate researchers an opportunity to conduct in-depth study of social factors that contribute to home power consumption levels.

However, there is substantial public resistance to smart meters because they present potential privacy breaches [1]. The privacy concerns would naturally be exacerbated by releasing a social network structure that has been labeled with smart meter microdata, independent of the research opportunities proffered. But among those volunteering to associate their consumption data with their social network account, the degrees of concern vary. Some volunteers may, in fact, support smart meters, be excited about contributing to climate studies, and freely permit the disclosure of their microdata and social network structure. Others will want a guarantee of anonymity.

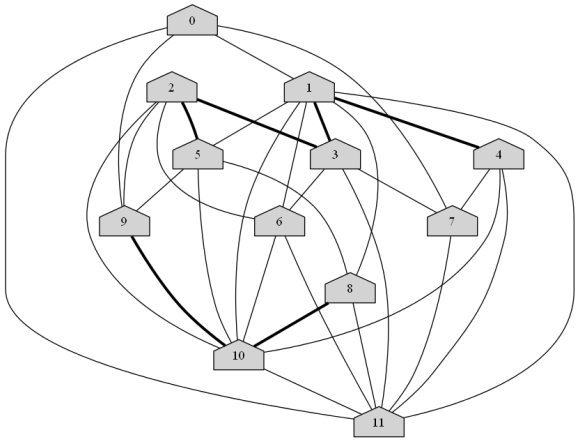
This is a setting for the Subset Graph Anonymization problem. One is given a social network graph and a prespecified subset of members in the social network. The task is to ensure that within this subset, each person is indistinguishable from at



(a) BC Hydro example.



(b) BC Hydro example naively anonymized.



(c) BC Hydro example that is 3-degree-anonymous.

Figure 1.2: Example of BC Hydro data as a graph.

least $k-1$ others. In this way, the data can be published freely because any adversary would be unable to identify any member with certainty better than $\frac{1}{k}$.

Consider another example, a bipartite social network constructed from movies and reviewers where a link corresponds to a reviewer having reviewed a movie. This is another case of *subset anonymity*, because there is no need to offer privacy to the movies. However, it is plausible that an adversary knows how many movies his target has reviewed.

We are the first to develop an algorithm to k-degree anonymize a given subset X of a graph G. We elaborate on our contributions in Section 1.1.

1.1 Contributions

We study the Subset Graph Anonymization problem, wherein the input is a graph G and a subset of its vertices X. The output is a new graph G$'$ similar to G except that enough edges have been added to ensure all the vertices in X have the same degree as at least $k-1$ others. We provide an effective algorithm that we validate empirically. More specifically, we:

- generalize the notion of k-degree anonymity on a given graph G introduced by Liu and Terzi [14] to k-degree anonymity on a given subset X of a graph G, which reflects the common scenario that not an entire graph needs to be anonymized.

- introduce an effective algorithm for the Subset Graph Anonymization problem, the first to take into account the particulars of subset anonymity. The algorithm is based on a novel reduction in the graph's complement to the degree-constrained subgraph problem.

- greatly increase the chance of successfully anonymizing an input graph over the state-of-the-art technique for full graph anonymization of Liu and Terzi [14]. Also, we greatly increase the speed with which an outcome is known. This is due to the advantage of our focus on subset anonymization.

- introduce and implement a greedy method of determining which edges to add within X that greatly improves time and space efficiency, while sacrificing little in terms of minimizing the number of edges added.

1.2 Related Work

Research on data anonymity and the notion of k-anonymity originated in research on table data. Sweeney [21] introduced the idea that table records could be made somewhat anonymous by suppressing enough data values so that each record becomes identical to at least $k-1$ others with respect to the *quasi-identifiers*.[2] The problem of achieving k-anonymity was shown to be NP-hard for $k \geq 3$ by Meyerson and Williams [16].[3]

The notion of k-anonymity was recently adapted for social network graphs. Backstrom et al. [3] demonstrated that even without labels, vertices of a social network graph can be uniquely identified by an adversary who possesses very reasonable background knowledge about the structure of the graph, such as the degree of his target vertex. The structural properties of the graph provide an analogue in social networks to the quasi-identifiers in the table literature.

Degree is perhaps the easiest structural knowledge to acquire and then subsequently exploit. Lui and Terzi [14] introduced k-degree anonymity as a means to protect against this sort of attack. They, and later Chester et al. [5], provided algorithms to produce a k-degree anonymous graph from an arbitrary input graph based on dynamic programming.

Subsequent to [14], numerous other notions of k-anonymity for graphs have been proposed [25, 13, 26, 22, 23, 8, 7], each progressively assuming that the adversary has increasingly sophisticated background knowledge about the structure of the graph, and then strengthening the privacy requirement for publishing. Zheleva and Getoor [25] focused on preserving the privacy of sensitive relationships in graph data, in which an adversary has an accurate predictive model for links, by developing five different methods to deal with such adversaries. Hay et al. [13] assumed that an adversary has a parameterized model of structural knowledge and demonstrated the anonymity risks in random graphs. They also proposed a method of anonymizing a graph by generalizing the nodes into partitions. Zhou and Pei [26] extended l-diversity models, in addition to k-anonymity models, from tabular data to social networks, where the adversary knows the neighbourhood of a target vertex. Thompson and Yao [22] assumed the adversary knows the target vertex degree and the degrees of neighbours within

[2] Quasi-identifiers are attributes such as postal code and birthdate that, when combined, can identify records uniquely.

[3] Subject to conditions on the size of the alphabet that were shown to be unnecessary in subsequent papers.

i hops of the target vertex. Based on this adversarial knowledge, they anonymized a graph in two ways. First, they anonymized a graph in a cluster-based approach. Second, they anonymized a graph by removing and adding edges based on nodes' social roles. Wu et al. [23] assumed the adversary has some background structural knowledge, and therefore formalized the notion of k-symmetry anonymity, where for each vertex there are $k-1$ structurally equivalent counterparts. Cormode et al. [8] worked with bipartite graph data, which, for example, look at an interaction such as that between consumers and products, and anonymized the mapping from entities to nodes of the graph, denoted as (k,l)-groupings. Chester and Srivastava [7] developed an approach to anonymize social networks that have labelled vertices.

Recently, Yuan et al. [24] introduced the idea of subset anonymization, namely that one does not need to anonymize the entire network. This is because different users typically have varied levels of concern for privacy, such as in our smart meter scenario. Also, some vertices may not require privacy at all, such as the movies in our movie reviewer example. The notion of subset anonymity for labeled graphs was formalized by Chester et al. [6], who also showed that achieving subset anonymity for many of the variants is in fact NP-hard. However, for the Subset Graph Anonymization problem, the subject of this thesis, the computational complexity is open.

Finally, research by Gabow [11] contributes to the design of our algorithm. It offers a reduction from the degree-constrained subgraph problem to maximum matching; this can be solved with an established algorithm such as Edmonds' matching algorithm [10] that runs in $\mathcal{O}(|\mathrm{V}|^4)$ but of which there are widely available implementations. The fastest known maximum matching algorithm is due to Mucha and Sankowski [17] and runs in time $\mathcal{O}(|\mathrm{V}|^{2.376})$.

1.3 Overview

Our work here differs from the described k-anonymity papers in that we focus on the subset anonymity problem. Other works that were designed specifically for anonymizing entire graphs do not straight-forwardly apply to subset anonymization because they do not consider that certain edges have higher levels of desirability than others (Lemma 1). Furthermore, while Yuan et al. [24] considered varied levels of concern for privacy and Chester et al. [6] formalized k-subset anonymity, we are the first to tackle the algorithmic question of *how* to produce a good k-degree anonymous subset of a graph. This work has been accepted for publication in the proceedings of

the International Conference on Advances in Social Networks Analysis and Mining (*ASONAM 2012*) [4].

Chapter 2

Background

In this chapter, we begin in Section 2.1 by introducing the necessary notation and definitions. Then in Section 2.2, we define the main problem, Subset Graph Anonymization, as well as other problems useful to solving the Subset Graph Anonymization problem.

2.1 Notation and Definitions

We first start by defining a fundamental way to represent a collection of elements (or objects), termed *set*.

2.1.1 Set Theory

Definition 1. *A* set *is a collection of finite or infinite distinct elements. Let a set be denoted by a collection of elements contained within curly brackets* $\{\}$*. Let the notation* $s \in \mathrm{S}$ *signify that the element* s *is in the set* S *and let* $s \notin \mathrm{S}$ *signify that the element* s *is not in the set* S*. Let the notation* $|\mathrm{S}|$ *signify the size of the set* S*.*

For example, there is the set of all digits, $\mathrm{D} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, or the set of the alphabet from a to g, $\mathrm{A} = \{a, b, c, d, e, f, g\}$. As well, a set can be empty, denoted by either $\{\}$ or $\emptyset$.

A set has a variety of notations that can relate one set to another. One such important notation is when one set is a subset of another.

Definition 2. *A* subset *of a set* S *is a set where every element is in* S*. A set* S' *can be denoted as a* subset *of* S *by* $\mathrm{S}' \subseteq \mathrm{S}$*. Also, a subset can be termed a* proper subset

if a subset S' *of* S *can contain any element in* S*, but not all of* S*. A set* S' *can be denoted as a proper subset of* S *by* $S' \subset S$.

Two examples of a subset of D are $\{1, 7, 8\}$ or $\{2, 3, 4, 5, 9\}$. Two examples of a subset of A are $\{a, c, e, g\}$ or $\{a, b, c, d, e, f, g\}$.

2.1.2 Graph Theory

Definition 3. *A* graph *consists of a set of* vertices *and a set of pairs of vertices called* edges*. In general, a set of vertices is denoted by* V*, a set of edges by* E*, and a graph by* $G = (V, E)$*. We assume all graphs are* simple *(each edge in the graph is distinct and the vertices in each edge are distinct) and* undirected *(the order of the pair of vertices for any edge does not matter). In general, we denote the number of vertices by either* $|V|$ *or* n *and the number of edges by either* $|E|$ *or* m*.*

An example of a graph is where the set of vertices is a collection of people and the set of edges contains each pair of friends out of the collection of people. Facebook is a popular example that can be intuitively thought of as a graph, where each individual using Facebook can be represented by a vertex and any two individuals who are Facebook friends can be represented by an edge. Figure 2.1a shows an example of a graph $G = (V, E)$ where V = {Jack, John, Jason, Jim} and E = {(Jack,John),(John,Jason),(John,Jim),(Jack,Jim)}.

The number of relationships that a person has is captured by the concept of *degree*:

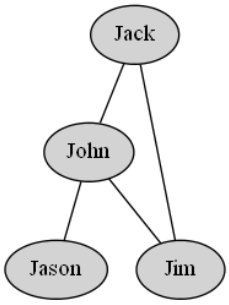
Definition 4. *The* degree d *of a vertex* $v \in V$ *is the number of edges incident to* v*:* $|\{v_i \in V : (v_i, v) \in E, v_i \neq v\}|$.

Figure 2.1b shows an example of a graph $G = (V, E)$ where each vertex is labelled by its degree.

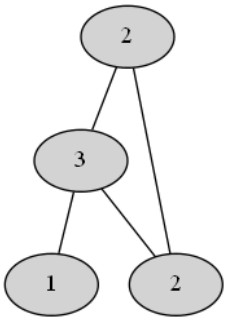
Within this thesis, we often refer to *subgraphs* of graphs, which are derived by taking subsets of the vertex or edge sets:

Definition 5. *Given a graph* $G = (V, E)$*, a subgraph is another graph* $G' = (V', E')$ *where* $V' \subseteq V$ *and* $E' \subseteq E$*. An* induced subgraph *on a vertex set* V' *is the subgraph that contains all edges* $(u, v) \in E$ *where* $u \in V'$ *and* $v \in V'$*.*

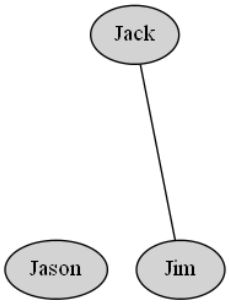
Figure 2.1c shows an example of a graph $G' = (V', E')$ which is a subgraph of the graph in Figure 2.1a, where V = {Jack, Jason, Jim} and E = {(Jack,Jim)}. The



(a) An example of a simple, undirected graph.



(b) An example of a graph where each vertex is labelled with its degree.



(c) An example of the induced subgraph of Figure 2.1a for $V' = \{\text{Jack, Jason, Jim}\}$.

Figure 2.1: Examples of graphs that demonstrate graph notation.

graph G' is also an induced subgraph of the graph in Figure 2.1a. In this thesis, we work with induced subgraphs.

A succinct (although not complete) representation of a graph (or subset of a graph) is its *degree sequence*, which is an ordered list of all its degrees:

Definition 6. *The* degree sequence $S_{\mathrm{X}} = \left(d_1, d_2, \ldots, d_{|\mathrm{X}|}\right)$ *of a subset* $\mathrm{X} \subseteq \mathrm{V}$ *of vertices of graph* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *is the sequence of degrees in* G *for each* $v \in \mathrm{X}$*. For the degree sequence* $\left(d_1, d_2, \ldots, d_{|\mathrm{X}|}\right)$*, we order the degrees such that* $d_1 \geq d_2 \geq \cdots \geq d_{|\mathrm{X}|}$*. Also, we assume the vertices are indexed such that* $\left(d_1, d_2, \ldots, d_{|\mathrm{X}|}\right)$ *corresponds to the vertices* $\left(v_1, v_2, \ldots, v_{|\mathrm{X}|}\right)$*, respectively. Let* $d_i = S_{\mathrm{X}}(v_i)$*, where* v_i *is a vertex in* X*.*

The degree sequence of Figure 1.2a is $(8, 6, 6, 5, 4, 4, 4, 3, 3, 3, 3, 3)$ for (Thomas, Anderson, Smith, Moore, Brown, Jones, Miller, Davis, Johnson, Taylor, Williams, Wilson), respectively. The vertices are assigned and indexed as follows: $v_1 =$ Thomas, $v_2 =$ Anderson, $v_3 =$ Smith, $v_4 =$ Moore, $v_5 =$ Brown, $v_6 =$ Jones, $v_7 =$ Miller, $v_8 =$ Davis, $v_9 =$ Johnson, $v_{10} =$ Taylor, $v_{11} =$ Williams, and $v_{12} =$ Wilson. The degree sequence of Figure 2.1a is $(3, 2, 2, 1)$ for (John, Jack, Jim, Jason), respectively.

The primary objective in this thesis is to produce k-degree anonymous subsets of graphs. Whether a subset of vertices is *k-anonymous* is ascertainable from its degree sequence:

Definition 7. *A degree sequence* S_{X} *is said to be* k-anonymous *if every* d_i *appearing in* S_{X} *appears at least* k *times.*

In Figure 1.2c, the degree sequence is $(8, 8, 8, 5, 5, 5, 5, 4, 4, 4, 4, 4)$ for $(v_1, v_2, \ldots, v_{12})$, respectively, which can observed to be 3-degree anonymous, as there are three or more vertices of degree 4, 5, and 8, and no vertices of any other degree.

Then, a set of vertices is said to be *k-degree anonymous* if its degree sequence is k-anonymous:

Definition 8. *A set* X *of vertices of a graph* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ *is said to be* k-degree anonymous *if the degree sequence* S_{X} *of* X *is itself* k*-anonymous.*

It is also useful to have a measure of how far a degree sequence is from being k-anonymous:

Definition 9. *The* unanonymity *of a set of vertices is the minimum number of degrees that must be removed from the set's degree sequence in order for it to become* k*-anonymous. If the unanonymity is zero, the set of vertices is* k*-degree anonymous.*

The unanonymity in Figure 1.2a for k equal to 3, where its degree sequence is $(8, 6, 6, 5, 4, 4, 4, 3, 3, 3, 3, 3)$, is four, as we must remove the first four degrees in the degree squence. The unanonymity of Figure 1.2c for k equal to 3, is zero, as there are three or more vertices of degree 4, 5, and 8.

