Improved Upper and Lower Bound Heuristics for Degree Anonymization in Social Networks

Sepp Hartung, Clemens Hoffmann, and André Nichterlein

Institut für Softwaretechnik und Theoretische Informatik, TU Berlin
{sepp.hartung, andre.nichterlein}@tu-berlin.de,
clemens.hoffmann@campus.tu-berlin.de

Abstract. Motivated by a strongly growing interest in anonymizing social network data, we investigate the NP-hard Degree Anonymization problem: given an undirected graph, the task is to add a minimum number of edges such that the graph becomes *k-anonymous*. That is, for each vertex there have to be at least $k-1$ other vertices of exactly the same degree. The model of degree anonymization has been introduced by Liu and Terzi [ACM SIGMOD'08], who also proposed and evaluated a two-phase heuristic. We present an enhancement of this heuristic, including new algorithms for each phase which significantly improve on the previously known theoretical and practical running times. Moreover, our algorithms are optimized for large-scale social networks and provide upper and lower bounds for the optimal solution. Notably, on about 26 % of the real-world data we provide (provably) optimal solutions; whereas in the other cases our upper bounds significantly improve on known heuristic solutions.

1 Introduction

In recent years, the analysis of (large-scale) social networks received a steadily growing attention and turned into a very active research field [6]. Its importance is mainly due the easy availability of social networks and due to the potential gains of an analysis revealing important subnetworks, statistical information, etc. However, as the analysis of networks may reveal sensitive data about the involved users, before publishing the networks it is necessary to preprocess them in order to respect privacy issues [8]. In a landmark paper [11] initiating a lot of follow-up work [4, 9, 12],[1] Liu and Terzi transferred the so-called *k-anonymity* concept known for tabular data in databases [8, 13, 14, 15] to social networks modeled as undirected graphs. A graph is called *k-anonymous* if for each vertex there are at least $k-1$ other vertices of the same degree. Therein, the larger k is, the better the expected level of anonymity is.

In this work we describe and evaluate a combination of heuristic algorithms which provide (for many tested instances matching) lower and upper bounds, for the following NP-hard graph anonymization problem:

[1] According to Google Scholar (accessed Feb. 2014) it has been cited more than 300 times.

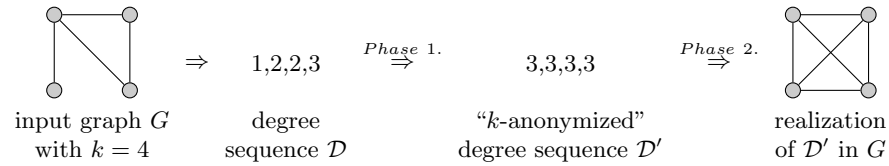

Fig. 1: A simple example for the two phases in the heuristic of Liu and Terzi [11]. Phase 1: Anonymize the degree sequence $\mathcal{D}$ of the input graph G by increasing the numbers in it such that each resulting number occurs at least k times. Phase 2: Realize the k-anonymized degree sequence $\mathcal{D}'$ as a super-graph of G.

Degree Anonymization [11]
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Task: Find a minimum-size edge set E' over V such that adding E' to G results in a k-anonymous graph.

As Degree Anonymization is NP-hard even for constant $k \geq 2$ [9], all known (experimentally evaluated) algorithms, are heuristics in nature [3, 11, 12, 16]. Liu and Terzi [11] proposed a heuristic which, in a nutshell, consists of the following two phases: i) Ignore the graph structure and solve a corresponding number problem and ii) try to transfer the solution from the number problem back to the graph instance. More formally (see Figure 1 for an example), given an instance (G, k), first compute the *degree sequence* $\mathcal{D}$ of G, that is, the multiset of positive integers corresponding to the vertex degrees in G. Then, Phase 1 consists of k-anonymizing the degree sequence $\mathcal{D}$ (each number occurs at least k times) by a minimum amount of increments to the numbers in $\mathcal{D}$ resulting in $\mathcal{D}'$. In Phase 2, try to realize the k-anonymous sequence $\mathcal{D}'$ as a super-graph of G, meaning that each vertex gets a *demand*, which is the difference of its degree in $\mathcal{D}'$ compared to $\mathcal{D}$, and then a "realization" algorithm adds edges to G such that for each vertex the amount of incident new edges equals its demand.

Note that, since the minimum "k-anonymization cost" of the degree sequence $\mathcal{D}$ (sum over all demands) is always a lower bound on the k-anonymization cost of G, the above described algorithm, if successful when trying to realize $\mathcal{D}'$ in G, optimally solves the given Degree Anonymization instance.

