Foundations of computing Volume 2

André Nichterlein

Degree-Constrained Editing of Small-Degree Graphs

Universitätsverlag der TU Berlin

Technische Universität Berlin

André Nichterlein

Degree-Constrained Editing of Small-Degree Graphs

Die Schriftenreihe *Foundations of Computing* der Technischen Universität Berlin wird herausgegeben von:
Prof. Dr. Stephan Kreutzer,
Prof. Dr. Uwe Nestmann,
Prof. Dr. Rolf Niedermeier

Foundations of Computing | 02

André Nichterlein

Degree-Constrained Editing of Small-Degree Graphs

Universitätsverlag der TU Berlin

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Universitätsverlag der TU Berlin, 2015
http://www.verlag.tu-berlin.de

Fasanenstr. 88, 10623 Berlin
Tel.: +49 (0)30 314 76131 / Fax: -76133
E-Mail: publikationen@ub.tu-berlin.de

Zugl.: Berlin, Techn. Univ., Diss., 2014
1. Gutachter: Prof. Dr. Rolf Niedermeier
2. Gutachter: Prof. Dr. Matthias Müller-Hannemann
3. Gutachter: Prof. Dr. Dimitrios M. Thilikos
Die Arbeit wurde am 18. Dezember 2014 an der Fakultät IV unter Vorsitz von Prof. Dr. Sabine Glesner erfolgreich verteidigt.

Druck: docupoint GmbH
Satz/Layout: André Nichterlein

Umschlagfoto:
gabe popa | https://www.flickr.com/photos/gabepopa/14605341837 | CC BY 2.0
https://creativecommons.org/licenses/by/2.0/

ISBN 978-3-7983-2761-0 (print)
ISBN 978-3-7983-2762-7 (online)

ISSN 2199-5249 (print)
ISSN 2199-5257 (online)

Zugleich online veröffentlicht auf dem Digitalen Repositorium der Technischen Universität Berlin:
URN urn:nbn:de:kobv:83-opus4-65205
http://nbn-resolving.de/urn:nbn:de:kobv:83-opus4-65205

Zusammenfassung

Diese Dissertation beschäftigt sich mit Graphmodifikationsproblemen mit Knotengradbedingungen. Insbesondere wird die Berechnungskomplexität der Probleme DAG Realization und Degree Anonymity untersucht. Hierbei ist bei DAG Realization eine Multimenge an Paaren von natürlichen Zahlen gegeben und die Frage ist ob ein gerichteter kreisfreier Graph (DAG) existiert der die Multimenge realisiert. Das heißt, jedes Zahlenpaar ist genau einem Knoten des DAG zuzuordnen und dessen Knotengrad, bestehend aus Eingangs- und Ausgangsgrad, soll den Zahlen in dem zugeordneten Paar entsprechen. Die Zielstellung des Degree Anonymity-Problems ist, gegeben ein ungerichteter Graph G und zwei natürliche Zahlen k und s, in G höchstens s Modifikationsoperationen durchzuführen, sodass ein k-anonymer Graph entsteht. Ein Graph ist k-anonym, wenn für jeden Knoten $k-1$ andere Knoten mit dem gleichen Knotengrad enthalten sind.

Sowohl DAG Realization als auch Degree Anonymity werden in dieser Arbeit als NP-vollständig klassifiziert. Das heißt, es gibt vermutlich keine Polynomzeitalgorithmen die jede Eingabeinstanz der Probleme lösen können. Daher wird eine Studie der parametrisierten Berechnungskomplexität durchgeführt um effizient lösbare Spezialfälle zu identifizieren die noch praktisch relevant sind. Das Ziel hierbei ist die Entwicklung von *Festparameteralgorithmen* bei denen der vermutlich unvermeidliche exponentielle Anteil in der Laufzeit auf einen Parameter der Eingabe begrenzt wird. Ist der Parameter klein, so ist der entsprechende Festparameteralgorithmus schnell. Bei Degree Anonymity werden zwei Parameter in der Eingabe direkt mitgeliefert: die Anonymitätsstufe k und die Lösungsgröße s. Allerdings wird in dieser Arbeit gezeigt, dass Degree Anonymity W[1]-schwer bezüglich des Parameters s ist, selbst bei $k = 2$. Dies bedeutet, dass es sehr unwahrscheinlich ist, dass Festparameteralgorithmen für s und k existieren. Deshalb müssen andere Parameter untersucht werden.