The general strategy employed in any of the k-degree anonymity papers is, as a first step in anonymization, to compute from the input graph's degree sequence a new *target degree sequence*. This gives rise to the important concept of a vertex *deficiency*:

Definition 10. *Given* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, $\mathrm{X} \subseteq \mathrm{V}$, *and two equal-length degree sequences, one termed the* source $S_{\mathrm{X}} = \left(d_1, d_2 \ldots, d_{|\mathrm{X}|}\right)$ *and the other termed the* target $S'_{\mathrm{X}} = \left(d'_1, d'_2, \ldots, d'_{|\mathrm{X}|}\right)$, *the* deficiency δ_i *of a vertex* $v_i \in \mathrm{X}$ *is the difference between its degrees in the target and source degree sequences,* $\delta_i = d'_i - d_i$. *A* minimum target degree sequence *is a target degree sequence where, for all vertices* $v_i \in \mathrm{X}$, *the sum of their deficiencies,* $\sum_{v_i \in X} \delta_i$, *is minimum.*

2.2 Problem Definitions

We start by introducing the target problems in this thesis, those of k-degree anonymity for a given subset of a graph:

Problem 1. *Subset Graph Anonymization*
Given an input graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, an integer k, where $1 \leq k \leq n$, and a subset $\mathrm{X} \subseteq \mathrm{V}$ to be anonymized, add a set of edges, E', to G to produce an output graph $\mathrm{G}' = (\mathrm{V}, \mathrm{E} \cup \mathrm{E}')$ such that X is k-degree anonymous and $|\mathrm{E}'|$ is minimum.

While X can be any subset of V, in general we consider $|\mathrm{X}|$ to be small, relative to $|\mathrm{V}|$. We also introduce a auxiliary version of the problem, which is the one that we attempt to solve. However, our algorithm rarely (in fact, throughout our experimental results in Chapter 4, never) produces a solution with an unanonymity higher than zero.

Problem 2. *Near Subset Graph Anonymization*
Given an input graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, an integer k, where $1 \leq k \leq n$, and a subset $\mathrm{X} \subseteq \mathrm{V}$ to be anonymized, add a set of edges, E', to G to produce an output graph $\mathrm{G}' = (\mathrm{V}, \mathrm{E} \cup \mathrm{E}')$ such that *1)* the unanonymity of X is minimized and *2)* subject to the unanonymity being minimized, $|\mathrm{E}'|$ is minimized.

Note that in Problem 2, the subset X is not necessarily k-degree anonymous, but

in order to solve the problem, one must minimize the unanonymity of X, thereby making X as close as possible to being k-degree anonymous. As well, subject to minimizing the unanonymity, one must minimize the number of added edges.

In the context of social networks, our problem corresponds to finding a minimum number of nonexistent relationships that can be added as noise distortion in order to guarantee that nearly everyone has a 100% guarantee of being unrecognizable by degree beyond a $1/k$ probability.

In Figure 2.2a we illustrate the Subset Graph Anonymization problem in the context of our smart meter scenario. The darker shaded houses correspond to hydro customers who require 3-anonymity, the set X, and the lighter houses are the set $\mathrm{V} \setminus \mathrm{X}$. The degrees are indicated within the vertices. The dotted edges are edges which are incident to a vertex in $\mathrm{V} \setminus \mathrm{X}$. The solid edges are edges which are incident to vertices in X. Figure 2.2b illustrates how the addition of three edges, the bold ones, provide 3-anonymity to everyone in X. One can verify this solution is optimal, because there is no way to provide 3-anonymity without adding at least three edges.

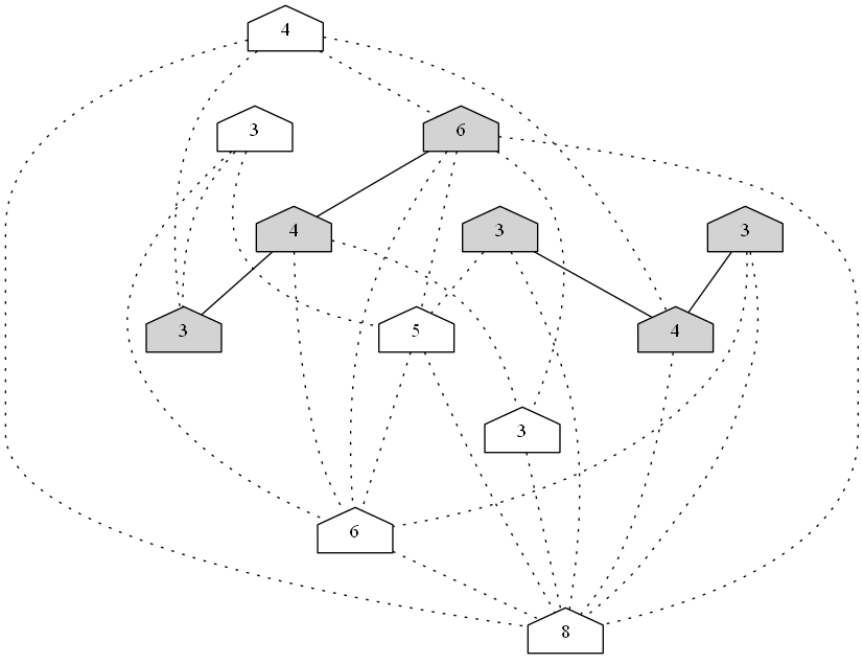
Now, we introduce some known problems from graph theory by Gabow [11] that will be useful in Chapter 3:

Problem 3. *Upper Degree-Constrained Subgraph (UDCS)*
Given an input graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ and an upper bound u_i where $0 \leq u_i \leq d_i$ for each vertex $v_i \in \mathrm{V}$, find a subgraph H of G with the maximum possible number of edges, such that for every vertex $v_i \in \mathrm{H}$, the degree is at most u_i. This problem is also known as *maximum b-matching*.

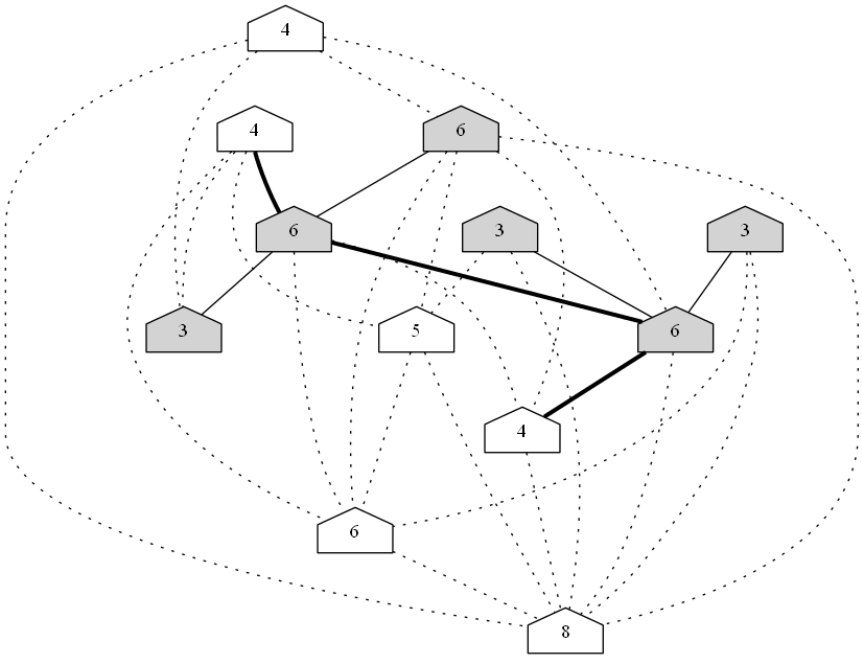
For a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ where a subgraph H is to be determined subject to the UDCS problem, denote the sequence of upper bounds by $U = \big(u_1, u_2, \ldots, u_{|\mathrm{V}|}\big)$ for $\big(v_1, v_2, \ldots, v_{|\mathrm{V}|}\big)$, respectively. As well, let $u_i = U(v_i)$, where v_i is a vertex in V.

Problem 4. *Degree-Constrained Subgraph (DCS)*
Given an input graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, and a lower bound l_i and an upper bound u_i where $0 \leq l_i \leq u_i \leq d_i$ for each vertex $v_i \in \mathrm{V}$, find a subgraph H of G with the maximum possible number of edges, such for every vertex $v_i \in \mathrm{H}$, the degree is between l_i and u_i, inclusive.

For a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ where a subgraph H is to be determined subject to the DCS problem, denote the sequence of lower bounds by $L = \big(l_1, l_2, \ldots, l_{|\mathrm{V}|}\big)$ for $\big(v_1, v_2, \ldots, v_{|\mathrm{V}|}\big)$, respectively. As well, let $l_i = L(v_i)$, where v_i is a vertex in V. The upper bounds have the same notation as the UDCS problem.



(a) BC Hydro example where X is not 3-degree anonymous.



(b) BC Hydro example where X is 3-degree anonymous.

Figure 2.2: Example of BC Hydro data as a graph with a given subset.

Chapter 3

Solving Subset Graph Anonymization and Near Subset Graph Anonymization

In this chapter, we begin in Section 3.1 by discussing a naive solution to the Subset Graph Anonymization problem and the computational complexity of the Subset Graph Anonymization problem. Next, in Section 3.2, we describe a general solution to the Near Subset Graph Anonymization problem. Lastly, in Section 3.3, we present three algorithms for the Near Subset Graph Anonymization problem.

3.1 Subset Graph Anonymization

In this section, we describe a naive solution to Subset Graph Anonymization. Then, we will discuss the computational complexity of Subset Graph Anonymization.

3.1.1 Naive Solution to the Subset Graph Anonymization Problem

While considering an efficient solution to a problem, it is often beneficial to look at a naive solution to the problem. In the case of the Subset Graph Anonymization problem, given an input graph $G = (V, E)$, an integer k, where $1 \leq k \leq n$, and a

subset X $\subseteq$ V, a way to minimize the number of edges added while making X k-degree anonymous is by simply trying every possible selection of edges that could be added, starting at adding one edge and finishing when either X is k-anonymous or at adding the maximum possible number of edges (which creates a complete graph and must be k-degree anonymous). We denote the maximum possible number of edges that could be added to be $e_{\max}$, which is equal to $\binom{n}{2} - m$. In the worst case, where we end up adding the maximum possible number of edges, we would take $\sum_{i=1}^{e_{\max}} \binom{e_{\max}}{i} \cdot \mathcal{O}(n)$ time, where the term $\binom{e_{\max}}{i}$ in the summation is the number of combinations of chosing i edges to add from the possible $e_{\max}$ choices and $\mathcal{O}(n)$ is the time needed to verify that X is k-degree anonymous. While this solves the Subset Graph Anonymization problem, it does so inefficiently.

3.1.2 Computational Complexity of Subset Graph Anonymization

While the solution in the previous section can be improved upon, it is currently unknown whether there exists any efficient solution to Subset Graph Anonymization. If Graph Anonymization is NP-complete, this implies that Subset Graph Anonymization is as well, as the size of X is not constrained in the definition of Subset Graph Anonymization and can be equal to V. This means no solution for Subset Graph Anonymization can be computed in polynomial time if Graph Anonymization is NP-complete and P $\neq$ NP. This gives motivation to solve a relaxation of the Subset Graph Anonymization problems efficiently, which we show can be done in Section 3.3.1, as it is unknown whether there is an efficient solution to the Subset Graph Anonymization problem.

3.2 Near Subset Graph Anonymization

In this section, we describe a solution to the Near Subset Graph Anonymization problem. The solution is broken down into three parts:

1. determining the target degree sequence;
2. adding edges within X; and
3. adding edges outside X.

Lastly, we summarize these algorithms.

3.2.1 Determining the Target Degree Sequence

The first part to solving the Near Subset Graph Anonymization problem is, given an input graph $G = (V, E)$ and a subset $X \subseteq V$, to determine a target degree sequence. We follow the lead of Lui and Terzi [14], who offer a dynamic programming algorithm, DP, with running time $\mathcal{O}(nk)$ and space $\mathcal{O}(n^2)$ to find a minimum target degree sequence. For example, consider Figure 1.2a, where the source degree sequence is $(8, 6, 6, 5, 4, 4, 4, 3, 3, 3, 3, 3)$ for $(v_1, v_2, \ldots, v_{12})$, respectively. For k equal to 3, the minimum target degree sequences are $(8, 8, 8, 5, 5, 5, 5, 3, 3, 3, 3, 3)$ and $(8, 8, 8, 8, 4, 4, 4, 3, 3, 3, 3, 3)$. Notice that the target degree sequence does not specify the assignment of vertices, it only specifies the degree sequence. To account for this, we arbitrarily assign the same ordering of vertices in the source degree sequence in the target degree sequence. For example, given the source degree sequence $(8, 6, 6, 5, 4, 4, 4, 3, 3, 3, 3, 3)$ for $(v_1, v_2, \ldots, v_{12})$, we assign the minimum target degree sequences $(8, 8, 8, 5, 5, 5, 5, 3, 3, 3, 3, 3)$ and $(8, 8, 8, 8, 4, 4, 4, 3, 3, 3, 3, 3)$ an ordering of vertices $(v_1, v_2, \ldots, v_{12})$.

3.2.2 Adding Edges Within X

The second part to solving the Near Subset Graph Anonymization problem is, given a target degree sequence (and in turn, the deficiency of every vertex in X), to determine how to add edges within X (edges where both endpoints are in X). For a solution of the Near Subset Graph Anonymization problem, we denote the set of added edges within X as $E_{X,X}$.

The DCS Method

Our objective is to demonstrate that the DCS problem is a means with which to solve a crucial phase of the Near Subset Graph Anonymization problem. In particular, given the target degree sequence of an optimal solution and its corresponding assignment of vertices, we would have an immediate way to obtain the graph of an optimal solution of the Subset Graph Anonymization problem.

Theorem 1 (Optimality of DCS). *For a given graph* $G = (V, E)$*, subset* $X \subseteq V$*, and optimal target degree sequence* S'_X*, an optimal solution to the DCS problem identifies*

an optimal solution to the Subset Graph Anonymization *problem.*

To prove Theorem 1, we begin by observing that adding edges within X is more valuable than adding edges outside X (edges with one endpoint in X and the other endpoint in $V \setminus X$) in terms of deriving an optimal solution. To arrive at the target degree sequence, the degree of every vertex must be increased by its deficiency. Adding edges that are within X do double work in this regard, as shown in Lemma 1.

Lemma 1 (Adding edges within X is doubly effective)**.** *Given* $G = (V, E)$ *and* $X \subseteq V$*, for any two arbitrary vertices* $x_1, x_2 \in X$ *with deficiencies above zero and vertex* $v \notin X$ *(where* (x_1, x_2)*,* (x_1, v)*,* $(x_2, v) \notin E$ *or previously added), adding edge* (x_1, x_2) *is doubly effective (in terms of minimizing the total deficiency with the minimum number of edge additions) in comparison to adding either edge* (x_1, v) *or* (x_2, v)*.*

Proof. Every added edge containing x_1 as an endpoint reduces the deficiency of x_1 by one. Likewise for x_2. Since edge (x_1, x_2) contains both vertices as endpoints, adding it reduces the total deficiency in the graph by two, at the cost of only adding one edge. On the other hand, adding edges (x_1, v) or (x_2, v) reduce the total deficiency by only one. □

Knowing that adding edges within X is more desirable, we can conclude that solutions that have more edges within X are closer to optimal than solutions with less.