Related Work. We only discuss work on Degree Anonymization directly related to what we present here. Our algorithm framework is based on the two-phase algorithm due to Liu and Terzi [11] where also the model of graph (degree-)anonymization has been introduced. Other models of graph anonymization have been studied as well, see Zhou and Pei [18] (studying the neighborhood of vertices) and Chester et al. [4] (anonymizing vertex subsets). We refer to Zhou et al. [19] for a survey on anonymization techniques for social networks. Degree Anonymization is NP-hard for constant $k \geq 2$ and it is W[1]-hard (presumably not fixed-parameter tractable) with respect to the parameter size of a solution size [9]. On the positive side, there is polynomial-size kernel (efficient and effective preprocessing) with respect to the maximum degree of the input graph [9]. Lu et al. [12] and Casas-Roma et al. [3] designed and evaluated heuristic algorithms that are our reference points for comparing our results.

Our Contributions. Based on the two-phase approach of Liu and Terzi [11] we significantly improve the lower bound provided in Phase 1 and provide a simple heuristic for new upper bounds in Phase 2. Our algorithms are designed to deal with large-scale real world social networks (up to half a million vertices) and exploit some common features of social networks such as the power-law degree distribution [1]. For Phase 1, we provide a new dynamic programming algorithm of k-anonymizing a degree sequence $\mathcal{D}$ "improving" the previous running time $\mathcal{O}(nk)$ to $\mathcal{O}(\Delta k^2 s)$, where s denotes the solution size. Note that maximum degree Δ is in our considered instances about 500 times smaller than the number of vertices n. We also implemented a data reduction rule which leads to significant speedups of the dynamic program. We study two different cases to obtain upper bounds. If one of the degree sequences computed in Phase 1 is realizable, then this gives an optimal upper bound and otherwise we heuristically look for "near" realizable degree sequences. For Phase 2 we evaluate the already known "local exchange" heuristic [11] and provide some theoretical justification of its quality.

We implemented our algorithms and compare our upper bounds with a heuristic of Lu et al. [12], called *clustering-heuristic* in the following. Our empirical evaluation demonstrates that in about 26% of the real-world instances the lower bound matches the upper bound and in the remaining instances our heuristic upper bound is on average 40% smaller than the one provided by the clustering-heuristic. However, this comes at a cost of increased running time: the clustering-heuristic could solve all instances within 15 seconds whereas there are a few instances where our algorithms could not compute an upper bound within one hour.

Due to the space constraints, all proofs and some details are deferred to a full version. Most details and proofs are also given in an arxiv-version [10].

2 Preliminaries

We use standard graph-theoretic notation. All graphs studied in this paper are undirected and simple without self-loops and multi-edges. For a given graph $G = (V, E)$ with vertex set V and edge set E we set $n := |V|$ and $m := |E|$. Furthermore, by $\deg_G(v)$ we denote the degree of a vertex $v \in V$ in G and Δ_G denotes the maximum degree in G. For $0 \leq d \leq \Delta_G$ let $B_d^G := \{v \in V \mid \deg_G(v) = d\}$ be the *block* of degree d, that is, the set of all vertices with degree d in G. Thus, being k-anonymous is equivalent to each block being of size either zero or at least k. For a set S of edges with endpoints in a graph G, we denote by $G + S$ the graph that results from inserting all edges from S into G. We call S an *edge insertion set* for G and if $G + S$ is k-anonymous, then it is an *k-insertion set*.

A *degree sequence* $\mathcal{D}$ is a multiset of positive integers and $\Delta_{\mathcal{D}}$ denotes its maximum value. The degree sequence of a graph G with vertex set $V = \{v_1, \ldots, v_n\}$ is $\mathcal{D}_G := \{\deg_G(v_1), \ldots, \deg_G(v_n)\}$. For a degree sequence $\mathcal{D}$, we denote by b_d how often value d occurs in $\mathcal{D}$ and we set $\mathcal{B} = \{b_0, \ldots, b_{\Delta_{\mathcal{D}}}\}$ to be the *block sequence* of $\mathcal{D}$, that is, $\mathcal{B}$ is just the list of the block sizes of G. Clearly, the block sequence of a graph G is the block sequence of G's degree sequence. The block sequence can be viewed as a compact representation of a degree sequence (just

storing the amount of vertices for each degree) and we use these two representations of vertex degrees interchangeably. Equivalently to graphs, a block sequence is k-anonymous if each value is either zero or at least k and a degree sequence is k-anonymous if its corresponding block sequence is k-anonymous.