Der Parameter maximaler Knotengrad stellt sich für beide Probleme, DAG Realization und Degree Anonymity, als vielversprechend heraus. Hierbei wird bei Degree Anonymity der Maximalgrad des Eingabegraphen benutzt. Bei DAG Realization wird der Maximalgrad in einem realisierendem DAG

genommen. Die Definition des Problems DAG Realization ermöglicht eine einfache Bestimmung des Parameters durch das Maximum aller Zahlen in der gegebenen Multimenge. Vorgestellt werden Festparameteralgorithmen bezüglich des Parameters Maximalgrad für die Probleme DAG Realization und Anonym E-Ins. Letzteres ist die Variante von Degree Anonymity bei der nur Kanteneinfügungen erlaubt sind. Die beiden Problemvarianten von Degree Anonymity, die nur Knoten- bzw. Kantenlöschungen erlauben, heißen Anonym V-Del bzw. Anonym E-Del und sind beide NP-vollständig in Graphen mit Maximalgrad sieben. Weiterhin bleiben Anonym V-Del und Anonym E-Del NP-vollständig auf stark eingeschränkten Graphklassen wie zum Beispiel Bäumen. Die Untersuchung der Approximierbarkeit von natürlichen Optimierungsvarianten von Anonym E-Del und Anonym V-Del ergibt ein ähnlich düsteres Bild, da keine der untersuchten Varianten in Polynomzeit besser approximiert werden kann als mit einem Faktor $n^{1/2}$, wobei n die Knotenanzahl ist. Diese Nichtapproximierbarkeit gilt für die Optimierungsprobleme bei denen die Lösungsgröße s vorgegeben ist und die Anonymitätsstufe k maximiert werden soll, selbst wenn eine in s exponentielle Laufzeit erlaubt wird.

Zu beachten ist, dass DAG Realization auch als nur Kanteneinfügungen erlaubendes Graphmodifikationsproblem mit Knotengradbedingungen angesehen werden kann: Ausgehend von einem kantenfreien Graphen ist die Aufgabe gerichtete Kanten einzufügen sodass ein realisierender DAG entsteht. Obiges Klassifizierungsresultat bezüglich des Parameters Maximalgrad zeigt, dass in Graphen mit kleinem Maximalgrad die Modifikationsoperation Kanteneinfügung einfacher ist als Knoten- oder Kantenlöschung. Dafür gibt es eine plausible Erklärung: In Graphen mit kleinem Maximalgrad gibt es einen hohen Freiheitsgrad wie Kanten eingefügt werden können, da für einen gegebenen Knoten so gut wie alle anderen Knoten als neuer Nachbar gewählt werden können. Durch die zusätzliche Einschränkung bei DAG Realization, dass der gerichtete Graph kreisfrei sein muss, wird dieser Freiheitsgrad wieder eingeschränkt. Bei Anonym E-Ins existieren keine Einschränkungen für den Freiheitsgrad. Tatsächlich kann dieser Freiheitsgrad auch in einer in dieser Arbeit angegebenen Implementierung ausgenutzt werden, die die entwickelten theoretischen Ideen in erfolgreiche Heuristiken und einen Algorithmus zur Berechnung unterer Schranken umsetzt. Experimente auf mehreren großen Datensätzen belegen, dass die angegebene Implementierung eine aktuelle Heuristik verbessert und auf 21 % (56 von 260) der Datensätze (beweisbar) optimale Lösungen liefert.

Abstract

This thesis deals with degree-constrained graph modification problems. In particular, we investigate the computational complexity of DAG Realization and Degree Anonymity. The DAG Realization problem is, given a multiset of positive integer pairs, to decide whether there is a realizing directed acyclic graph (DAG), that is, pairs are one-to-one assigned to vertices such that the indegree and the outdegree of every vertex coincides with the two integers of the assigned pair. The Degree Anonymity problem is, given an undirected graph G and two positive integers k and s, to decide whether at most s graph modification operations can be performed in G in order to obtain a k-anonymous graph, that is, a graph where for each vertex there are $k-1$ other vertices with the same degree.