Corollary 1 (Maximum $|E_{X,X}|$ is optimal)**.** *Given* $G = (V, E)$*, an integer* k*, and* $X \subseteq V$*, for any two output graphs (graphs which have an added set of edges* $E' \subseteq (V \times V) \setminus E$*) with the same degree sequence on* X *and the same induced subgraph on* $V \setminus X$*, the one with more added edges within* X *is closer to an optimal solution.*

Proof. Because both output graphs have the same induced subgraph on $V \setminus X$, they only differ in the number of added edges within X and the number of added edges outside X (denote it as $E_{X,V}$). Both graphs have the same degree sequence on X; therefore, both have the same number of edges incident to vertices in X (call it $\mathcal{P}_X$). By Lemma 1, each graph must then have $2\mathcal{P}_X - \lfloor \frac{\mathcal{P}_X}{|E_{X,X}|} \rfloor$ edges, an expression that is smaller for the graph with the larger set $E_{X,X}$. □

Corollary 1 depicts why the DCS problem is so useful to us. We wish to identify as many edges within X as possible in order to do as much work towards satisfying vertex deficiencies with as few edges as possible. In Lemma 2, we show that by

selecting the set of edges for an input to the DCS problem to be only those within X, we can maximize $\mathrm{E}_{\mathrm{X,X}}$.

Lemma 2 (DCS identifies maximum $|\mathrm{E}_{\mathrm{X,X}}|$)**.** *For a given graph* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$*, subset* $\mathrm{X} \subseteq \mathrm{V}$*, and a lower bound* l_i *and an upper bound* u_i *where* $0 \leq l_i \leq d_i \leq u_i$ *for each vertex* $v_i \in \mathrm{X}$*, the graph* G' *with maximum* $|\mathrm{E}_{\mathrm{X,X}}|$ *is identified by the solution to the DCS problem on the complement of the induced subgraph of* G *on* X*, where for every vertex in* $v_i \in \mathrm{X}$*, the degree in* G' *is between* l_i *and* u_i*, inclusive.*

Proof. The complement of G (denoted as $\overline{\mathrm{G}}$) is the graph with the same set of vertices as G, but for every pair of vertices $(x, y) \in \mathrm{V}$, if (x, y) is an edge in G it is not an edge in $\overline{\mathrm{G}}$ and vice versa. By searching for a solution to the DCS problem in the complement of the induced subgraph of G on X, only edges within X that do not already exist in E will be identified to be added to H. The algorithm given by Gabow [11] for the DCS problem maximizes the number of edges added to subgraph H for a given graph, so if this algorithm uses as input the complement of the induced subgraph of G on X, it will output a subgraph H with a set of edges $\mathrm{E}_{\mathrm{X,X}}$ that is maximum and satifies the degree constraints for all vertices in X. □

Lemma 2 shows how an algorithm that solves the DCS problem identifies maximum $|\mathrm{E}_{\mathrm{X,X}}|$, leading to Theorem 1. By solving the DCS problem on the complement of the induced subgraph of G on X that is constrained to add edges for each vertex to satisfy the deficiency, we can, indeed, find the optimal graph that corresponds to an optimal target degree sequence.

Theorem 1 (Optimality of DCS)**.** *For a given graph* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$*, subset* $\mathrm{X} \subseteq \mathrm{V}$*, and optimal target degree sequence* S'_{X}*, an optimal solution to the DCS problem identifies an optimal solution to the* Subset Graph Anonymization *problem.*

Proof. For each vertex $v_i \in \mathrm{X}$, let *outside* be the number of vertices in $\mathrm{V} \setminus \mathrm{X}$ to which v_i is not connected. Then, set the upper bound u_i to be the deficiency of v_i with respect to S'_{X} and set the lower bound l_i to be the larger of zero and $u_i - \mathit{outside}$. By Lemma 2, we can identify the maximum number of edges that we can add strictly within X without overflowing the deficiency of any vertex and ensuring that any vertex with outstanding deficiency can be connected to sufficiently many *outside* vertices. By Corollary 1, this is an optimal solution for the optimal target degree sequence. □

In Theorem 1 we established lower bounds to the DCS problem to ensure that eventually those deficiencies not satisfied with edges within X can be satisfied by

vertices in $\mathrm{V} \setminus \mathrm{X}$. The UDCS problem can be substituted for the DCS problem, however, a solution to the UDCS problem may not identify an optimal solution to the Subset Graph Anonymization problem.

Corollary 2 (UDCS as a simplification)**.** *For a given graph* $\mathrm{G} = (\mathrm{V}, \mathrm{E})$*, subset* $\mathrm{X} \subseteq \mathrm{V}$*, and optimal target degree sequence* S'_{X}*, a solution to the UDCS problem may identify an optimal solution to the* Subset Graph Anonymization *problem.*

Proof. This follows from Theorem 1 by setting the lower bounds of all vertices in X to zero. □

However, there is a possibility that the solution returned by UDCS does not lead to an optimal solution to the Subset Graph Anonymization problem because it could potentially return a solution in which some vertex has not reached its target degree (upper bound) and insufficiently many vertices exist in $\mathrm{V} \setminus \mathrm{X}$ with which to connect it.

In the event that a solution to the DCS or UDCS problem satisfies every deficiency, we have an optimal solution to the Subset Graph Anonymization problem. In other cases, we must then try and add edges outside X, as shown in Section 3.2.3. Similar to the DCS method, using the UDCS method provides a solution to the part of the Near Subset Graph Anonymization problem where we add edges within X.

Lastly, we examine the time and space needed to determine a solution to the DCS or UDCS problem, in order to show that our algorithm is indeed efficient. Given a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ and a set of upper bounds U, the running time to determine a solution to the DCS or UDCS problem is $\mathcal{O}(\sqrt{\sum_{u_i \in U} u_i}|\mathrm{E}|)$ and the space used is $\mathcal{O}(\mathrm{E})$ [11]. However, in solving the Near Subset Graph Anonymization problem, we input the complement of the induced subgraph of G on X which contains $|\mathrm{X}|$ vertices and $\binom{|\mathrm{X}|}{2} - |\mathrm{E}|$ edges. Therefore, the term $\mathcal{O}(|\mathrm{E}|)$ in the original running time is $\mathcal{O}(\binom{|\mathrm{X}|}{2} - |\mathrm{E}|) = \mathcal{O}(|\mathrm{X}|^2)$, which is $\mathcal{O}(n^2)$, as there are at most n vertices in X. As well, $\mathcal{O}(\sqrt{\sum_{u_i \in U} u_i}) = \mathcal{O}(\sqrt{n^2}) = \mathcal{O}(n)$, as the upper bound on any vertex is at most n and again, there are at most n vertices in X. Therefore, the running time is $\mathcal{O}(n^3)$. The space used is $\mathcal{O}(\binom{|\mathrm{X}|}{2} - |\mathrm{E}|) = \mathcal{O}(n^2)$.

The Greedy Method

Another method of determining which edges to add within X is the Greedy algorithm described by Mestre [15]. Specifically, we arbitrarily check every edge X to see whether the upper bounds of the vertices incident to the edge are greater than zero and if so, we add it, making sure to decrement the upper bounds of each of the vertices incident to the edge before checking another edge. For the UDCS problem (or alternatively, maximum b-matching), the solution has been shown to be a 1/2-approximation [15], meaning that, in the worst case, the number of added edges by the Greedy method will be half the maximum number of edges added (in practice it is closer to the maximum number of edges added, which can be seen in the experimental results in Chapter 4). Greedy provides a 1/2-approximation of the UDCS problem in $\mathcal{O}(n^2)$ time and space [15].

3.2.3 Adding Edges Outside X

The last part to solving the Near Subset Graph Anonymization problem is to add any edges outside X. Specifically, we add edges (not already in G) from any deficient vertex in X (a vertex where its deficiency is greater than zero after adding edges within X) to any vertex in $V \setminus X$. We arbitrarily choose any such edges and add them until either the total deficiency is zero or there are no more edges that can be added. We can complete this problem in $\mathcal{O}(n^2)$ time and space.

3.2.4 Relaxation to the Near Subset Graph Anonymization Problem

While, for certain instances of the Near Subset Graph Anonymization problem, it is possible to determine an optimal solution with our previously described solution, in other instances we may not determine the optimal solution. This is due to the fact that the assignment of vertices to the minimum target degree sequence is arbitrary (the same order as the vertices of the source degree sequence). For example, consider a graph G, an integer k, and a subset $X = \{v_1, v_2, \ldots, v_{10}\}$ with a source degree sequence $(4, 4, 3, 2, 2, 1, 1, 1, 1, 1)$ for $(v_1, v_2, \ldots, v_{10})$, respectively. For $k = 3$, the minimum target degree sequence is $(4, 4, 4, 2, 2, 2, 1, 1, 1, 1)$. As we arbitrarily specify the ordering of the vertices the same as the ordering in the source degree sequence, v_3 and v_6 have a deficiency of one and all other vertices have a deficiency of zero. If there

exists an edge between v_3 and v_6, we would end up adding two edges outside X to solve for these deficiencies. However, in the situation where there is an edge between v_3 and v_6, there must not be an edge between v_3 and one or more of v_7, v_8, v_9, v_{10} (as the degree of v_3 is only 3), meaning there exists a solution in which only one edge addition was needed. Therefore, our solution, in this instance, would not find an optimal solution.

We solve a relaxed version of the Near Subset Graph Anonymization problem, not the exact Near Subset Graph Anonymization or Subset Graph Anonymization problem. We instead attempt to minimize the number of edges added. If the found target degree sequence with the arbitrarily assigned order of the vertices is optimal, we find an optimal solution with the DCS method.

3.3 The Algorithm

Overall, Algorithm 1 describes the DCS method, Algorithm 2 describes the UDCS method, and Algorithm 3 describes the Greedy method, all of which we designed. All three algorithms use the DP algorithm from Lui and Terzi [14] in order to determine the minimum target degree sequence and Algorithms 1 and 2 use ideas from Gabow [11] to determine a subgraph satisfying specified degree-constraints. The inputs to all three algorithms are a graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$, a subset $\mathrm{X} \subseteq \mathrm{V}$, and an integer k. The output to all three algorithms is a modified graph G' with a set of edge additions $\mathrm{E}_{\mathrm{X,X}} \cup \mathrm{E}_{\mathrm{X,V}}$.

Algorithm 1 contains the same three parts, as described in Section 3.2, split between lines 2 to 8, 9 to 16, and 17 to 25, for determining the target degree sequence (and the deficiency of each vertex), adding edges within X, and adding edges outside X, respectively.

Lines 2 and 3 compute the degree sequence of X and the target degree sequence, respectively. The target degree sequence has minimum total deficiency and is calculated by the DP algorithm, with k as input, introduced by Lui and Terzi [14]. Next, in lines 4 to 8, the upper and lower bounds are determined for every vertex in X by using the degree sequence of X and the target degree sequence. The upper bounds are calculated in line 5. The lower bounds are calculated in lines 6 and 7. For a vertex $v \in \mathrm{X}$, the set $V_{\mathrm{outside,v}}$ contains the vertices outside X in G which could potentially have an edge added to them from v. The lower bounds are the number of edges that need to be added to v, from another vertex in X, in order for the target degree of v

Algorithm 1 Solution to the NEAR SUBSET GRAPH ANONYMIZATION problem using the DCS method.

procedure NEARSUBSETGRAPHANONYMIZATIONDCS($\mathrm{G}, \mathrm{X}, k$)
  $S_{\mathrm{X}} \leftarrow$ degree sequence of X
  $S'_{\mathrm{X}} \leftarrow$ minimum target degree sequence S_{X} with DP(k)
  for all $v \in \mathrm{X}$ **do**
    $U(v) \leftarrow S'_{\mathrm{X}}(v) - S_{\mathrm{X}}(v)$
    $V_{\mathrm{outside,v}} \leftarrow \{u \in (\mathrm{V} \setminus \mathrm{X}) : (u, v) \notin \mathrm{E}\}$
    $L(v) \leftarrow \max(0, U(v) - |V_{\mathrm{outside,v}}|)$
  end for
  $\overline{\mathrm{G_X}} \leftarrow$ complement of induced subgraph of G on X
  $\mathrm{H} \leftarrow$ subgraph of $\overline{\mathrm{G_X}}$ satisfying degree-constraints L, U
  $\mathrm{E_{X,X}}, \mathrm{E_{X,V}} \leftarrow \{\}$
  for all $e = (u, v) \in \mathrm{H}$ **do**
    $\mathrm{E_{X,X}} \leftarrow \mathrm{E_{X,X}} \cup \{e\}$
    $U(u) \leftarrow U(u) - 1$
    $U(v) \leftarrow U(v) - 1$
  end for
  for all $v \in \mathrm{X}$ **do**
    $\mathrm{E}_{\mathrm{outside,v}} \leftarrow \{(u, v) : u \in (\mathrm{V} \setminus \mathrm{X}), (u, v) \notin \mathrm{E} \cup \mathrm{E_{X,V}}\}$
    while $U(v) > 0$ and $|\mathrm{E}_{\mathrm{outside,v}}| > 0$ **do**
      $e_{\mathrm{outside}} \leftarrow$ arbitrary edge $e \in \mathrm{E}_{\mathrm{outside,v}}$
      $\mathrm{E_{X,V}} \leftarrow \mathrm{E_{X,V}} \cup \{e_{\mathrm{outside}}\}$
      $\mathrm{E}_{\mathrm{outside,v}} \leftarrow \mathrm{E}_{\mathrm{outside,v}} \setminus \{e_{\mathrm{outside}}\}$
      $U(v) \leftarrow U(v) - 1$
    end while
  end for
  return $\mathrm{G}' = (\mathrm{V}, \mathrm{E} \cup \mathrm{E_{X,X}} \cup \mathrm{E_{X,V}})$
end procedure

Algorithm 2 Solution to the NEAR SUBSET GRAPH ANONYMIZATION problem using the UDCS method.

procedure NEARSUBSETGRAPHANONYMIZATIONUDCS($\mathrm{G}, \mathrm{X}, k$)
  $S_{\mathrm{X}} \leftarrow$ degree sequence of X
  $S'_{\mathrm{X}} \leftarrow$ minimum target degree sequence S_{X} with DP(k)
  for all $v \in \mathrm{X}$ **do**
    $U(v) \leftarrow S'_{\mathrm{X}}(v) - S_{\mathrm{X}}(v)$
  end for
  $\overline{\mathrm{G_X}} \leftarrow$ complement of induced subgraph of G on X
  $\mathrm{H} \leftarrow$ subgraph of $\overline{\mathrm{G_X}}$ satisfying upper degree-constraints U
  $\mathrm{E_{X,X}}, \mathrm{E_{X,V}} \leftarrow \{\}$
  for all $e = (u, v) \in \mathrm{H}$ **do**
    $\mathrm{E_{X,X}} \leftarrow \mathrm{E_{X,X}} \cup \{e\}$
    $U(u) \leftarrow U(u) - 1$
    $U(v) \leftarrow U(v) - 1$
  end for
  for all $v \in \mathrm{X}$ **do**
    $\mathrm{E_{outside,v}} \leftarrow \{(u, v) : u \in (\mathrm{V} \setminus \mathrm{X}), (u, v) \notin \mathrm{E} \cup \mathrm{E_{X,V}}\}$
    while $U(v) > 0$ and $|\mathrm{E_{outside,v}}| > 0$ **do**
      $e_{\mathrm{outside}} \leftarrow$ arbitrary edge $e \in \mathrm{E_{outside,v}}$
      $\mathrm{E_{X,V}} \leftarrow \mathrm{E_{X,V}} \cup \{e_{\mathrm{outside}}\}$
      $\mathrm{E_{outside,v}} \leftarrow \mathrm{E_{outside,v}} \setminus \{e_{\mathrm{outside}}\}$
      $U(v) \leftarrow U(v) - 1$
    end while
  end for
  return $\mathrm{G}' = (\mathrm{V}, \mathrm{E} \cup \mathrm{E_{X,X}} \cup \mathrm{E_{X,V}})$
end procedure