Let $\mathcal{D} = \{d_1, \ldots, d_n\}$ and $\mathcal{D}' = \{d'_1, \ldots, d'_n\}$ be two degree sequences with corresponding block sequences $\mathcal{B}$ and $\mathcal{B}'$. We define $\|\mathcal{B}\| = |\mathcal{D}| = \sum_{i=1}^{n} d_i$. We write $\mathcal{D}' \geq \mathcal{D}$ and $\mathcal{B}' \ominus \mathcal{B}$ if for both degree sequences sorted in ascending order it holds that $d'_i \geq d_i$ for all i. Intuitively, this captures the interpretation "$\mathcal{D}'$ can be obtained from $\mathcal{D}$ by increasing some values". If $\mathcal{D}' \geq \mathcal{D}$, then (for sorted degree sequences) we define the degree sequence $\mathcal{D}' - \mathcal{D} = \{d'_1 - d_1, \ldots, d'_n - d_n\}$ and set $\mathcal{B}' \ominus \mathcal{B}$ to be its block sequence. We omit sub- and superscripts if the graph is clear from the context.

3 Description of the Algorithm Framework

In this section we present the details of our algorithm framework to solve Degree Anonymization. We first provide a general description how the problem is split into several subproblems (basically corresponding to the two-phase approach of Liu and Terzi [11]) and then describe the corresponding algorithms in detail.

3.1 General Framework Description

We first provide a more formal description of the two-phase approach due to Liu and Terzi [11] and then describe how we refine it: Let $(G = (V, E), k)$ be an input instance of Degree Anonymization.

Phase 1: For the degree sequence $\mathcal{D}$ of G, compute a k-anonymous degree sequence $\mathcal{D}'$ such that $\mathcal{D}' \geq \mathcal{D}$ and $|\mathcal{D} - \mathcal{D}'|$ is minimized.

Phase 2: Try to realize $\mathcal{D}'$ in G, that is, try to find an edge insertion set S such that the degree sequence of $G + S$ is $\mathcal{D}'$.

The minimum k-anonymization cost of $\mathcal{D}$, formally $|\mathcal{D}' - \mathcal{D}|/2$, is a lower bound on the number of edges in a k-insertion set for G. Hence, if succeeding in Phase 2 to realize $\mathcal{D}'$, then a minimum-size k-insertion set S for G has been found.

Liu and Terzi [11] gave a dynamic programming algorithm which exactly solves Phase 1 and they provided the so-called local exchange heuristic algorithm for Phase 2. If Phase 2 fails, then the heuristic of Liu and Terzi [11] relaxes the constraints and tries to find a k-insertion set yielding a graph "close" to $\mathcal{D}'$.

We started with a straightforward implementation of the dynamic programming algorithm and the local exchange heuristic. We encountered the problem that, even when iterating through all minimum k-anonymous degree sequences $\mathcal{D}'$, one often fails to realize $\mathcal{D}'$ in Phase 2. More importantly, we observed the difficulty that iterating through all minimum sequences is often to time consuming because the same sequence is recomputed multiple times. This is because the dynamic program iterates through all possibilities to choose "sections" of consecutive degrees in the (sorted) degree sequence $\mathcal{D}$ that end up in the same

Fig. 2: A graph (left side) with block sequence $\mathcal{B}$ that can be 2-anonymized by adding one edge (right side) resulting in $\mathcal{B}'$. Another 2-anonymous block sequence (also of cost two) that will be found by the dynamic programming is $\mathcal{B}'' = \{0, 2, 2, 4, 0, 0, 2\}$. The realization of $\mathcal{B}''$ in G would require to add an edge between a degree-five vertex (there is only one) and a degree-one vertex, which is impossible.

block in $\mathcal{D}'$. These sections have to be of length at least k (the final block has to be full) but at most $2k - 1$ (longer sections can be split into two). However, if there is a huge block B (of size $\gg 2k$) in $\mathcal{D}$, then the algorithm goes through all possibilities to split B into sections, although it is not hard to show that at most $k - 1$ degrees from each block are increased. Thus, different ways to cut these degrees into sections result in the same degree sequence.

We thus redesigned the dynamic program for Phase 1. The main idea is to consider the block sequence of the input graph and exploiting the observation that at most $k - 1$ degrees from a block are increased in a minimum-size solution. Therefore, we avoid to partition one block into multiple sections and the running time dependence on the number of vertices n can be replaced by the maximum degree Δ, yielding a significant performance increase.