We classify both problems as NP-complete, that is, there are presumably no polynomial-time algorithms that can solve every instance of these problems. Confronted with this worst-case intractability, we perform a parameterized complexity study in order to detect efficiently solvable special cases that are still practically relevant. The goal herein is to develop *fixed-parameter algorithms* where the seemingly unavoidable exponential dependency in the running time is confined to a parameter of the input. If the parameter is small, then the corresponding fixed-parameter algorithm is fast. The parameter thus measures some structure in the input whose exploitation makes the particular input tractable. Considering Degree Anonymity, two natural parameters provided with the input are anonymity level k and solution size s. However, we will show that Degree Anonymity is W[1]-hard with respect to the parameter s even if $k = 2$. This means that the existence of fixed-parameter algorithms for s and k is very unlikely. Thus, other parameters have to be considered.

We will show that the parameter maximum vertex degree is very promising for both DAG Realization and Degree Anonymity. Herein, for Degree Anonymity, we consider the maximum degree of the input graph. Considering DAG Realization, we take the maximum degree in a realizing DAG. Due to the problem definition, we can easily determine the maximum degree by taking the maximum over all integers in the given multiset. We provide fixed-parameter

algorithms with respect to the maximum degree for DAG Realization and for Anonym E-Ins. The later is the variant of Degree Anonymity when only edge insertions are allowed as modification operations. If we allow edge deletions or vertex deletions as graph modification operations, then we can show that the corresponding variants of Degree Anonymity—called Anonym V-Del and Anonym E-Del—are NP-complete even if the maximum vertex degree is seven. Moreover, we provide strong intractability results for Anonym E-Del and Anonym V-Del proving that they remain NP-complete in several restricted graph classes. Studying the approximability of natural optimization problems associated with Anonym E-Del or Anonym V-Del, we obtain negative results showing that none of the considered problems can be approximated in polynomial time better than within a factor of $n^{1/2}$ where n denotes the number of vertices in the input. Furthermore, for the optimization variants where the solution size s is given and the task is to maximize the anonymity level k, this inapproximability even holds if we allow a running time that is exponential in s.

Observe that DAG Realization also can be seen as degree-constrained graph modification problem where only arc insertions are allowed: Starting with an arcless graph, the task is to insert arcs to obtain a realizing DAG for the given multiset. The above classification with respect to the parameter maximum degree shows that in graphs with small maximum degree the modification operation edge respectively arc insertion is easier than vertex or edge deletion. There is a plausible explanation for this behavior: When the maximum degree is small, then there is a high freedom in inserting edges or arcs as for a given vertex almost all other vertices can be chosen as new neighbor. Observe that for DAG Realization the additional requirement that the directed graph shall be acyclic restricts this freedom. In Anonym E-Ins, we do not have restrictions on this freedom. In fact, exploiting this freedom in our implementation for Anonym E-Ins, we show that our theoretical ideas can be turned into successful heuristics and lower bounds. Experiments on several large-scale real-world datasets show that our implementation significantly improves on a recent heuristic and provides (provably) optimal solutions on about 21 % (56 of 260) of the real-world data.

Preface

This thesis summarizes part of my work as a research assistant at TU Berlin in the group of Prof. Rolf Niedermeier (from October 2010 until September 2014). The results presented in this thesis are partially contained in journal and conference publications that were produced in close cooperation with several coauthors. Below, I will explain which paper is contained in which chapter and discuss my contributions to the corresponding parts. Before doing this, I list further publications to which I contributed but that are not contained in this thesis: I worked on the parameterized complexity of tiling matrices [NDN11], of tabular data anonymization [BNN13, Bre+13b, Bre+14c], of graph anonymization by vertex addition [Bre+14b], of Metric Dimension [HN13], of 2-Club [HKN12, HKN13], of DAG Partitioning [Bev+13], of Shift Bribery [Bre+14a] and of Target Set Selection [Cho+14, Nic+13]; I further investigated the (parameterized) approximability of Target Set Selection [Baz+14a, Baz+14b] and Capacitated Arc Routing [Bev+14].