Algorithm 3 Solution to the NEAR SUBSET GRAPH ANONYMIZATION problem using the Greedy method.

procedure NEARSUBSETGRAPHANONYMIZATIONGREEDY($\mathrm{G}, \mathrm{X}, k$)
  $S_{\mathrm{X}} \leftarrow$ degree sequence of X
  $S'_{\mathrm{X}} \leftarrow$ minimum target degree sequence S_{X} with DP(k)
  for all $v \in \mathrm{X}$ **do**
    $U(v) \leftarrow S'_{\mathrm{X}}(v) - S_{\mathrm{X}}(v)$
  end for
  $\mathrm{E}_{\mathrm{X,X}}, \mathrm{E}_{\mathrm{X,V}} \leftarrow \{\}$
  for all $e = (u, v) \in \mathrm{X}$ **do**
    if $U(u) > 1$ and $U(v) > 1$ **then**
      $\mathrm{E}_{\mathrm{X,X}} \leftarrow \mathrm{E}_{\mathrm{X,X}} \cup \{e\}$
      $U(u) \leftarrow U(u) - 1$
      $U(v) \leftarrow U(v) - 1$
    end if
  end for
  for all $v \in \mathrm{X}$ **do**
    $\mathrm{E}_{\mathrm{outside,v}} \leftarrow \{(u, v) : u \in (\mathrm{V} \setminus \mathrm{X}), (u, v) \notin \mathrm{E} \cup \mathrm{E}_{\mathrm{X,V}}\}$
    while $U(v) > 0$ and $|\mathrm{E}_{\mathrm{outside,v}}| > 0$ **do**
      $e_{\mathrm{outside}} \leftarrow$ arbitrary edge $e \in \mathrm{E}_{\mathrm{outside,v}}$
      $\mathrm{E}_{\mathrm{X,V}} \leftarrow \mathrm{E}_{\mathrm{X,V}} \cup \{e_{\mathrm{outside}}\}$
      $\mathrm{E}_{\mathrm{outside,v}} \leftarrow \mathrm{E}_{\mathrm{outside,v}} \setminus \{e_{\mathrm{outside}}\}$
      $U(v) \leftarrow U(v) - 1$
    end while
  end for
  return $\mathrm{G}' = (\mathrm{V}, \mathrm{E} \cup \mathrm{E}_{\mathrm{X,X}} \cup \mathrm{E}_{\mathrm{X,V}})$
end procedure

to be satisfied. This is equal to the upper bound of v minus the size of $V_{\text{outside,v}}$, if this difference is greater than zero. If this difference is less than or equal to zero, it means that we have more than enough edges which can be added to v from outside X to satisfy its target degree, so we set the lower bounds to zero.

Lines 9 to 10 compute the subgraph H with a maximum possible number of edges given the degree contraints L, U on the complement of X. In line 9, the complement of X, $\overline{G_{\text{X}}}$, is computed. In line 10, the DCS problem is applied to $\overline{G_{\text{X}}}$ with the degree-constraints L, U found in lines 4 to 8. The sets $\text{E}_{\text{X,X}}$ and $\text{E}_{\text{X,V}}$ are initialized to the empty set in line 11. Lines 12 to 16 add all of the edges found by the DCS problem on $\overline{G_{\text{X}}}$ with the degree-constraints L, U to $\text{E}_{\text{X,X}}$ and decrement the upper bounds of each vertex by one, so that $U(v)$ stores the deficiency for vertex v after adding edges within X.

Lines 17 to 25 arbitrarily add edges to any vertex $v \in \text{X}$ where its deficiency is not zero and possible edges exist that can be added from v to a vertex in $\text{V} \setminus \text{X}$. The set of possible edges which can be added from v to a vertex in $\text{V} \setminus \text{X}$ is denoted by $E_{\text{outside,v}}$. Again, we decrement the upper bounds of each vertex by one. As well, we remove the added edge from $E_{\text{outside,v}}$, so that it cannot be used again.

Algorithm 1 and 2 only differ in three lines of the algorithm (lines 6 and 7 in Algorithm 1 do not exist in Algorithm 2 and Algorithm 1 has lower bounds, as well as upper bounds in line 10, whereas the corresponding line in Algorithm 2 has only upper bounds, as the lower bounds are not needed for UDCS). As these are the only differences, we omit the description of Algorithm 2.

Algorithm 2 and Algorithm 3 only differ in lines 7 to 14. In these lines, Algorithm 3 goes through each possible edge $e \in \text{X}$, adding e to our final graph if the upper bounds of both endpoints is greater than zero, then decrements the upper bound of each endpoint by one if the edge is added. This algorithm can also be completed by adding the edges in a sorted manner, such as descending degree of the vertices for all the edges.

3.3.1 Running Time

We can break the running time and space used of the algorithms into its three parts, shown in Table 3.1. The running time using the UDCS or DCS method is the sum of the three parts, $\mathcal{O}(nk) + \mathcal{O}(n^3) + \mathcal{O}(n^2) = \mathcal{O}(n^3 + nk)$, which is $\mathcal{O}(n^3)$, as k is at most n. The running time using the Greedy method is $\mathcal{O}(n^2)$. The total space used

	Running Time	Space Used
Determining the target degree sequence	$\mathcal{O}(nk)$	$\mathcal{O}(n^2)$
Adding edges within X (DCS or UDCS)	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$
Adding edges within X (Greedy)	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$
Adding edges outside X	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2)$

Table 3.1: Running time of each part of solving the Near Subset Graph Anonymization problem.

is the sum of these three parts, $\mathcal{O}(n^2)$. This shows that our algorithm is efficient.

In Chapter 4, we will demonstrate the usefulness of our algorithms by running them on a variety of datasets.

Chapter 4

Experimental Results

In this chapter, we begin in Section 4.1 by discussing the experimental setup we use to test our algorithms on the Near Subset Graph Anonymization problem. Then, in Section 4.2, we present the results of our experiments on small-world graphs. Next, in Section 4.3, we present the results of our experiments on four real-world graphs. Lastly, in Section 4.4, we evaluate our experimental results, discussing the success rate, the relative numbers of edges added within X and outside X, and running time and space for all trials on all input graphs.

4.1 Experimental Setup

We implement our algorithms, Algorithm 2 and Algorithm 3, in C++ using the Boost 1.47 libraries.[1] For Algorithm 2, we solve the UDCS part of the algorithm with the bipartite-substitute-based UDCS algorithm described by Gabow [11], invoking the implementation of Edmonds' matching algorithm [10] provided by Boost. This involves creating a graph $\overline{G_X}' = (\overline{V}', \overline{E}')$ from $\overline{G_X} = (\overline{V}, \overline{E})$ that contains $\mathcal{O}(n^2)$ vertices and $\mathcal{O}(n^3)$ edges, which can result in time and space efficiency problems. These time and space efficiency problems when using the bipartite-substitute-based UDCS algorithm are a motivating factor for the implementation of Algorithm 3, which uses the Greedy method.

We generate synthetic input graphs in Boost with the small-world random graph generator, with graph settings k-neighbours = 10 and rewiring probability = 0.1. This allows us to report aggregates over many more trials, since for every trial, we

[1] `http://www.boost.org`

generate a new graph.

In general, a small-world graph is one where each vertex is connected to a small set of vertices such that the degrees of all vertices are relatively close, but any two vertices are likely somewhat far apart (in terms of the edges we must travel down until they meet). This can be illustrated by the idea of six degrees of separation, where a person knows a relatively small total number of people on Earth, but as we branch out to the friends of friends six times, we can eventually connect to a large portion of people on Earth. The Boost small-world random graph generator with k-neighbours = 10 and rewiring probability = 0.1 creates a graph where each vertex is connected to its 10 closest neighbours (a ring graph), but each edge is rewired (changed from two vertices to two other vertices) with a probability of 0.1.

We study the performance of Algorithm 2 and Algorithm 3 with respect to three independent variables, varying through all combinations of the input parameters. For each combination, we run one hundred trials and observe the aggregate (count or average) performance. The independent variables of study are:

1. the size of the graph,
   $|\mathrm{V}| \in \{500, 1000, 5000, 10000\}$;

2. the anonymity requirement,
   $k \in \{5, 10, 15, 20, 25\}$; and

3. the size of the anonymizing subset,
   $|\mathrm{X}| \in \{.2|\mathrm{V}|, .35|\mathrm{V}|, .5|\mathrm{V}|, .65|\mathrm{V}|, .8|\mathrm{V}|\}$.

For each trial, we randomly choose the subset $\mathrm{X} \subseteq \mathrm{V}$ of the generated graph $\mathrm{G} = (\mathrm{V}, \mathrm{E})$ for the various sizes of X.

We evaluate four dependent variables:

1. the success rate (produces a k-degree anonymous subset X);

2. the time it takes to produce the output (execution time);

3. the number of edge additions within X;

4. the number of edge additions from X to $\mathrm{V} \setminus \mathrm{X}$.

We measure the execution time of our algorithm with the *timer* class provided by Boost from the moment the input graph has been constructed until the moment our algorithm finishes reporting its solution.

As well as running our algorithm on the small-world graphs, we also run it on the Wikipedia vote network, Enron email network, Epinion social network, and the European research institution email network.[2] We are running both versions of our algorithm on the Wikipedia vote network, Enron email network and Epinion social network, and run our algorithm with the Greedy method on the European research institution email network. The independent variables of study are:

1. the anonymity requirement,
   $k \in \{2, 3, 4, 5\}$; and

2. the size of the anonymizing subset,
   $|\mathrm{X}| \in \{.2|\mathrm{V}|, .35|\mathrm{V}|, .5|\mathrm{V}|, .65|\mathrm{V}|, .8|\mathrm{V}|\}$.

We built the source in Visual Studio 2008 and then ran the experiments in a terminal on a 64 bit dual core Intel T8100 2.10GHz machine running Windows 7 with 4GB of memory.

4.2 Small-World Graphs

In this section we report the results of our experiments described in Section 4.1. Each data point in each plot represents an aggregation (count or average) of several trials run under the same parameters but with different graphs.

4.2.1 The UDCS Method

The experimental results for the implementation of Algorithm 2 run on random small-world graphs can be seen in Figures 4.1, 4.2, 4.3, and 4.4 for $|\mathrm{V}| = 500$, $|\mathrm{V}| = 1000$, $|\mathrm{V}| = 5000$, and $|\mathrm{V}| = 10000$, respectively.

Success Rate

A primary objective in our experimental analysis is to ascertain how often our algorithm successfully produces a k-degree anonymous graph, because there is a theoretical possibility that it will not on some inputs. However, we omit such plots because on all $10,000$ trials, the algorithm produced k-degree anonymous subsets X of the given random small-world graphs.

[2]Network data found at `http://snap.stanford.edu/data/index.html`

Plots of Results

We report the running times for the experiments in Figures 4.1a, 4.2a, 4.3a, and 4.4a. Each point in this plot corresponds to the 100 trials run on graphs for the specified independent variables and reports the average execution time. To summarize the figure, for all 10000 trials run, each took less than seven seconds to anonymize the graph.

The number of edges added within X can be seen in Figures 4.1b, 4.2b, 4.3b, and 4.4b. The number of edges added from X to V \ X can be seen in Figure 4.1c, 4.2c, 4.3c, and 4.4c.

4.2.2 The Greedy Method

The experimental results for the implementation of Algorithm 3 run on random small-world graphs can be seen in Figures 4.5, 4.6, 4.7, and 4.8 for $|V| = 500$, $|V| = 1000$, $|V| = 5000$, and $|V| = 10000$, respectively.

Success Rate

Similar to the UDCS method, the Greedy method proved successful for all 10000 trials and we omit such plots.

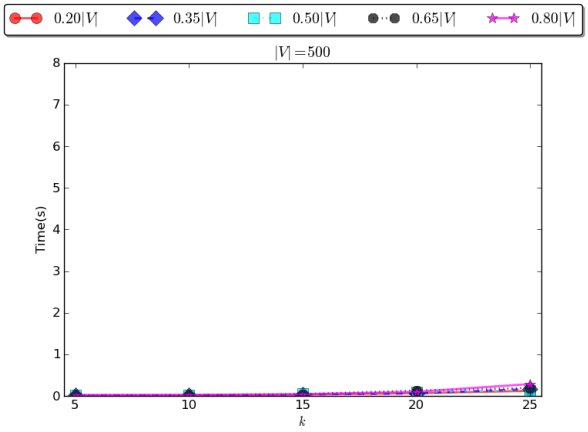
Plots of Results

We report the running times for the experiments in Figures 4.5a, 4.6a, 4.7a, and 4.8a. Each point in this plot corresponds to the 100 trials run on graphs for the specified independent variables and reports the average execution time. To summarize the figure, for all 10000 trials run, each took less than eight seconds to anonymize the graph.

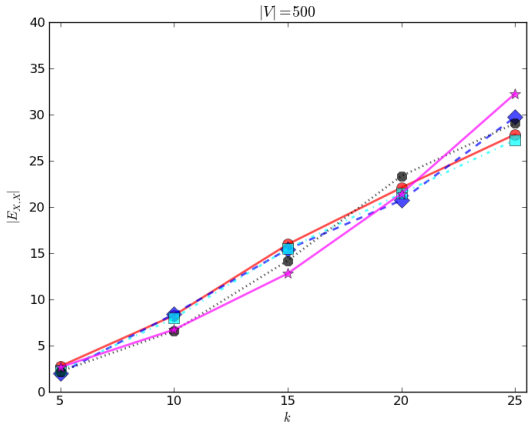
The number of edges added within X can be seen in Figures 4.5b, 4.6b, 4.7b, and 4.8b. The number of edges added from X to V \ X can be seen in Figure 4.5c, 4.6c, 4.7c, and 4.8c.