We also improved the lower bound provided by $\mathcal{D}' - \mathcal{D}$ on the k-anonymization cost of G. To this end, the basic observation was that while trying to realize one of the minimum k-anonymous sequences $\mathcal{D}'$ in Phase 2 (failing in almost all cases), we encountered that by a simple criterion on the sequence $\mathcal{D}' - \mathcal{D}$ one can even prove that $\mathcal{D}'$ is not realizable in G. That is, a k-insertion set S for G corresponding to $\mathcal{D}'$ would induce a graph with degree sequence $\mathcal{D}' - \mathcal{D}$. Hence, the requirement that there is a graph with degree sequence $\mathcal{D}' - \mathcal{D}$ is a necessary condition to realize $\mathcal{D}'$ in G in Phase 2. Thus, for increasing cost c, by iterating through all k-anonymous sequences $\mathcal{D}'$ with $|\mathcal{D}' - \mathcal{D}| = c$ and excluding the possibility that $\mathcal{D}'$ is realizable in G by the criterion on $\mathcal{D}' - \mathcal{D}$, one can step-wisely improve the lower bound on the k-anonymization cost of G. We apply this strategy and thus our dynamic programming table allows to iterate through all k-anonymous sequences $\mathcal{D}'$ with $|\mathcal{D}' - \mathcal{D}| = c$. Unfortunately, even this criterion might not be sufficient because the already present edges in G might prevent the insertion of a k-insertion set which corresponds to $\mathcal{D}' - \mathcal{D}$ (see Figure 2 for an example). We thus designed a test which not only checks whether $\mathcal{D}' - \mathcal{D}$ is realizable but also takes already present edges in G into account while preserving that $|\mathcal{D}' - \mathcal{D}|$ is a lower bound on the k-anonymization cost of G. With this further requirement on the resulting sequences $\mathcal{D}'$ of Phase 1, in our experiments we observe that Phase 2 of realizing $\mathcal{D}'$ in G is in 26 % of the real-world instances successful. Hence, 26 % of the instances can be solved optimally. See Subsection 3.2 for a detailed description of our algorithm for Phase 1.

For Phase 2 the task is to decide whether a given k-anonymization $\mathcal{D}'$ can be realized in G. As we will show that this problem is NP-hard, we split the problem into two parts and try to solve each part separately by a heuristic. First, we find a degree-vertex mapping, that is, we assign each degree $d'_i \in \mathcal{D}'$ to a vertex v in G such that $d'_i \geq \deg_G(v)$. Then, the demand of vertex v is set to $d'_i - \deg_G(v)$. Second, given a degree-vertex mapping with the corresponding demands we try the find an edge insertion set such that the number of incident new edges for each vertex is equal to its demand. While the second part could in principle be done optimally in polynomial-time by solving an f-factor problem [9], we show that already a heuristic refinement of the "local exchange" heuristic due to Liu and Terzi [11] is able to succeed in most cases. Thus, theoretically and also in our experiments, the "hard part" is to find a good degree-vertex mapping. Roughly speaking, the difficulties are that, according to $\mathcal{D}'$, there is more than one possibility of how many vertices from degree i are increased to degree $j > i$. Even having settled this it is not clear which vertices to choose from block i. See Subsection 3.3 for a detailed description of our algorithm for Phase 2.

3.2 Phase 1: Exact k-Anonymization of Degree Sequences

We start with providing a formal problem description of k-anonymizing a degree sequence $\mathcal{D}$ and describe our dynamic programming algorithm to find such sequences $\mathcal{D}'$. We then describe the criteria that we implemented to improve the lower bound $|\mathcal{D}' - \mathcal{D}|$.

Basic Number Problem. The decision version of the degree sequence anonymization problem reads as follows.

k-Degree Sequence Anonymity (k-DSA)
Input: A block sequence $\mathcal{B}$ and integers $k, s \in \mathbb{N}$.
Question: Is there a k-anonymous block sequence $\mathcal{B}' \supseteq \mathcal{B}$ such that $\|\mathcal{B}' \ominus \mathcal{B}\| = s$?

The requirements on $\mathcal{B}'$ in the above definition ensure that $\mathcal{B}'$ can be obtained by performing exactly s many increases to the degrees in $\mathcal{B}$. Liu and Terzi [11] gave a dynamic programming algorithm that solves k-DSA optimally in $\mathcal{O}(nk)$ time and space. Here, besides using block instead of degree sequences, we added another dimension to the dynamic programming table storing the cost of a solution.

Lemma 1. k-Degree Sequence Anonymity *can be solved in* $\mathcal{O}(\Delta \cdot k^2 \cdot s)$ *time and* $\mathcal{O}(\Delta \cdot k \cdot s)$ *space.*

There might be multiple minimum solutions for a given k-DSA instance while only one of them is realizable, see Figure 2 for an example. Hence, instead of just computing one minimum-size solution, we iterate through these minimum-size solutions until one solution is realizable or *all* solutions are tested. Observe that there might be exponentially many minimum-size solutions: In the block sequence $\mathcal{B} = \{0, 3, 1, 3, 1, \ldots, 3, 1, 3\}$, for $k = 2$, each subsequence $3, 1, 3$ can be either changed to $2, 2, 3$ or to $3, 0, 4$. We use a data reduction rule to reduce the amount of considered solutions in such instances.