Chapter 4 is based on joint work with Sepp Hartung on the DAG Realization problem. I attended the talk of Annabell Berger (formerly at Universität of Halle-Wittenberg) at the *FCT'11* conference where she introduced the DAG Realization problem and presented her joint work with Matthias Müller-Hannemann (Universität of Halle-Wittenberg) [BM11]. Their main open question was whether the problem is polynomial-time solvable. We started to discuss at the conference about the problem and about their algorithmic strategies to attack the problem. When I was back at TU Berlin, I continued working on the problem and found an NP-completeness proof. At that point, my roommate Sepp joined the project and asked whether the problem may be fixed-parameter tractable with respect to the maximum degree. Answering this question took us about two months and resulted in a rather complicated algorithm classifying the problem as fixed-parameter tractable. I presented these results at the *8th International Conference on Computability in Europe (CiE 2012)* [HN12] where we won the best student paper award. We prepared a full version which additionally contains results about very dense and very sparse graphs (see

Section 4.4). It is currently under third review for *SIAM Journal on Discrete Mathematics*. I prepared the revised version which includes a shortening of the NP-completeness proof, new proofs for two crucial lemmas of the algorithm, and an overall unification and clarification of the used notation.

Chapter 5 is based on joint work with Cristina Bazgan (University Paris-Dauphine, LAMSADE, Paris), Robert Bredereck, Sepp Hartung, and Gerhard J. Woeginger (TU Eindhoven) on the Anonym V-Del and Anonym E-Del problems. Earlier work on Anonym E-Ins [Har+13] (which is presented in Chapter 6) motivated me to consider the vertex deletion variant of Anonym E-Ins. I presented the problem at our annual group-internal workshop in March 2013. Together with Robert, Sepp and, Gerhard, I quickly encountered the hardness of this problem. Robert discovered the reduction proving NP-completeness on trees and I found the reduction proving NP-completeness on bounded-degree graphs. We developed in close cooperation NP-completeness results on restricted graph classes, and discovered a few polynomial-time solvable and fixed-parameter tractable cases. I presented our results at the *24th International Symposium on Algorithms and Computation (ISAAC '13)* [Bre+13a].

In November 2013 and in March 2014 I visited Cristina at Université Paris-Dauphine. Motivated by the hardness of Anonym V-Del, I proposed to work on the parameterized approximability of Anonym V-Del and Anonym E-Del. We extended the reductions for Anonym V-Del to gap-reductions proving the inapproximability of Anonym V-Del and Anonym E-Del. I devised the fixed-parameter tractability result with respect to the combined parameter solution size and maximum degree. I also developed the fpt gap-reduction showing inapproximability for Anonym E-Del and the NP-completeness proof of Anonym E-Del on caterpillars. I contributed to the development of the fpt gap-reduction showing the parameterized intractability of Anonym V-Del. I presented the work at the *9th International Symposium on Parameterized and Exact Computation (IPEC '14)* [BN14]. I merged both papers mentioned above in Chapter 5 and unified the notation. A full journal version merging both papers is submitted to *Theoretical Computer Science*.

Chapter 6 is based on joint work with Vincent Froese, Sepp Hartung, Clemens Hoffmann, Rolf Niedermeier, and Ondřej Suchý (Czech Technical University in Prague). In summer 2012, Rolf gave me the paper of Liu and Terzi [LT08] who transferred the k-anonymity concept to graphs and introduced Anonym E-Ins. He proposed to work on this problem since we already did some research on the k-anonymity concept applied to tabular data [BNN13, Bre+13b, Bre+14c].

Together with Sepp, I studied the computational complexity of Anonym E-Ins and obtained in close cooperation the W[1]-hardness with respect to the parameter solution size when $k = 2$. This motivated us, again, to study the parameter maximum degree leading to the polynomial problem kernel. I encountered the connection with f-factors which simplified and improved the bound of the solution size in the maximum degree. During a visit in Berlin, Ondřej helped developing the polynomial problem kernel with respect to the combined parameter solution size and maximum degree. I presented our results at the *40th International Colloquium on Automata, Languages, and Programming (ICALP '13)* [Har+13]. A full version has been accepted by a special issue of the journal *Information and Computation* dedicated to selected papers from ICALP'13.

Motivated by this work, Rolf and I proposed to generalize the approach that led to the polynomial problem kernel. Together with Vincent, we started to study the Degree Constraint Editing (DCE) problem introduced by Mathieson and Szeider [MS12]. Answering one of their open questions, we transfered our ideas from Anonym E-Ins and developed a polynomial problem kernel for DCE when only allowing edge insertions. I devised the generalization of the ideas to a wide class of degree sequence completion problems and contributed to the other theoretical results. This thesis contains the part about the generalization to degree sequence completion problems. Vincent presented the paper at the *14th Scandinavian Workshop on Algorithm Theory (SWAT '14)* [FNN14]. A full journal version is submitted.