4.3 Real-World Graphs

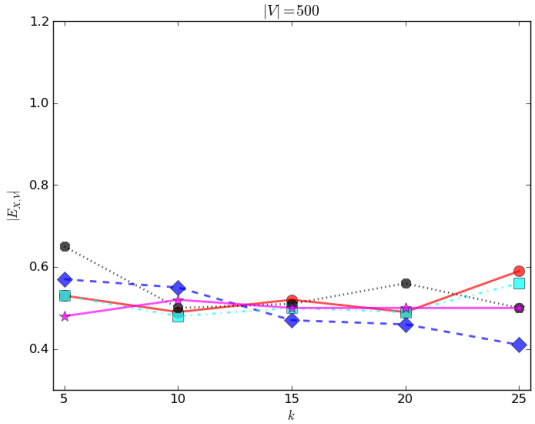
Similar to the experiments for the small-world graphs, in this section we report the results of our experiments described in Section 4.1. Each data point in each plot



(a) Average time taken to run an experiment

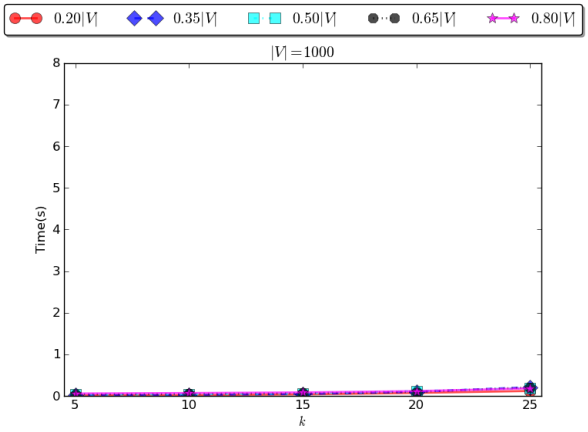


(b) Average number of edges added within X

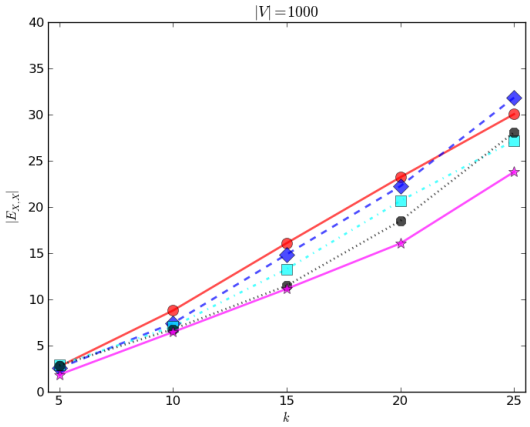


(c) Average number of edges added from X to V \ X

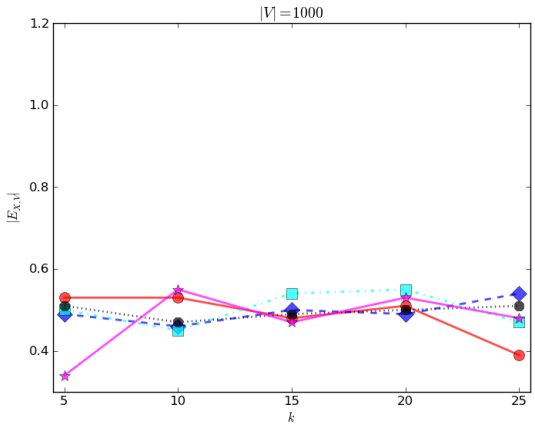
Figure 4.1: Plots of experiments on small-world graphs using the UDCS method for $|V| = 500$.



(a) Average time taken to run an experiment

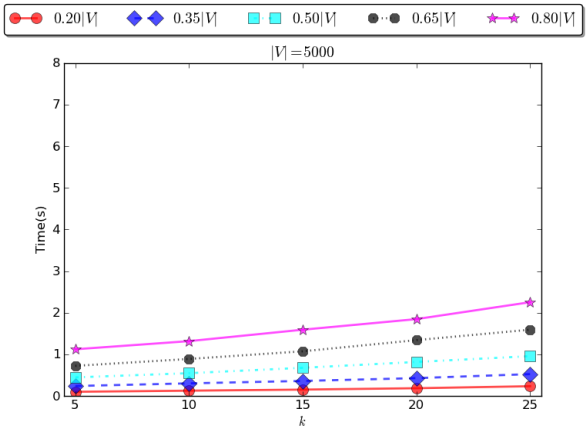


(b) Average number of edges added within X

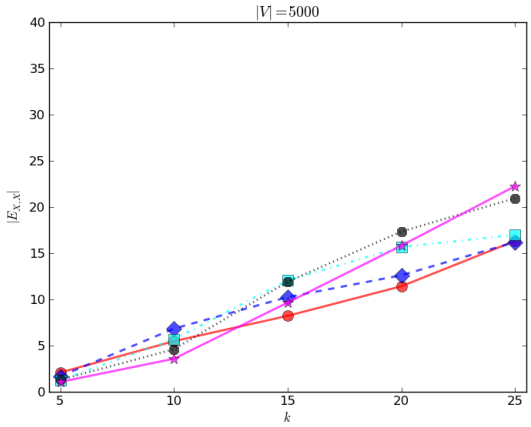


(c) Average number of edges added from X to V \ X

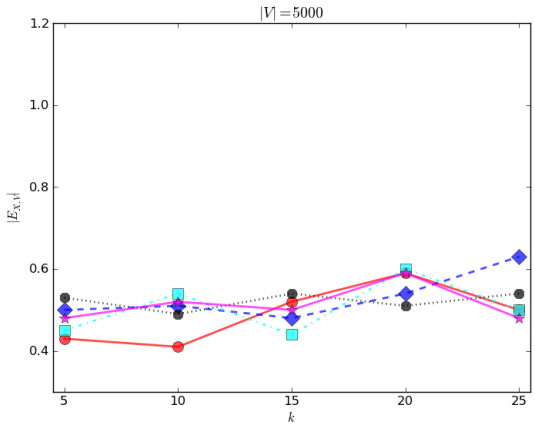
Figure 4.2: Plots of experiments on small-world graphs using the UDCS method for $|V| = 1000$.



(a) Average time taken to run an experiment

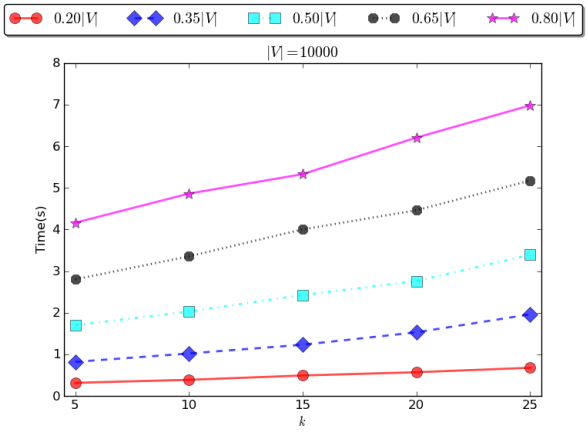


(b) Average number of edges added within X

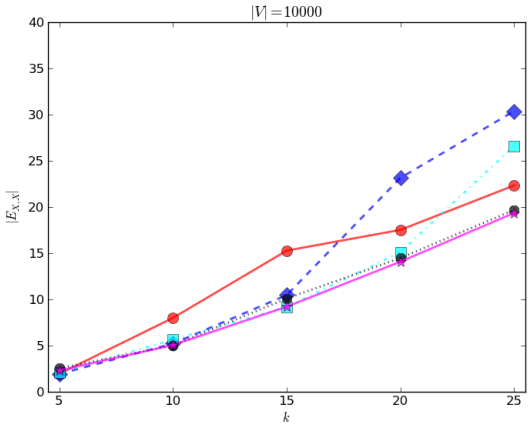


(c) Average number of edges added from X to V \ X

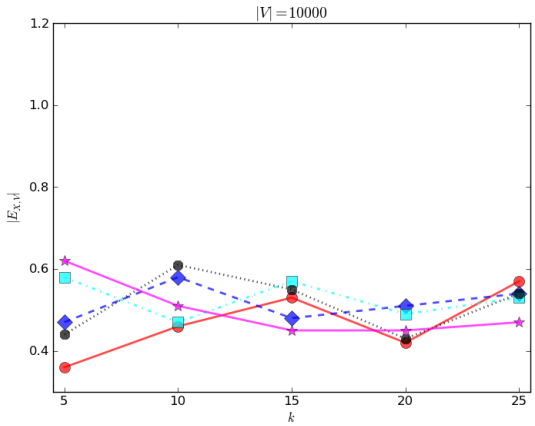
Figure 4.3: Plots of experiments on small-world graphs using the UDCS method for $|V| = 5000$.



(a) Average time taken to run an experiment

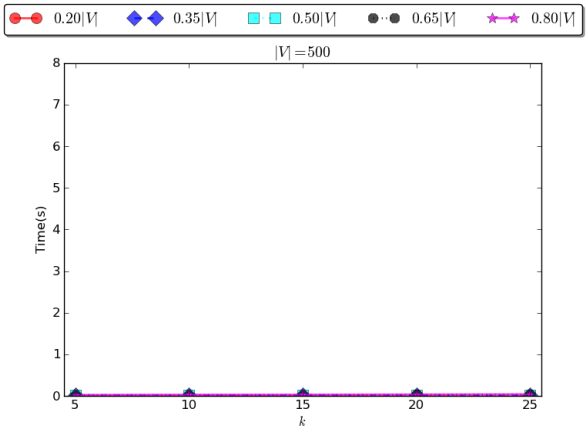


(b) Average number of edges added within X

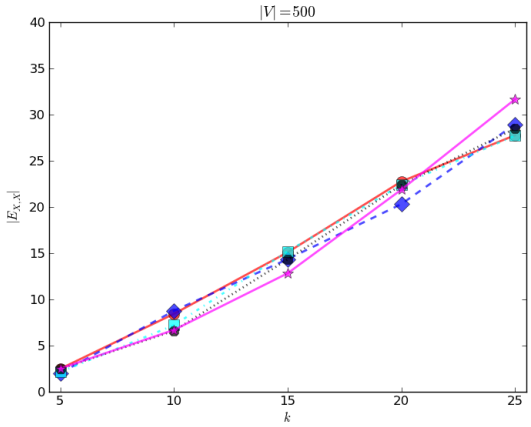


(c) Average number of edges added from X to V \ X

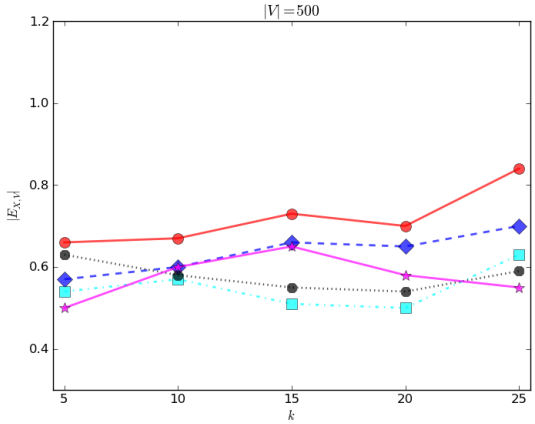
Figure 4.4: Plots of experiments on small-world graphs using the UDCS method for $|V| = 10000$.



(a) Average time taken to run an experiment

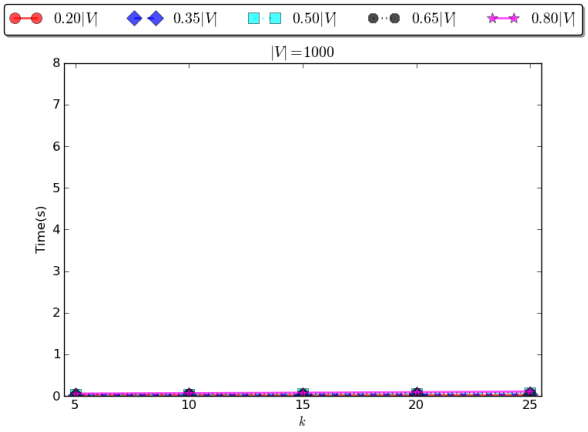


(b) Average number of edges added within X

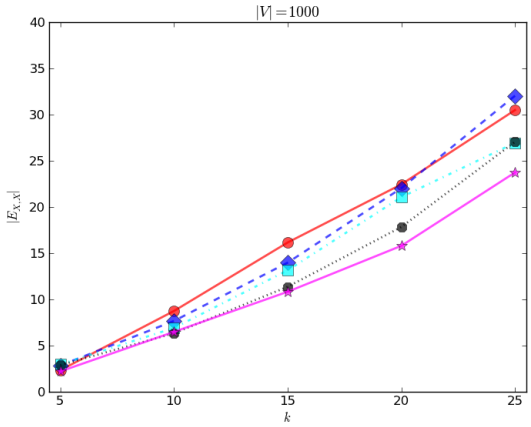


(c) Average number of edges added from X to V \ X

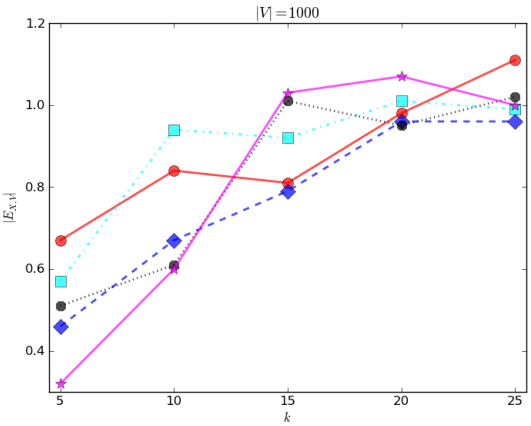
Figure 4.5: Plots of experiments on small-world graphs using the Greedy method for $|V| = 500$.



(a) Average time taken to run an experiment

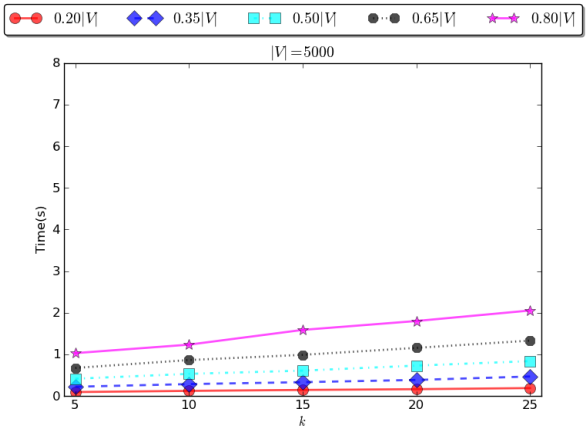


(b) Average number of edges added within X

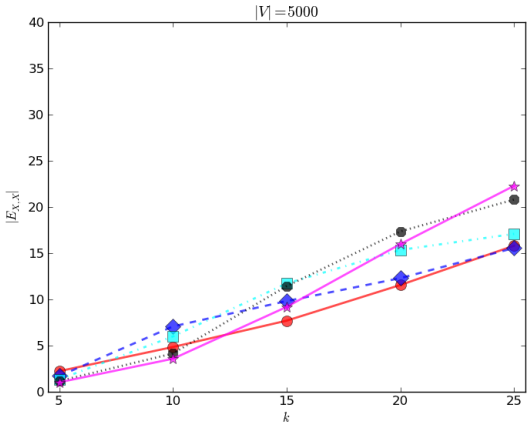


(c) Average number of edges added from X to V \ X

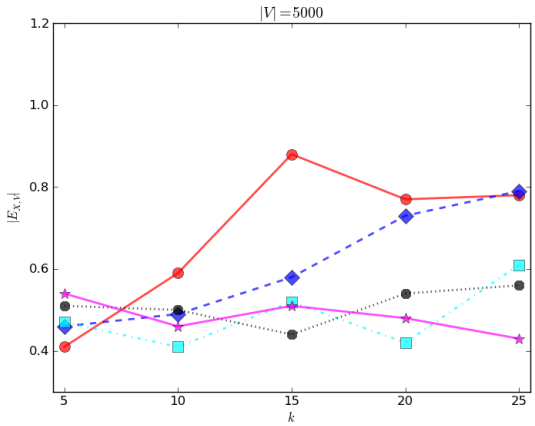
Figure 4.6: Plots of experiments on small-world graphs using the Greedy method for |V| = 1000.