Criteria on the Realizability of k-DSA Solutions. A difficulty in the solutions provided by Phase 1, encountered in our preliminary experiments and as already observed by Lu et al. [12] on a real-world network, is the following: If a solution increases the degree of one vertex v by some amount, say 100, and the overall number of vertices with increased degree is at most 100, then there are not enough neighbors for v to realize the solution. We overcome this difficulty as follows: For a k-DSA-instance $(\mathcal{B}, k)$ and a corresponding solution $\mathcal{B}'$, let S be a k-insertion set for G such that the block sequence of $G + S$ is $\mathcal{B}'$. By definition, the block sequence of the graph induced by the edges S is $\mathcal{B}' \ominus \mathcal{B}$. Hence, it is a necessary condition (for success in Phase 2) that $\mathcal{B}' \ominus \mathcal{B}$ is a *realizable* block sequence, that is, there is a graph with block sequence $\mathcal{B}' \ominus \mathcal{B}$. Tripathi and Vijay [17] have shown that it is enough to check to following *Erdős-Gallai characterization* of realizable degree sequence just once for each block.

Lemma 2 ([7]). *Let $\mathcal{D} = \{d_1, \ldots, d_n\}$ be a degree sequence sorted in descending order. Then $\mathcal{D}$ is realizable if and only if $\sum_{i=1}^{n} d_i$ is even and for each $1 \leq r \leq n-1$ it holds that*

$$\sum_{i=1}^{r} d_i \leq r(r-1) + \sum_{i=r+1}^{n} \min(r, d_i). \tag{1}$$

We call the characterization provided by Lemma 2 the *Erdős-Gallai test*. Unfortunately, there are k-anonymous sequences $\mathcal{D}'$, passing the Erdős-Gallai test, but still or not realizable in the input graph G (see Figure 2 for an example).

We thus designed an advanced version of the Erdős-Gallai test that takes the structure of the input graph into account. To explain the basic idea behind, we first discuss how Inequality (1) in Lemma 2 can be interpreted: Let V^r be the set of vertices corresponding to the first r degrees. The left-hand side sums over the degrees of all vertices in V^r. This amount has to be at most as large as the number of edges (counting each twice) that can be "obtained" by making V^r a clique ($r(r-1)$) and the maximum number of edges to the vertices in $V \setminus V^r$ (a degree-d_i vertex has at most $\min\{d_i, r\}$ neighbors in V^r). The reason why the Erdős-Gallai test might not be sufficient to determine whether a sequence can be realized in G is that it ignores the fact that some vertices in V^r might be already adjacent in G and it also ignores the edges between vertices in V^r and $V \setminus V^r$. Hence, the basic idea of our *advanced Erdős-Gallai test* is, whenever some of the vertices corresponding to the degrees can be uniquely determined, to subtract the corresponding number of edges as they cannot contribute to the right-hand side of Inequality (1).

While the difference between using just the Erdős-Gallai test and the advanced Erdős-Gallai test resulted in rather small differences for the lower bound (at most 10 edges), this small difference was important for some of our instances to succeed in Phase 2 and to optimally solve the instance. We believe that further improving the advanced Erdős-Gallai test is the best way to improve the rate of success in Phase 2.

Complete Strategy for Phase 1. With the above described restriction for realizable k-anonymous degree sequences, we finally arrive at the following problem for Phase 1, stated in the optimization form we solve:

Realizable k-Degree Sequence Anonymity (k-RDSA)
Input: A degree sequence $\mathcal{B}$ and an integer $k \in \mathbb{N}$.
Task: Compute all k-anonymous degree sequences $\mathcal{B}'$ such that $\mathcal{B}' \circledcirc \mathcal{B}$, $\|\mathcal{B}' \ominus \mathcal{B}\|$ is minimum, and $\mathcal{B}' \ominus \mathcal{B}$ is realizable.

Our strategy to solve k-RDSA is to iterate (for increasing solution size) through the solutions of k-DSA and run for each of them the advanced Erdős-Gallai test. Thus, we step-wisely increase the respective lower bound $\mathcal{B}' - \mathcal{B}$ until we arrive at some $\mathcal{B}'$ passing the test. Then, for each solution of this size we test in Phase 2 whether it is realizable (if so, then we found an optimal solution). If the realization in Phase 2 fails, then, for each such block sequence $\mathcal{B}'$, we compute how many degrees have to be "wasted" in order to get a realizable sequence. Wasting means to greedily increase some degrees in $\mathcal{B}'$ (while preserving k-anonymity) until the resulting degree sequence is realizable in the input graph. The cost $\mathcal{B}' - \mathcal{B}$ plus the amount of degrees needed to waste in order to realize $\mathcal{B}'$ is stored as an upper-bound. A minimum upper-bound computed in this way is the result of our heuristic.