In our work on Anonym E-Ins, we proved that the heuristic two-phase approach of Liu and Terzi [LT08] produces optimal solutions when the solution size s is larger than some polynomial in Δ (more precisely when $s \geq (\Delta^2 + 4\Delta + 3)^2$). Since this bound does not look like being tight (a polynomial of degree four should not be the answer) and since I do not know how to improve the bound, I proposed to perform experiments to see whether optimal solutions can be provided even if the bound does not hold. In summer 2013, Clemens started his bachelor thesis on this topic and implemented the two-phase approach due to Liu and Terzi under the supervision of Rolf, Sepp, and me. During the implementation phase, Clemens encountered several problems and together we solved most of them. Since some of the solutions where far beyond what can be expected from a bachelor thesis, Sepp and I started to implement as well. While the basic idea on how to improve a dynamic program in phase one of the algorithm was proposed by Clemens, the details were tricky and thus Sepp implemented the first version of the dynamic program. I mainly developed the

Erdős-Gallai test and the linked wasting of costs and evaluated the experiments. Clemens concentrated on providing an implementation of the second phase. Sepp presented our practical and theoretical results at the *13th International Symposium on Experimental Algorithms (SEA '14)* [HHN14]. In preparation of a full journal version I improved the implementation of the first phase while Sepp improved phase two. Compared to the conference version I gave an extended description of our algorithms and conducted new experiments on a wider range of real-world networks. This extension of the data sets is the reason that we could solve in the conference version 26 % of the instances and now we can solve 21 % although our implementation became faster.

Acknowledgments. I would like to express my special appreciation to Rolf Niedermeier. I am grateful for the opportunity to work in his group, for his advice, encouragement, and patience. Furthermore, I want to thank all my former and current colleagues Nadja Betzler, René van Bevern, Robert Bredereck, Laurent Bulteau, Jiehua Chen, Michael Dom, Stefan Fafianie, Vincent Froese, Sepp Hartung, Falk Hüffner, Christian Komusiewicz, Stefan Kratsch, Rolf Niedermeier, Manuel Sorge, Ondřej Suchý, Nimrod Talmon, Johannes Uhlmann, Anh Quyen Vuong, Mathias Weller, and Gerhard J. Woeginger for creating a pleasant working atmosphere and for many fruitful discussions. Morover, I enjoyed working with my groupexternal coauthors Cristina Bazgan, Morgan Chopin, Piotr Faliszewski, Clemens Hoffmann, Geevarghese Philip, and Florian Sikora.

Contents

1. Introduction and Overview **1**
1.1. DAG Realization 4
1.2. Degree Anonymity 5

2. Preliminaries and Notation **13**
2.1. Graphs 13
2.2. Computational Complexity 20
2.3. Parameterized Complexity 21
2.4. Approximation 24

3. Basics on Degree Factors **29**
3.1. Realizable Degree Sequences 30
3.1.1. Realizing Undirected Graphs 31
3.1.2. Realizing Directed Graphs 35
3.2. f-Factors and More General Degree Factors 40
3.2.1. f-Factors 40
3.2.2. Generalizations of f-Factors 41

4. DAG Realization **47**
4.1. Introduction 47
4.2. NP-Completeness 50
4.3. Fixed-Parameter Tractability with Respect to Maximum Degree 58
4.3.1. General Terms and Observations 60
4.3.2. High-Potential Sequences 68
4.3.3. Low-Potential Sequences 77
4.4. Fixed-Parameter Tractability with Respect to $\binom{n}{2}-m$ and $m-n+1$ 83
4.5. Conclusion 89

5. Degree Anonymity by Vertex and Edge Deletion **91**
5.1. Introduction 91

5.2. Vertex Deletion 98
5.2.1. NP-Hardness on Trees 99
5.2.2. Generic Reduction 102
5.2.3. Inapproximability Results 108
5.2.4. Polynomially-Time Solvable Cases 112
5.3. Edge Deletion 116
5.3.1. NP-Hardness on Caterpillars 116
5.3.2. Inapproximability Results 119
5.4. Fixed-Parameter Tractable Cases 125
5.5. Conclusion 130