(a) Average time taken to run an experiment

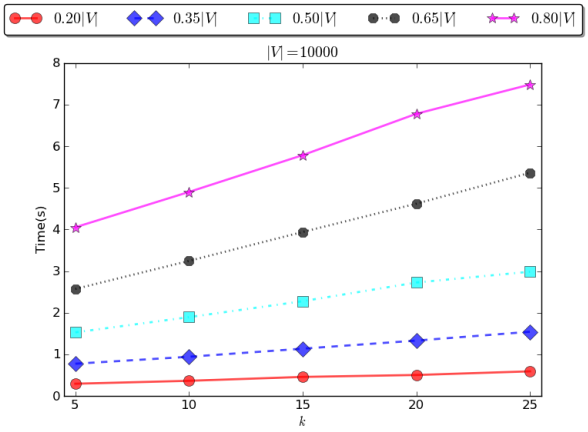


(b) Average number of edges added within X

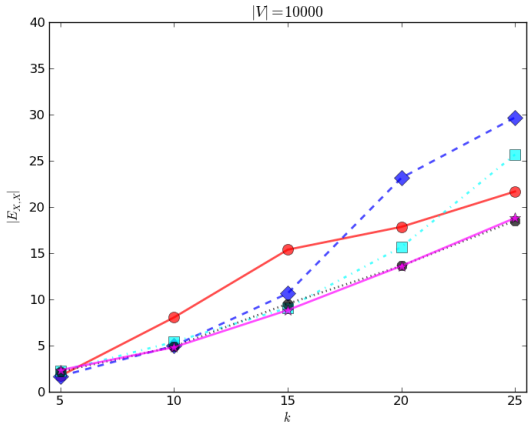


(c) Average number of edges added from X to V \ X

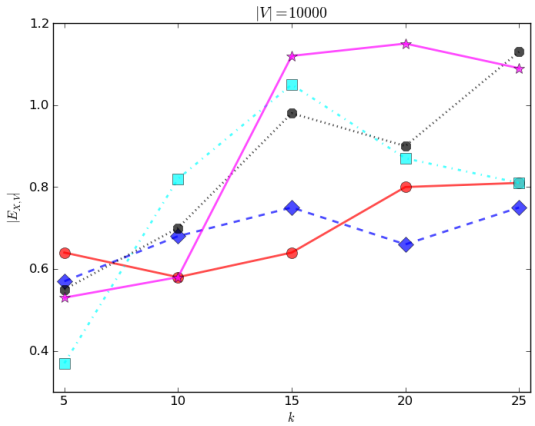
Figure 4.7: Plots of experiments on small-world graphs using the Greedy method for $|V| = 5000$.



(a) Average time taken to run an experiment



(b) Average number of edges added within X



(c) Average number of edges added from X to V \ X

Figure 4.8: Plots of experiments on small-world graphs using the Greedy method for $|V| = 10000$.

	|V|	|E|
Wikipedia vote network	7,115	100,762
Enron email network	36,692	367,662
Epinions social network	75,879	405,740
European research institution email network	265,214	365,570

Table 4.1: Overview of the real-world graphs.

represents an aggregation (count or average) of several trials run under the same parameters but with different graphs. The number of vertices and edges of the graphs experimented are summarized in Table 4.1.

4.3.1 The UDCS Method

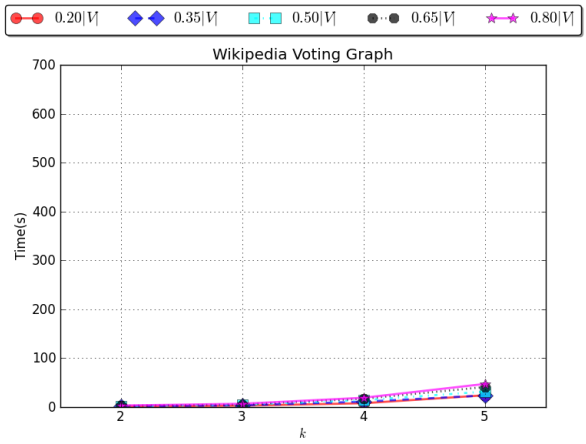
The experimental results for the implementation of Algorithm 2 run on the Wikipedia vote network, the Enron email network, and the Epinions social network can be seen in Figures 4.9, 4.10, and 4.11, respectively.

Success Rate

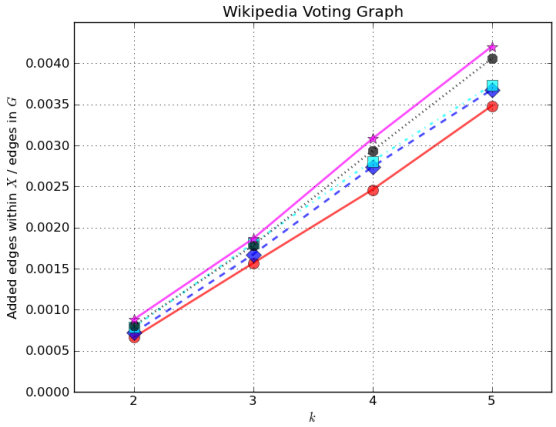
Similar to the small-world graphs, it was found that all trials of all experiments on all of the real-world graphs successfully k-degree anonymized the randomly chosen subset X.

Plots of Results

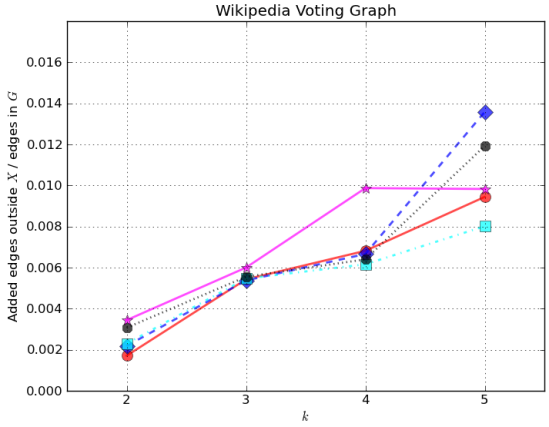
We report the running times for all experiments in Figure 4.9a, 4.10a, and 4.11a. Each point in this plot corresponds to 10 trials (with the exception being the Epinions social network for $k = \{4, 5\}$ and $|\mathrm{X}| = \{0.50|\mathrm{V}|, 0.65|\mathrm{V}|, 0.80|\mathrm{V}|\}$ where only 1 trial was completed due to memory limitations) and reports the average execution time. For all trials, the longest a single trial took was less than 700 seconds to anonymize the graph. Figure 4.9b, 4.10b, and 4.11b show the percentage of edges added, relative to the total number of edges in the graph, within X. Figure 4.9c, 4.10c, and 4.11c show the percentage of edges added, relative to the total number of edges in the graph, from X to $\mathrm{V} \setminus \mathrm{X}$.



(a) Average time taken to run an experiment

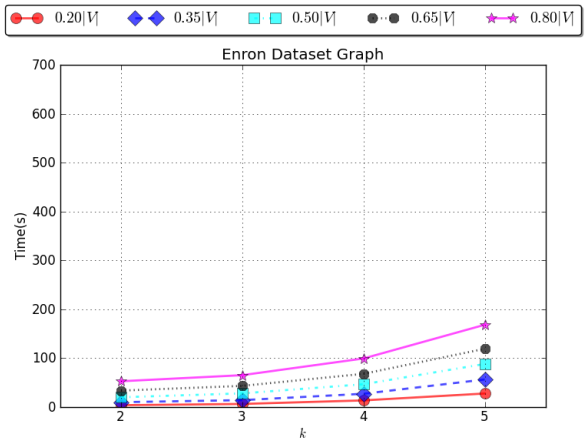


(b) Average number of edges added within X relative to |E|

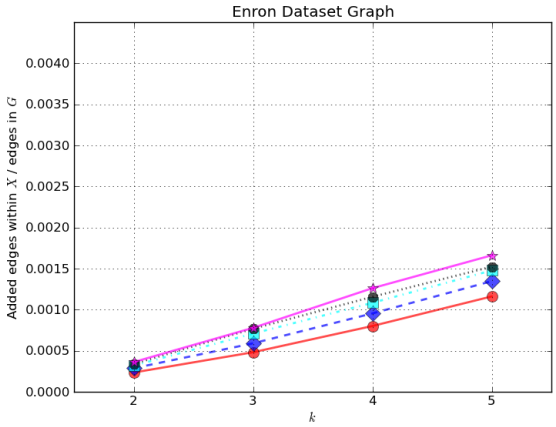


(c) Average number of edges added from X to V \ X relative to |E|

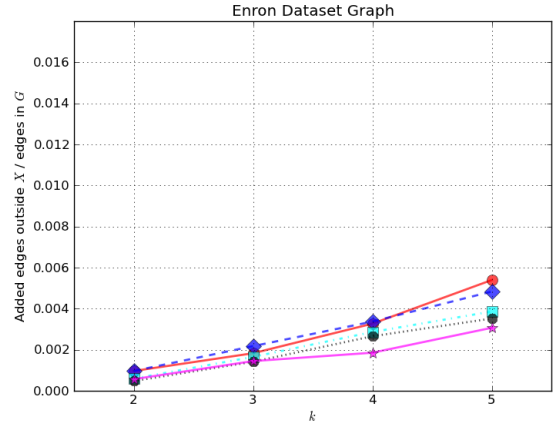
Figure 4.9: Plots of experiments on the Wikipedia vote network using the UDCS method.



(a) Average time taken to run an experiment

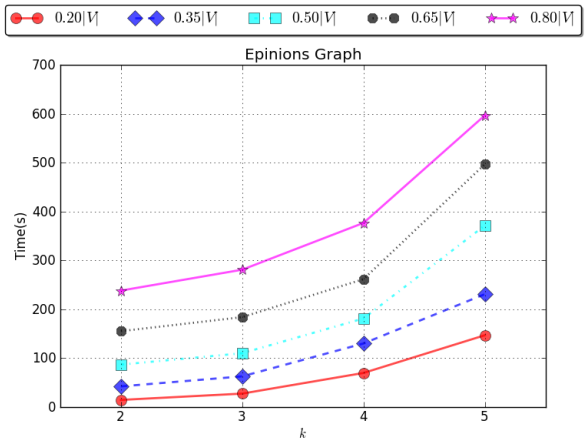


(b) Average number of edges added within X relative to |E|

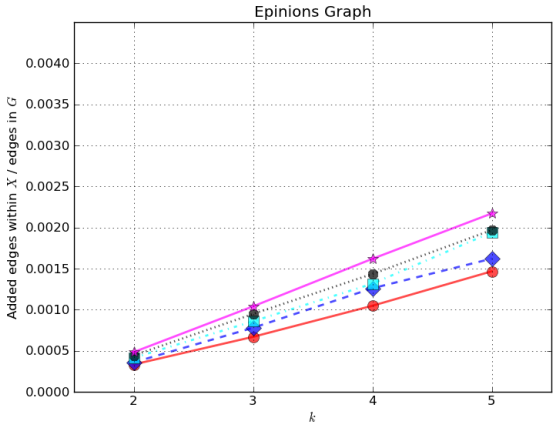


(c) Average number of edges added from X to V \ X relative to |E|

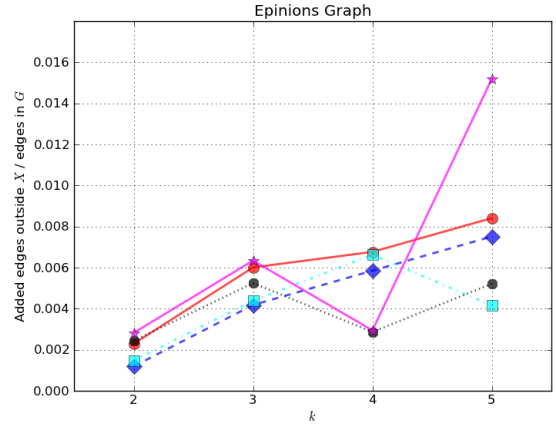
Figure 4.10: Plots of experiments on the Enron email network using the UDCS method.



(a) Average time taken to run an experiment



(b) Average number of edges added within X relative to |E|



(c) Average number of edges added from X to V \ X relative to |E|

Figure 4.11: Plots of experiments on the Epinions social network using the UDCS method.

4.3.2 The Greedy Method

The experimental results for the implementation of Algorithm 3 run on the Wikipedia vote network, the Enron email network, the Epinions social network, and the European research institute email network can be seen in Figures 4.12, 4.13, 4.14, and 4.15, respectively.

Success Rate

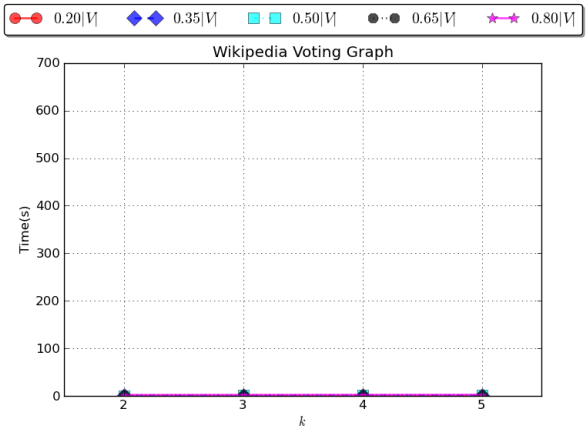
Similar to the small-world graphs, it was found that all trials of all experiments on all of the real-world graphs successfully k-degree anonymized the randomly chosen subset X.

Plots of Results

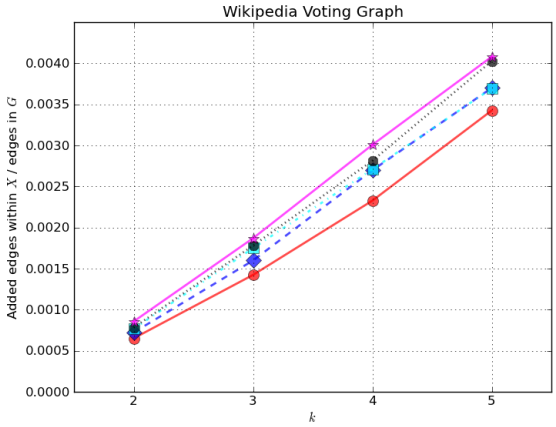
We report the running times for all experiments in Figure 4.12a, 4.13a, 4.14a, and 4.15a. Each point in this plot corresponds to 10 trials and reports the average execution time. For all trials on the Wikipedia vote network, the Enron email network, and the Epinions social network, the longest a single trial took was less than 300 seconds to anonymize the graph. The European research institute email network took less than an hour. Figure 4.12b, 4.13b, 4.14b, and 4.15b show the percentage of edges added, relative to the total number of edges in the graph, within X. Figure 4.12c, 4.13c, 4.14c, and 4.15c show the percentage of edges added, relative to the total number of edges in the graph, from X to $\mathrm{V} \setminus \mathrm{X}$.

4.3.3 The UDCS Method Versus the Greedy Method

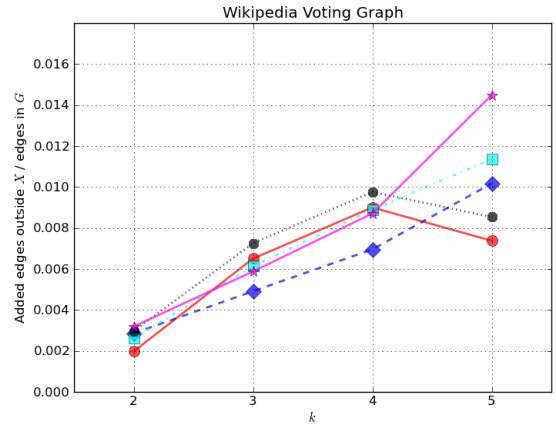
In addition to lower running times, another compelling reason for using the Greedy method versus the UDCS method is the vertex and edge space growth when using the UDCS method, as shown in Figure 4.16, 4.17, and 4.18 for the Wikipedia vote network, the Enron email network, and the Epinions social network, respectively. We denote the bipartite-substitute-based construction for the UDCS method by $\overline{G_{\mathrm{X}}}' = (\overline{\mathrm{V}}', \overline{\mathrm{E}}')$, which is constructed from $\overline{G_{\mathrm{X}}} = (\overline{\mathrm{V}}, \overline{\mathrm{E}})$. The vertex and edge growth ranges from at least 50 times the size of the complement of X when $k = 2$ to just over 450 times the size of the complement of X when $k = 5$, which will negatively affect the time and space efficiency of the UDCS method versus the Greedy method.