Due to the power law degree distribution in social networks, the degree of most of the vertices is close to the average degree, thus one typically finds in such instances two large blocks B_i and B_{i+1} containing many thousands of vertices. Hence, "wasting" edges is easy to achieve by increasing degrees from B_i by one to B_{i+1} (this is optimal with respect to the Erdős-Gallai characterization). For the case that two such blocks cannot be found, as a fallback we also implemented a straightforward dynamic programming to find all possibilities to waste edges to obtain a realizable sequence.

Remark. We do not know whether the decision version of k-RDSA (find only one such solution $\mathcal{B}'$) is polynomial-time solvable and resolving this question remains as challenge for future research.

3.3 Phase 2: Realizing a k-Anonymous Degree Sequence

Let (G, k) be an instance of Degree Anonymization and let $\mathcal{B}$ be the block sequence of G. In Phase 1 a k-anonymization $\mathcal{B}'$ of $\mathcal{B}$ is computed such that $\mathcal{B}' \circledcirc \mathcal{B}$. In Phase 2, given G and $\mathcal{B}'$, the task is to decide whether there is a set S of edge insertions for G such that the block sequence of $G + S$ is equal to $\mathcal{B}'$. We call this the Degree Realization problem and first prove that it is NP-hard.

Theorem 1. Degree Realization *is NP-hard even on cubic planar graphs.*

We next present our heuristics for solving Degree Realization. First, we find a degree-vertex mapping, that is, for $\mathcal{D}' = d'_1, \ldots, d'_n$ being the degree sequence corresponding to $\mathcal{B}'$, we assign each value d'_i to a vertex v in G such

that $d'_i \geq \deg_G(v)$ and set $\mathrm{d}(v)$, the demand of v, to $d'_i - \deg_G(v)$. Second, we try to find, mainly by the local exchange heuristic, an edge insertion set S such that in $G + S$ the amount of incident new edges for each vertex v is equal to its demand $\mathrm{d}(v)$. The details in the proof of Theorem 1 indeed show that already finding a realizable degree-vertex mapping is NP-hard. This coincides with our experiments, as there the "hard part" is to find a good degree-vertex mapping and the local exchange heuristic is quite successful in realizing it (if possible). Indeed, we prove that "large" solutions can be always realized by it. As a first step for this, we prove that any demand function can be assumed to require to increase the vertex degrees at most up to $2\Delta^2$.

Lemma 3. *Any minimum-size k-insertion set for an instance of* Degree Anonymization *yields a graph with maximum degree at most $2\Delta^2$.*

Theorem 2. *A demand function d is always realizable by the local exchange heuristic in a maximum degree-Δ graph $G = (V, E)$ if $\sum_{v \in V} d(v) \geq 20\Delta^4 + 4\Delta^2$.*

4 Experimental Results

Implementation Setup. All our experiments are performed on an Intel Xeon E5-1620 3.6GHz machine with 64GB memory under the Debian GNU/Linux 6.0 operating system. The program is implemented in Java and runs under the OpenJDK runtime environment in version 1.7.0_25. The time limit for one instance is set to one hour per k-value and we tested for $k = 2$, 3, 4, 5, 7, 10, 15, 20, 30, 50, 100, 150, 200. After reaching the time limit, the program is aborted and the upper and lower bounds computed so far by the dynamic program for Phase 1 are returned. The source code is freely available.[2]

Real-World Instances. We considered the five social networks from the co-author citation category in the 10$^{\text{th}}$ DIMACS challenge [5].

We compared the results of our upper bounds against an implementation of the clustering-heuristic provided by Lu et al. [12] and against the lower bounds given by the dynamic program. Our algorithm could solve 26% of the instances to optimality within one hour. Interestingly, our exact approach worked best with the coPapersCiteseer graph from the 10$^{\text{th}}$ DIMACS challenge although this graph was the largest one considered (in terms of $n + m$). For all tested values of k except $k = 2$, we could optimally k-anonymize this graph and for $k = 2$ our upper bound heuristic is just two edges away from our lower bound. The coAuthorsDBLP graph is a good representative for the results on the DIMACS-graphs, see Table 1: A few instances could be solved optimally and for the remaining ones our heuristic provides a fairly good upper bound. One can also see that the running times of our algorithms increase (in general) exponentially in k. This behavior captures the fact that our dynamic program for Phase 1 iterates over all minimal solutions and for increasing k the number of these solutions increases dramatically. Our heuristic also suffers from the following effect: Whereas the maximum running