6. Degree Anonymity by Edge Insertion 131
6.1. Introduction 131
6.2. Computational Hardness 134
6.3. Polynomial Kernel for the Parameter Maximum Degree 138
6.3.1. The f-Factor Problem 139
6.3.2. A Polynomial-Time Algorithm for "Large"-Solution Instances 141
6.3.3. Polynomial Kernel 148
6.4. Fixed-Parameter Algorithm for the Parameter Maximum Degree 155
6.5. A General Approach for Degree Sequence Completion Problems . 157
6.5.1. Fixed-Parameter Tractability of Π-DSC 157
6.5.2. Applications of Degree Sequence Completion Problems . . 161
6.6. Upper and Lower Bounds for Degree Anonymity by Edge Insertion 162
6.6.1. General Framework Description 163
6.6.2. Phase 1: Exact k-Anonymization of Degree Sequences . . 167
6.6.3. Phase 2: Realizing a k-Anonymous Degree Sequence . . . 178
6.6.4. Further Upper-Bound Heuristics 183
6.6.5. Experimental Evaluation 186
6.7. Conclusion 198

7. Conclusion and Outlook 199

Bibliography 203

A. Complete List of Experimental Results 215

Chapter 1.

Introduction and Overview

Graph modification problems arise in many theoretical and practical settings, including computational biology, numerical algebra, and machine learning [BBC04, Sha02]. Their importance was underlined by a recent Dagstuhl seminar [BHL14] solely devoted to graph modification problems. The general question herein is, given a graph G, whether there is a graph that is "close" to G and has a desired property. Here, two graphs are considered being "close" to each other if one graph can be obtained from the other graph by a restricted number of modification operations. This leads to the following general problem.

Graph Modification
Input: A graph $G = (V, E)$, a graph property Π, and an integer $s \in \mathbb{N}$.
Question: Can G be transformed with at most s modification operations into a graph satisfying Π?

In this thesis, we consider the three probably most basic modification operations: vertex deletion, edge deletion, and edge insertion. Other modification operations have been studied as well; for example, vertex insertions [Bre+14b, Che+13a], edge contractions [GHP13, Gol+13], or edge editing [NSS01].

We concentrate on graph modification problems where the graph property Π depends on the vertex degrees. Observe that *degree-constrained graph modification problems* are still very general as they contain natural problems like Vertex Cover and Dominating Set: Given an undirected graph and an integer $h \in \mathbb{N}$, can one delete at most h vertices such that the degree of each remaining vertex is zero (Vertex Cover) or is decreased by at least one (Dominating Set)? Note that the degree constraints for these two problems have a *local* flavor: for checking whether a vertex in the goal graph satisfies the degree constraint, it is sufficient to know the degree of the particular vertex. The two problems that we study in this thesis, namely DAG Realization and Degree Anonymity (see Sections 1.1 and 1.2 for formal definitions),

have *global* degree constraints: checking whether the degree constraints are satisfied requires the knowledge of *all* vertex degrees in the graph. We show that both problems are NP-complete implying that, in general, one presumably cannot solve the problems efficiently. Following the approach of *parameterized algorithmics* [DF13, FG06, Nie06], we aim to solve relevant special cases where the input admits a certain structure. Whereas in classical complexity theory the running time of an algorithm is solely measured with respect to the overall input size, parameterized algorithmics additionally considers a *parameter*. This parameter is a measurement of some structure of the input and measuring different structures leads to different *parameterizations*. Examples for such parameters are the size of the sought solution or the maximum degree of the input graph. The general idea is that the corresponding *fixed-parameter algorithms* are fast if the parameters are small, that is, one searches for a small solution or the input graph admits a low maximum degree. However, such fixed-parameter algorithms do not necessarily exist and if they do, then the problem is called *fixed-parameter tractable* with respect to the particular parameter. In fact, we will show that Degree Anonymity is presumably fixed-parameter *intractable* with respect to the parameter solution size. On the positive side, a central result of this thesis is that Degree Anonymity as well as DAG Realization are fixed-parameter tractable with respect to the parameter maximum degree. This is in stark contrast to the known NP-completeness of other degree-constrained graph modification problems, like Vertex Cover or Dominating Set, on bounded degree graphs [GJ79].