(a) Average time taken to run an experiment

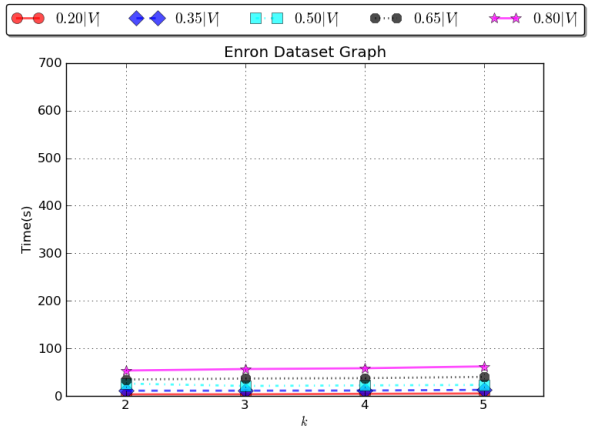


(b) Average number of edges added within X relative to |E|

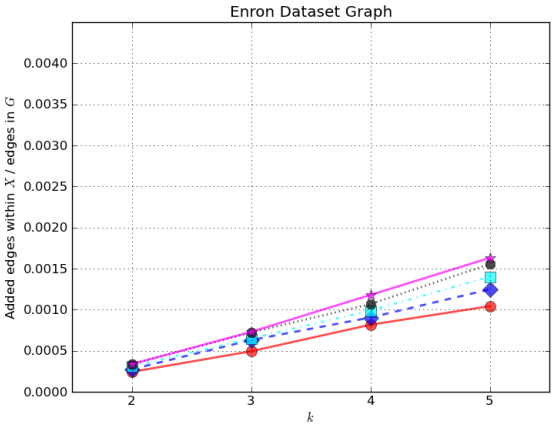


(c) Average number of edges added from X to V \ X relative to |E|

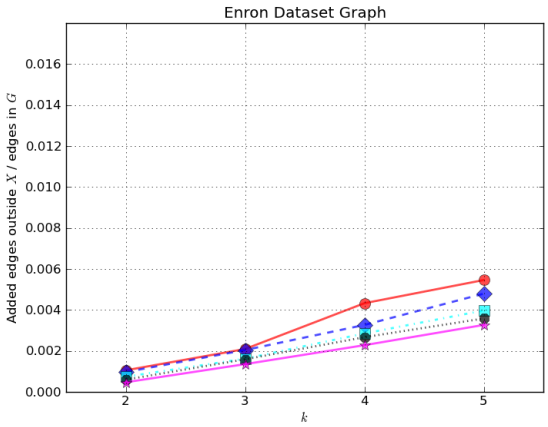
Figure 4.12: Plots of experiments on the Wikipedia vote network using the Greedy method.



(a) Average time taken to run an experiment

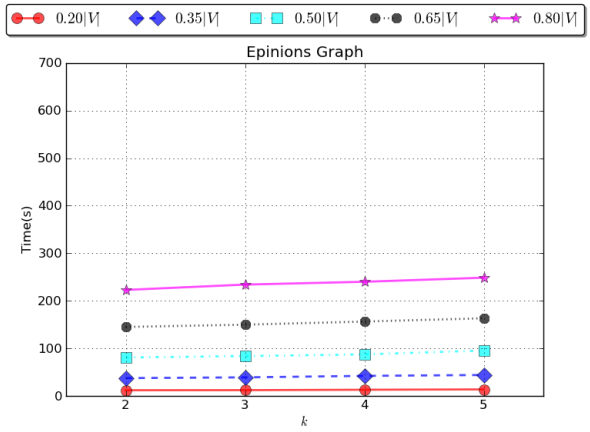


(b) Average number of edges added within X relative to |E|

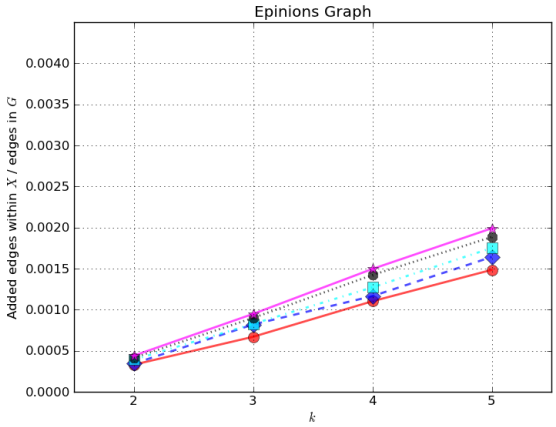


(c) Average number of edges added from X to V \ X relative to |E|

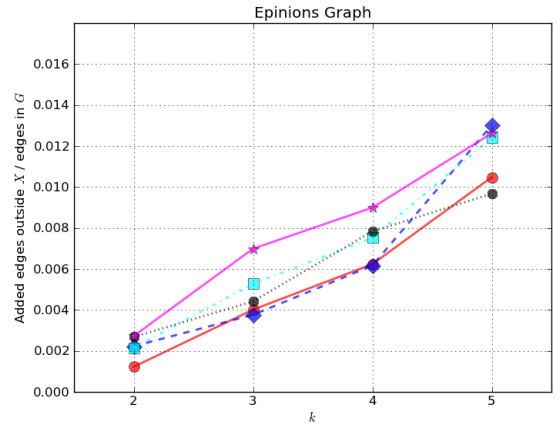
Figure 4.13: Plots of experiments on the Enron email network using the Greedy method.



(a) Average time taken to run an experiment

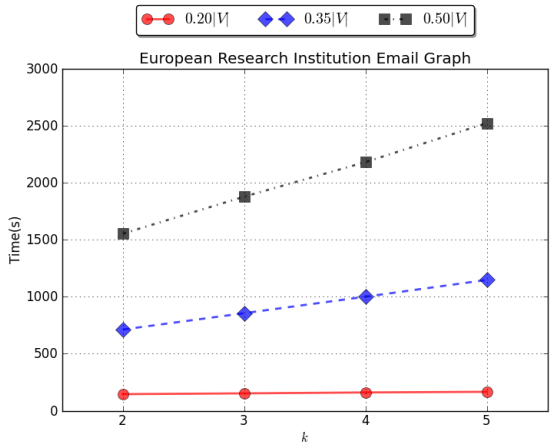


(b) Average number of edges added within X relative to |E|

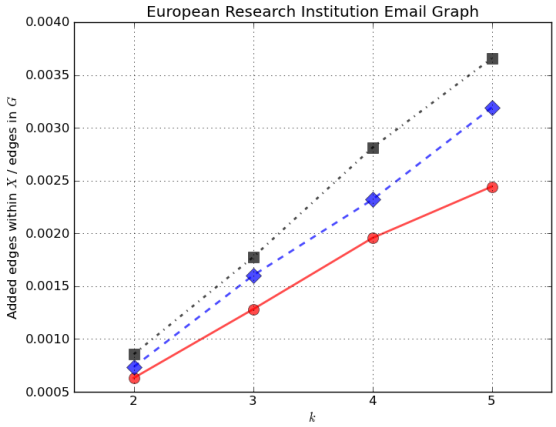


(c) Average number of edges added from X to V \ X relative to |E|

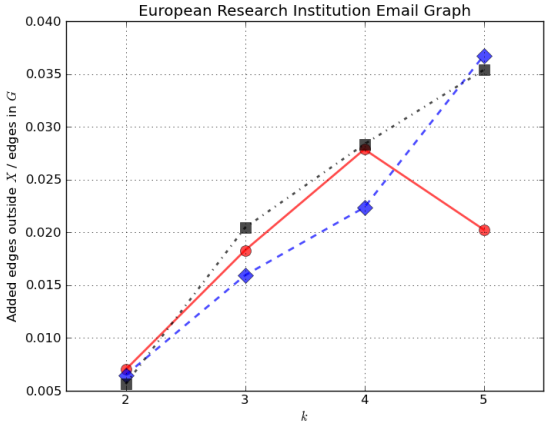
Figure 4.14: Plots of experiments on the Epinions social network using the Greedy method.



(a) Average time taken to run an experiment

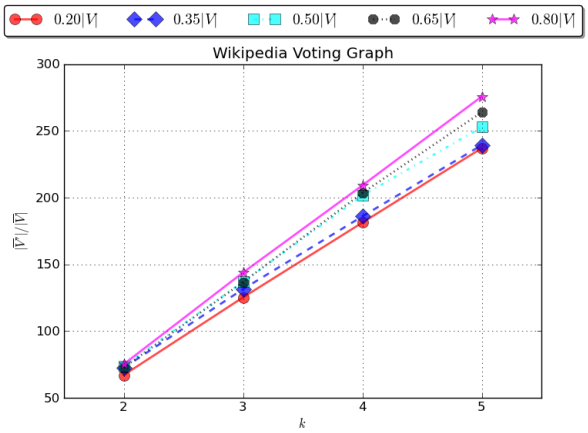


(b) Average number of edges added within X relative to |E|

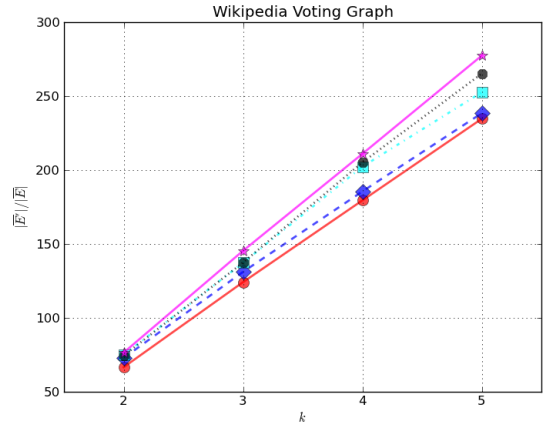


(c) Average number of edges added from X to V \ X relative to |E|

Figure 4.15: Plots of experiments on the European research institution email network using the Greedy method.

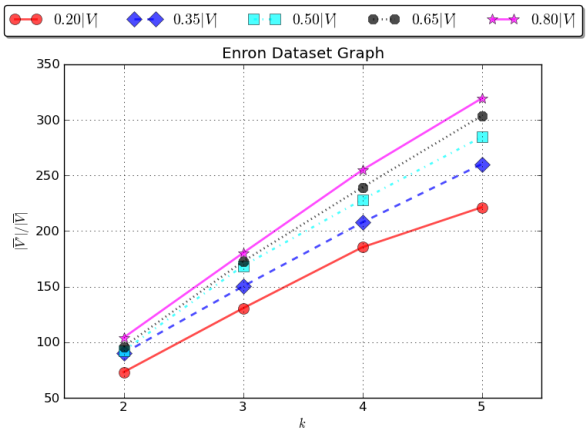


(a) Ratio of growth of the number of vertices from $\overline{\mathrm{V}}$ to $\overline{\mathrm{V}}'$

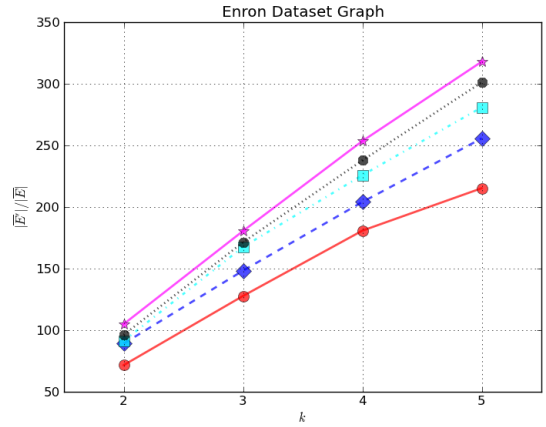


(b) Ratio of the growth of the number of edges from $\overline{\mathrm{E}}$ to $\overline{\mathrm{E}}'$

Figure 4.16: Plots of experiments on the Wikipedia vote network demonstrating the vertex and edge growth of the UDCS method relative to the Greedy method.

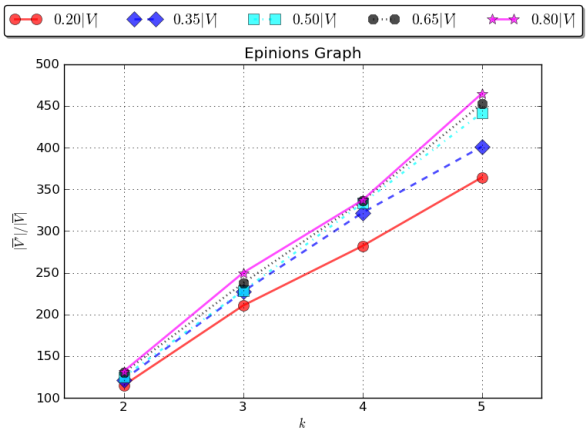


(a) Ratio of growth of the number of vertices from $\overline{\mathrm{V}}$ to $\overline{\mathrm{V}}'$

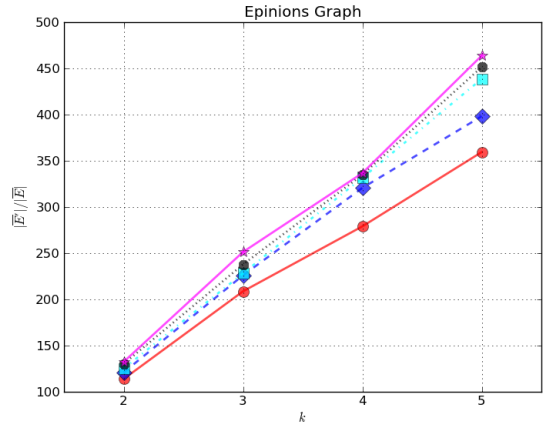


(b) Ratio of the growth of the number of edges from $\overline{\mathrm{E}}$ to $\overline{\mathrm{E}}'$

Figure 4.17: Plots of experiments on the Enron email network demonstrating the vertex and edge growth of the UDCS method relative to the Greedy method.



(a) Ratio of growth of the number of vertices from $\overline{\mathrm{V}}$ to $\overline{\mathrm{V}}'$



(b) Ratio of the growth of the number of edges from $\overline{\mathrm{E}}$ to $\overline{\mathrm{E}}'$

Figure 4.18: Plots of experiments on the Epinions social network demonstrating the vertex and edge growth of the UDCS method relative to the Greedy method.

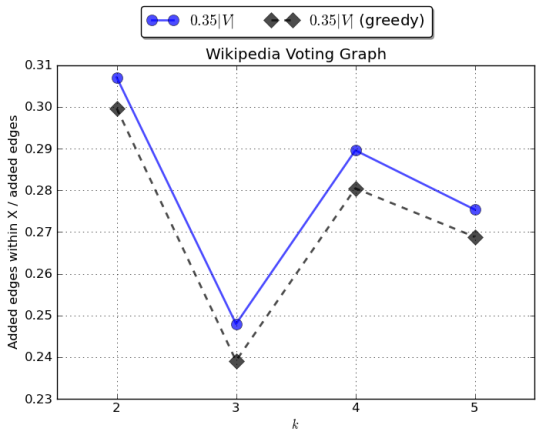


Figure 4.19: Comparison of UDCS Method versus the Greedy method of determining the edges to add within X on the Wikipedia vote network.