[2] http://fpt.akt.tu-berlin.de/kAnon/

Table 1: Experimental results on real-world instances. We use the following abbreviations: CH for clustering-heuristic of Lu et al. [12], OH for our upper bound heuristic, OPT for optimal value for the Degree Anonymization problem, and DP for dynamic program for the k-RDSA problem. If the time entry for DP is empty, then we could not solve the k-RDSA instance within one-hour and the DP bounds display the lower and upper bounds computed so far. If OPT is empty, then either the k-RDSA solutions could not be realized or the k-RDSA instance could not be solved within one hour.

		solution size			DP bounds		time (in seconds)		
graph	k	CH	OH	OPT	lower	upper	CH	OH	DP
coAuthorsDBLP	2	97	62		61	61	1.47	0.08	0.043
($n \approx 2.9 \cdot 10^5$,	5	531	321	317	317	317	1.41	0.29	26.774
$m \approx 9.7 \cdot 10^5$,	10	1,372	893		869	869	1.03	0.48	1.58
$\Delta = 336$)	100	21,267	15,050		10,577	11,981	1.13	885.79	
coPapersCiteseer	2	203	80		78	78	9.9	0.1	0.394
($n \approx 4.3 \cdot 10^5$,	5	998	327	327	327	327	10.32	0.19	0.166
$m \approx 1.6 \cdot 10^7$,	10	2,533	960	960	960	960	8.83	0.74	0.718
$\Delta = 1188$)	100	51,456	22,030	22,007	22,007	22,007	5.97	263.95	264.553
coPapersDBLP	2	1,890	1,747		950	1,733	11.28	2.13	
($n \approx 5.4 \cdot 10^5$,	5	9,085	8,219		4,414	8,121	10.66	28.83	
$m \approx 1.5 \cdot 10^7$,	10	19,631	17,571		9,557	17,328	9.95	149.56	
$\Delta = 3299$)	100	258,230			128,143	233,508	22.16		

time of the clustering-heuristic heuristic was one minute, our heuristic could solve 74% of the instances within one minute and did not finish within the one-hour time limit for 12% of the tested instances. However, the solutions produced by our upper bound heuristic are always smaller than the solutions provided by the clustering-heuristic, on average the clustering-heuristic results are 72% larger than the results of our heuristic.

Random Instances. We generated random graphs according to the model by Barabási–Albert [1] using the implementation provided by the AGAPE project [2] with the JUNG library[3]. Starting with $m_0 = 3$ and $m_0 = 5$ vertices these networks evolve in $t \in \{400, 800, 1200, \ldots, 34000\}$ steps. In each step a vertex is added and made adjacent to m_0 existing vertices where vertices with higher degree have a higher probability of being selected as neighbor of the new vertex. In total, we created 170 random instances.

Our experiments reveal that the synthetic instances are particular hard. For example, even for $k = 2$ and $k = 3$ we could only solve 14% of the instances optimal although our dynamic program produces solutions for Phase 1 in 96% of the instances. For higher values of k the results are even worse (for example zero exactly solved instances for $k = 10$). This indicates that the current lower bound provided by Phase 1 needs further improvements. However, the upper bound provided by our heuristic are not far away: On average the upper bound is 3.6% larger than the lower bound and the maximum is 15%. Further enhancing the advanced Erdős-Gallai test seem to be the most promising step towards closing this gap between lower and upper bound. Comparing our heuristic with