Parameter maximum degree. Why is the parameter "maximum vertex degree Δ" of specific interest? First, as we consider *degree-constrained* graph modification problems, it is natural to ask whether the problems become tractable if the maximum degree is small. Second, from a parameterized complexity perspective it seems to be a "tight" parameterization in the sense that for only little "stronger" (that is, provably smaller [KN12]) parameters either the problem becomes intractable (in case of Degree Anonymity) or the fixed-parameter tractability status is open (in case of DAG Realization). Third, social networks, which are the input to Degree Anonymity, typically have few vertices with relatively high degree and many vertices of small degree. Leskovec and Horvitz [LH08] studied a huge instant-messaging network (180 million vertices)

with maximum degree 600. For the DBLP co-author graph[1] generated in February 2012 and containing more than 715,000 vertices we measured a maximum degree of 804 and an H-index of 208, that is, there are not more than 208 vertices with degree larger than 208. Thus, while the maximum degree Δ is too large for an exact algorithm with running time say $\mathcal{O}(2^{\Delta} \cdot (n+m))$, it is a reasonable parameter when searching for polynomial problem kernels (as we do in case of Degree Anonymity). Finally, the maximum degree is easy to compute.

Related work on graph modification problems. Considering vertex deletion, Lewis and Yannakakis [LY80] proved that for every hereditary graph property Π the corresponding vertex deletion problem is NP-hard; no such general result is known when considering edge deletion or insertion. There are numerous results about the complexity of edge modification problems [EC88, Gol+95, HS81, NSS01, Sha02, Yan81a, Yan81b]; we refer to Burzyn et al. [BBD06] for an overview. Under these results there is just a single non-trivial positive one: Hammer and Simeone [HS81] showed that Split Graph Editing (perform a minimum number of edge insertions and deletions to obtain an undirected graph that can be partitioned into a clique and an independent set) is polynomial-time solvable. Notably, this result relies on a characterization of split graphs by their *degree sequence*; thus, viewing the problem as degree-constrained graph modification problem with *global* degree constraints was the key to success. This positive result is complemented with NP-hardness results for over 20 other graph modification problems [BBD06]. Due to the abundance of NP-hardness results, graph modification problems have been intensively studied under the viewpoint of parameterized algorithmics [Cai96, Fel+11, Gra+04, KST99, Man08, MS12, NSS01, Per11, Sha02].

From a (parameterized) computational complexity perspective, the closest work we are aware of in terms of degree-constrained graph modification problems is due to Mathieson and Szeider [MS12]. In their basic model, each vertex is equipped with a degree list and the task is to edit the graph such that each vertex achieves a degree contained in its degree list. They studied the same modification operations as we do in this thesis: vertex deletion, edge deletion, and edge insertion. Mathieson and Szeider [MS12] achieved a general classification

[1] In this graph the vertices represent the authors and an edge indicates that the two corresponding authors are co-authors of at least one paper. The dataset and a corresponding documentation are available online (http://dblp.uni-trier.de/xml/).

result for these problems showing that they are W[1]-hard with respect to the parameter solution size but fixed-parameter tractable with respect to the combined parameter solution size and the maximum allowed degree. We refer to Chapter 3 for further details on degree-constrained graph modification problems and degree factor problems—a subclass of degree-constrained graph modification problems that are an important tool in this thesis.

1.1. DAG Realization

The Hamilton Cycle problem is one of the first problems classified as NP-complete by Karp [Kar72]. Given an undirected graph G, the question is whether G contains a *Hamiltonian cycle*, that is, a cycle covering all vertices of G exactly once. Reformulated as a graph modification problem, the question is whether edges can be removed such that the degree of each vertex is two *and* the remaining graph is *connected*. If the input graph is a clique, then Hamilton Cycle becomes trivial as each clique contains a Hamiltonian cycle. Degree-constrained graph modification problems where the input graph is a clique are called *graph realization* problems: Since any n-vertex graph is a subgraph of an n-vertex clique, the question is just whether there exist some graph realizing (that is satisfying) the degree constraints. Graph realization problems are well-understood and mostly polynomial-time solvable, see Section 3.1 for an overview.

The first problem that we study in this thesis adds the acyclicity constraint to the graph realization problem. It is known that the corresponding problem in the setting of undirected graphs is polynomial-time solvable, see Section 3.1.1 for details. For the directed setting, the complexity status was open. We thus consider in Chapter 4 the following graph realization problem due to Berger and Müller-Hannemann [BM11]:

DAG Realization

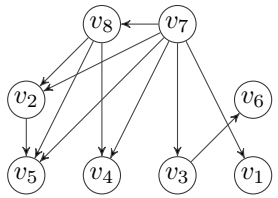
Input: A multiset $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n} \right\}$ of pairs of nonnegative integers.