One issue with applying the Greedy method versus the UDCS method is that for an optimal target degree sequence, the UDCS method will produce a solution which is optimal (except in the rare instances in real-world graphs where a lower bound is not satisfied, which did not occur in any of our trials), which is not true for the Greedy method. The comparison of the number of edges added within X relative to the total number of edges added for the UDCS method and the Greedy method can be seen in Figure 4.19, 4.20, and 4.21. For each of the experiments on real-world graphs, 10 trials were run where the X chosen is the same for both methods, the size of X is 0.35, and the size of k is one of $\{2, 3, 4, 5\}$.

4.4 Evaluation, Analysis and Comparisons

In general, the results of the experiments in Chapter 4 were very encouraging. Below we present our interpretation of the results on unanonymity (Section 4.4.1) and efficiency (Section 4.4.2).

4.4.1 Unanonymity

Our experiments evaulated the level of unanonymity left by using the UDCS method and the Greedy method on small-world and real-world graphs. Across all trials on all datasets, the performance was perfect, even for the very high values (.8|V|) of |X|. This demonstrates that algorithms using the UDCS method and the Greedy method

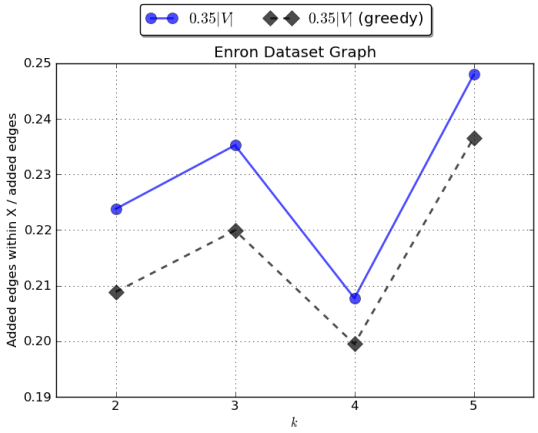


Figure 4.20: Comparison of UDCS method versus the Greedy method of determining the edges to add within X on the Enron email network.

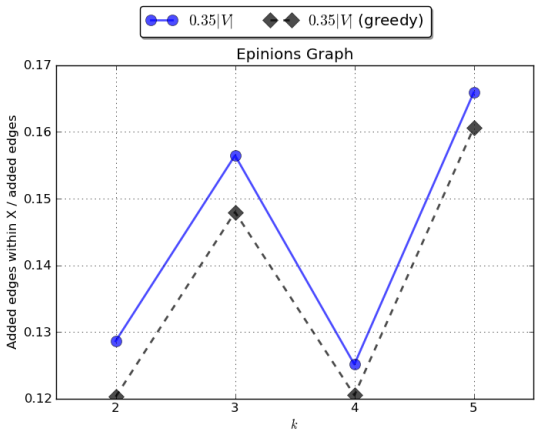


Figure 4.21: Comparison of UDCS method versus the Greedy method of determining the edges to add within X on the Epinions social network.

are an excellent alternative to using the DCS method of determining which edges to add within X.

For the small-world graphs, the number of edges added was fairly consistent for all of the independent variables of study, varying between 15 to 35 edges added within X and 0 to 2 edges added outside X. This is due to the structure of small-world graphs, which causes the degree of most of the vertices to be the same. As most of the degrees of the vertices are the same, there is not much variation in the deficiencies of the vertices and therefore edge additions remain fairly consistent between large variations in the size of the graph. This means that, as the size of the small-world graph increases, the edges required to be added relative to the total number of edges, shrinks, thereby altering the structure of the graph less.

For the real-world graphs, the percentage of edges added within X never exceeds 0.45% and the percentage of edges added from X to $V \setminus X$ never exceeds 4.0% for all datasets using either method of adding edges within X, thereby only altering a small percentage of G in order to make a subset X k-degree anonymous.

4.4.2 Execution Time

Social network anonymization is a task that only needs to be run once before data is released, so execution time is not of utmost importance, so long as it is reasonable. For the real-world graphs, the UDCS method requires eleven minutes at worst and typically only a few minutes. The Greedy method requires less than six minutes at worst on the same real-world graphs and less than an hour on the European research institute email network. For the small-world graphs, both methods required less than eight seconds. In the context of anonymizing data once before publishing it, the running time of our experiments are a manageable amount of time to make a subset of a graph k-degree anonymous.

One observation to make is that there was not a noticeable difference in using the UDCS method versus the Greedy method for small-world graphs. This is due to the fact that the total deficiency is small and the time and memory issues in the UDCS method correspond to the total deficiency. As a result of this, the UDCS method and the Greedy method perform similarly and the differences are most likely due to the fact that the graphs and the subset chosen for a trial is random.

4.4.3 Memory Footprint

We would have liked to experiment with larger synthetic and real-world graphs as well, but the bipartite substitutes in the UDCS method by Gabow [11] inflate an input subset to one with possibly $\mathcal{O}(n^2)$ vertices and $\mathcal{O}(n^3)$ edges, which we could no longer handle with the European research institute email network. However, with the Greedy method, this was accomplished, as the input subset size does not increase.

Chapter 5

Conclusions and Future Work

In this thesis, we build upon the Graph Anonymization problem introduced by Lui and Terzi [14]. They considered the problem of making a graph anonymous, with respect to the degree of the vertices. The idea behind making a graph anonymous is predicated on making the degree sequence of a graph k-anonymous, which entails making each vertex have a degree identical to at least $k-1$ other vertices by adding edges to the graph. However, in some cases only a small portion of a graph is needed to be anonymized, as certain people will willingly supply information, or some vertices may not carry any sensitive information (such as the movies in the reviewer/movie example discussed in the Chapter 1). Due to this possibly unnecessary anonymization of certain vertices, we introduce the Subset Graph Anonymization problem.

5.1 Contributions

The contributions of this thesis are as follows:

1. introduction to the Subset Graph Anonymization problem

   The first contribution is the definition of the Subset Graph Anonymization problem, which did not previously exist. The Graph Anonymization problem is an important stepping stone to many other areas of anonymizing graphs and lead to the focus of this thesis, which is the inclusion of only a subset of all the vertices in a given graph, as it is unnecessary in all situations to make all the vertices of a given graph anonymous.

2. observation of the connection of Subset Graph Anonymization to DCS

The second contribution gives an efficient solution to part of the problem. Specifically, it allows for the number of added edges to be minimized given a target degree sequence and the selection of said edges to be discovered in an efficient manner. This is important, as it shows an optimal solution to the Subset Graph Anonymization problem does not depend on the addition of edges within X, but with the discovery of an optimal target degree sequence.

3. development of the algorithm which solves the Near Subset Graph Anonymization problem

   The third contribution takes all parts of the problem and solves each in an efficient manner, allowing for an algorithm that can output a solution in an efficient manner. Namely, we discover a target degree sequence in $\mathcal{O}(nk)$ time, we can determine the addition of edges within X in $\mathcal{O}(n^3)$ time with the use of the DCS problem, and finally, if needed, we can add edges from X to $V \setminus X$ in $\mathcal{O}(n^2)$ time to discover a solution to a given instance of the Near Subset Graph Anonymization problem.

4. demonstration of usefulness of the Greedy algorithm, motivated by Mestre [15]

   The fourth contribution of this thesis replaces the UDCS method in determining the set of edges to add within X to satisfy a given target sequence. While the Greedy algorithm can guarantee only a 1/2-approximation of the optimal number of added edges, it does not require X to be altered as with the bipartite substitute UDCS method where X can have $\mathcal{O}(n^2)$ vertices and $\mathcal{O}(n^3)$ edges, allowing for a more efficient algorithm. In practice, it was shown that, in fact, the Greedy algorithm performed almost as well as the UDCS version in the number of edges added within X and did not run into the same memory limitations as the UDCS version when given larger graphs, due to the fact that X is unaltered in the Greedy version.

5. implementation of algorithms of Near Subset Graph Anonymization and demonstration of practicality with experiments

   The final contribution takes the algorithm previously developed, and runs it on various instances of real and synthetic data, to show the efficiency and effectiveness of the algorithm. While the implementation of the algorithm relaxed certain details, such as using the UDCS and Greedy method instead

of the DCS method and not implementing the quickest matching algorithm developed by Mucha and Sankowski [17] and instead using Edmonds's matching algorithm [10], it still performed on graphs with thousands of vertices and hundreds of thousands of edges without any failures to k-degree anonymize a given subset of a graph.

5.2 Future Work

We highlight possible directions for future research as follows:

1. NP-completeness of Graph Anonymization

   Proving that Graph Anonymization is NP-complete is an important area of study, as it will give a greater understanding of the Graph Anonymization problem, in that, it will reveal there is no efficient solution to the problem of Graph Anonymization if P $\neq$ NP. This will also show that determining the optimal target degree sequence of a graph cannot be done efficiently, as the DCS problem (which has an efficient solution) can determine the minimum number of edges to add to a given subset and the subset can be chosen such that it is the entire graph. NP-completeness of this will lead to another area of future work, the investigation of the parameterized complexity of Graph Anonymization.

2. parameterized complexity of Graph Anonymization

   A parameterized version of Graph Anonymization is an instance of Graph Anonymization where the running time is considered in terms of one or more parameters of the problem, such a k or e in this instance. We could then attempt to determine fixed-parameter tractable algorithms for the Graph Anonymization problem. For Graph Anonymization, in addition to k and e, we could also consider the maximum degree of any node, the number of different ways to group the various degrees in the anonymized sequence, or any other such parameters of the problem.

3. decreased memory footprint of a solution to Near Subset Graph Anonymization

Currently, as k grows, the size of the $\overline{G_X}'$ in the UDCS method increases, which limits the size of a given graph to the UDCS implementation. One train of thought is to create an implicit version of the graph in the UDCS problem, such that, instead of increasing the number of vertices and edges to solve the problem, each vertex would contain more information about the problem, thereby allowing it to be solved with better space efficiency.

4. alternative ways of adding edges from X to $V \setminus X$

   Currently, edges added from X to $V \setminus X$ are selected arbitrarily from the set of edges that connect a vertex with a deficiency in X to a vertex in $V \setminus X$ where no previous edge exists between the two vertices. However, this does not take into account the structure of the overall graph and whether any new degrees are created in the final degree sequence of G. This could be handled in a more clever fashion, as to preserve the original structure of the graph, as well as to hide any degree patterns that could appear in $V \setminus X$. The problem could also be extended to the addition and removal of edges within the given graph.

5. extended versions of Subset Graph Anonymization

   Other graph properties could be extended to the Subset Graph Anonymization problem, instead of simply the degree of a vertex. This could include the neighbourhood structure of a vertex, as well as any other possible adversarial models.

5.3 Summary

In summary, in this thesis we introduced the first algorithm designed for k-degree anonymity of a subset of a graph. The algorithm was based on a novel use of the DCS problem on the complement of our input graph. We also showed another method, the Greedy method, as a viable alternative to using the DCS problem to solve the problem. We showed empirically that the algorithm was fast and effective. Execution times are typically on the order of a few minutes and the number of new edges added to the graphs is always less than 4.0%. Moreover, in all our experiments, the anonymization was successful, whether on synthetic or real datasets. This is a strong advance on the state-of-the-art, largely in part because we focus on the practical scenario of subset

anonymity that has not been considered in literature. As such, our algorithm (and implementation) is of high practical utility.

Bibliography

[1] Smart meters alleged to pose security risk. *CBC News*. Internet: `http://www.cbc.ca/news/canada/british-columbia/story/2011/10/21/bc-smart-meter-privacy.html`, October 21 2011. [Mar. 20, 2012].

[2] G.M. Anderman and M. Rogers. *Translation today: trends and perspectives*. Multilingual Matters, 2003.

[3] Lars Backstrom, Cynthia Dwork, and Jon M. Kleinberg. Wherefore art thou r3579x? anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the Conference on World Wide Web (WWW)*, pages 181–190. Association for Computing Machinery, 2007.

[4] Sean Chester, Jared Gaertner, Ulrike Stege, and S. Venkatesh. Anonymizing subsets of social networks with degree constrained subgraphs. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE Computer Society, 2012.

[5] Sean Chester, Bruce M. Kapron, Ganesh Ramesh, Gautam Srivastava, Alex Thomo, and S. Venkatesh. k-anonymization of social networks by vertex addition. In *Proceedings of the Advances in Databases and Information Systems (ADBIS)*, pages 107–116. Springer, September 2011.

[6] Sean Chester, Bruce M. Kapron, Gautam Srivastava, and S. Venkatesh. Complexity of social network anonymization. *Social Network Analysis and Mining*, pages 1–16.

[7] Sean Chester and Gautam Srivastava. Social network privacy for attribute disclosure attacks. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE Computer Society, 2011.

[8] Graham Cormode, Divesh Srivastava, Ting Yu, and Qing Zhang. Anonymizing bipartite graph data using safe groupings. *Proceedings of the VLDB Endowment*, 19(1):115–139, 2010.

[9] Cynthia Dwork. Differential privacy. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 1–12. Springer, 2006.

[10] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[11] Harold N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 448–456. Association for Computing Machinery, 1983.

[12] Michael R. Garey and David S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness*. W. H. Freeman & Co Ltd, 1st edition, January 1979.

[13] Michael Hay, Gerome Miklau, David Jensen, Donald F. Towsley, and Philipp Weis. Resisting structural re-identification in anonymized social networks. *Proceedings of the VLDB Endowment*, 1(1):102–114, 2008.

[14] Kun Lui and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of the ACM Special Interest Group on Management of Data (SIGMOD)*, pages 93–106. Association for Computing Machinery, 2008.

[15] Julián Mestre. Greedy in approximation algorithms. In *Proceedings of the European Symposium on Algorithms (ESA)*, pages 528–539. Springer, 2006.

[16] A. Meyerson and R. Williams. General k-anonymization is hard. In *Proceedings of the Symposium on Principles of Database Systems (PODS)*. Association for Computing Machinery, 2004.

[17] Marcin Mucha and Piotr Sankowski. Maximum matchings via gaussian elimination. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 248–255. IEEE Computer Society, 2004.

[18] OED Online. "privacy, n.". *Oxford University Press*. Internet: `http://www.oed.com.ezproxy.library.uvic.ca/view/Entry/151596?redirectedFrom=privacy`, June 2012. [June 14, 2012].

[19] Derrick Penner and Scott Simpson. Smart-meter plan's costs covered by savings: Hydro. *Vancouver Sun*. Internet: `http://www.canada.com/vancouversun/news/business/story.html?id=09daafa8-afd7-49ed-a28f-c85a6664d027&k=89434`, January 19 2011. [Mar. 20, 2012].

[20] D.J. Solove, M. Rotenberg, and P.M. Schwartz. *Privacy, information, and technology*. Elective Series. Aspen Publishers, 2006.

[21] Latanya Sweeney. k-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, October 2002.

[22] Brian Thompson and Danfeng Yao. The union-split algorithm and cluster-based anonymization of social networks. In *Proceedings of the Symposium on Information, Computer and Communications Security (ASIACCS)*, pages 218–227. Association for Computing Machinery, 2009.

[23] Wentao Wu, Yanghua Xiao, Wei Wang, Zhenying He, and Zhihui Wang. k-symmetry model for identity anonymization in social networks. In *Proceedings of the International Conference on Extending Database Technology (EDBT)*, pages 111–122, 2010.

[24] Mingxuan Yuan, Lei Chen, and Philip S. Yu. Personalized privacy protection in social networks. *Proceedings of the VLDB Endowment*, 4(2):141–150, 2010.

[25] Elena Zheleva and Lise Getoor. Preserving the privacy of sensitive relationships in graph data. In *Proceedings of the Privacy, Security, and Trust in KDD (PinKDD)*, pages 153–171. Association for Computing Machinery, 2007.

[26] Bin Zhou and Jian Pei. The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks. *Knowledge and Information Systems*, 28(1):47–77, July 2011.