[3] http://jung.sourceforge.net/

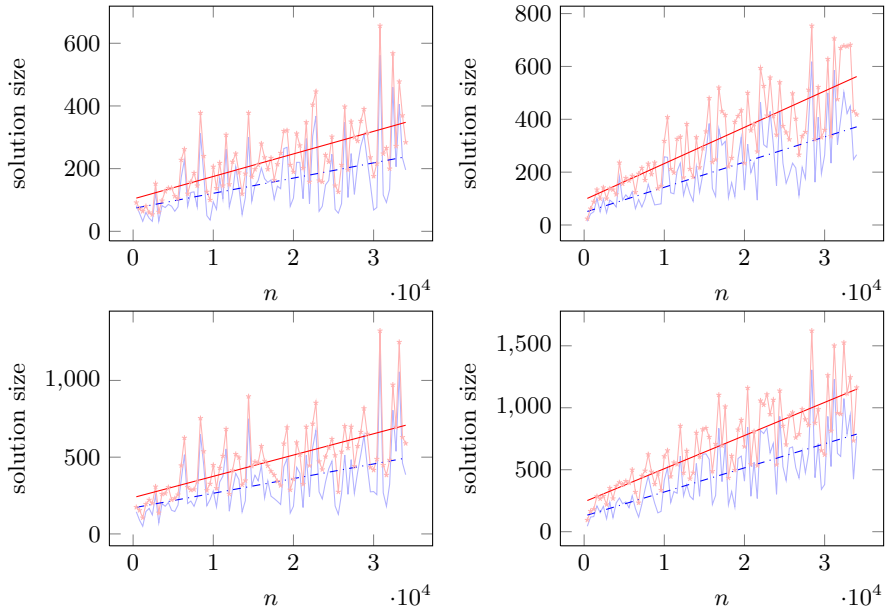

Fig. 3: Comparison of our heuristic (always the light blue line without marks) with the clustering-heuristic (always the light red line with little star as marks) on random data with different parameters: Top row is for $k = 2$, bottom row for $k = 3$; the left column is for $m_0 = 3$, and the right column for $m_0 = 5$. The linear, solid dark red line and dash-dotted blue line are linear regressions of the corresponding data plot. One can see that our heuristic produces always smaller solutions.

the clustering-heuristic reveal similar results as for real-world instances. Our heuristic always beats the clustering-heuristic in terms of solution size, see Figure 3 for $k = 2$ and $k = 3$. We remark that for larger values of k the running time of the heuristic increases dramatically: For $k = 30$ our algorithm provides upper bounds for 96% of the instances, whereas for $k = 150$ this value drops to 18%.

5 Conclusion

We have demonstrated that our algorithm framework is suitable to solve Degree Anonymization on real-world social networks. The key ingredients for this is an improved dynamic programming for the task to k-anonymize degree sequences together with certain lower bound techniques, namely the advanced Erdős-Gallai test. We have also demonstrated that the local exchange heuristic due to Liu and Terzi [11] is a powerful algorithm for realizing k-anonymous sequences and provided some theoretical justification for this effect.

The most promising approach to speedup our algorithm and to overcome its limitations on the considered random data, is to improve the lower bounds provided by the advanced Erdős-Gallai test. Towards this, and also to improve the respective running times, one should try to answer the question whether one

can find in polynomial-time a minimum k-anonymization $\mathcal{D}'$ of a given degree sequence $\mathcal{D}$ such that $\mathcal{D}' - \mathcal{D}$ is realizable.

Bibliography

[1] A. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[2] P. Berthomé, J.-F. Lalande, and V. Levorato. Implementation of exponential and parametrized algorithms in the AGAPE project. *CoRR*, abs/1201.5985, 2012.

[3] J. Casas-Roma, J. Herrera-Joancomartí, and V. Torra. An algorithm for k-degree anonymity on large networks. In *Proc. ASONAM'13*, pages 671–675. ACM Press, 2013.

[4] S. Chester, J. Gaertner, U. Stege, and S. Venkatesh. Anonymizing subsets of social networks with degree constrained subgraphs. In *Proc. ASONAM'12*, pages 418–422. IEEE Computer Society, 2012.

[5] DIMACS'12. Graph partitioning and graph clustering. 10th DIMACS challenge, 2012. URL `http://www.cc.gatech.edu/dimacs10/`. Accessed April 2012.

[6] D. Easley and J. Kleinberg. *Networks, Crowds, and Markets*. Cambridge University Press, 2010.

[7] P. Erdős and T. Gallai. Graphs with prescribed degrees of vertices (in Hungarian). *Math. Lapok*, 11:264–274, 1960.

[8] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys*, 42(4):14:1–14:53, 2010.

[9] S. Hartung, A. Nichterlein, R. Niedermeier, and O. Suchý. A refined complexity analysis of degree anonymization in graphs. In *Proc. 40th ICALP*, volume 7966 of *LNCS*, pages 594–606. Springer, 2013.

[10] S. Hartung, C. Hoffmann, and A. Nichterlein. Improved upper and lower bound heuristics for degree anonymization in social networks. *CoRR*, abs/1402.6239, 2014.

[11] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *Proc. SIGMOD '08*, pages 93–106. ACM, 2008.

[12] X. Lu, Y. Song, and S. Bressan. Fast identity anonymization on graphs. In *Proc. DEXA'12, Part I*, volume 7446 of *LNCS*, pages 281–295. Springer, 2012.

[13] P. Samarati. Protecting respondents identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[14] P. Samarati and L. Sweeney. Generalizing data to provide anonymity when disclosing information. In *Proc. PODS'98*, pages 188–188. ACM, 1998.

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

[16] B. Thompson and D. Yao. The union-split algorithm and cluster-based anonymization of social networks. In *Proc. 4th ASIACCS'09*, pages 218–227. ACM, 2009.

[17] A. Tripathi and S. Vijay. A note on a theorem of Erdös & Gallai. *Discrete Math.*, 265(1-3):417–420, 2003.

[18] B. Zhou and J. 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, 2011.

[19] B. Zhou, J. Pei, and W. Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *ACM SIGKDD Explorations Newsletter*, 10(2):12–22, 2008.