Question: Is there a directed acyclic graph (without parallel arcs and self-loops) with vertex set $\{v_1, v_2, \ldots, v_n\}$ such that for every $v_i \in V$ the indegree is a_i and the outdegree is b_i?

Input:

$$\mathcal{S} = \left\{ \binom{1}{0}, \binom{2}{1}, \binom{1}{1}, \binom{2}{0}, \binom{3}{0}, \binom{1}{0}, \binom{0}{6}, \binom{1}{3} \right\}$$

Solution:



Related Work. Deciding whether a given degree sequence (a multiset of positive integers) is realizable by an *undirected graph* is polynomial-time solvable. There are characterizations for realizable degree sequences due to Erdős and Gallai [EG60] and algorithms by Havel [Hav55] and Hakimi [Hak62]. The problem variant where one asks whether there is a *directed* graph realizing the given degree sequence (a multiset of positive integer pairs) has also been intensively studied; see Chen [Che66], Fulkerson [Ful60], Gale [Gal57], and Ryser [Rys57] for characterizations of realizable degree sequences and Kleitman and Wang [KW73] for polynomial-time algorithms. We refer to Section 3.1 for a detailed overview on these results.

Berger and Müller-Hannemann [Ber11, BM11, BM12] investigated restricted variants of DAG Realization that are polynomial-time solvable and performed an extensive experimental study on the general problem.

Our contributions. We present two main results for DAG Realization: First, answering the open question of Berger and Müller-Hannemann [BM11], we show that DAG Realization is NP-complete. Second, we classify DAG Realization as fixed-parameter tractable with respect to the maximum degree $\Delta := \max\{a_1, b_1, a_2, b_2, \ldots, a_n, b_n\}$.

1.2. Degree Anonymity

For many scientific disciplines, including the understanding of the spread of diseases in a globalized world or power consumption habits with impact on fighting global warming, the availability of social network data becomes more and more important. To respect privacy issues, there is a strong demand to anonymize the associated data in a preprocessing phase [Fun+10]. If a graph

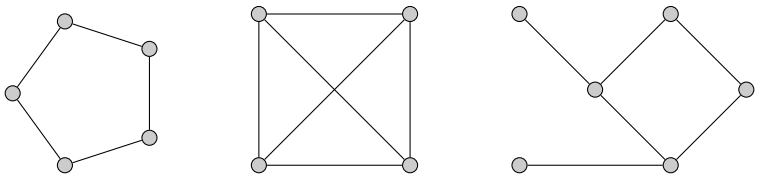
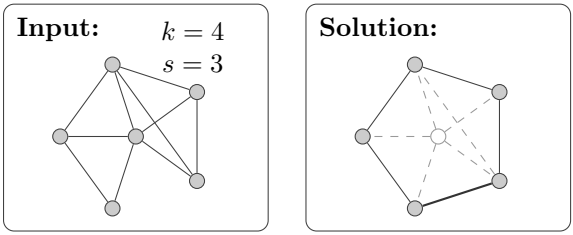


Figure 1.1.: The three displayed graphs are (from left to right) 5-anonymous, 4-anonymous, and 2-anonymous.

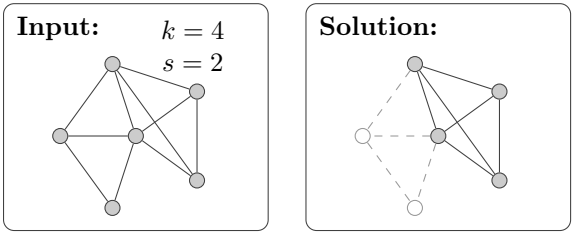
contains only few vertices with some distinguished feature, then this might allow the identification (and violation of privacy) of the underlying real-world entities with that particular feature. Hence, in order to ensure pretty good privacy and anonymity behavior, every vertex should share its feature with many other vertices. In a landmark paper, Liu and Terzi [LT08] (also see Clarkson et al. [CLT10] for an extended version) considered the vertex degrees as feature; see Wu et al. [Wu+10] for other features considered in the literature. Correspondingly, a graph is called k-anonymous if for each vertex there are at least $k-1$ other vertices of same degree, see Figure 1.1 for illustrations. Therein, different values of k reflect different privacy demands and the natural computational task arises to perform few changes to a graph in order to make it k-anonymous. This leads to the following degree-constrained graph modification problem.

Degree Anonymity (Anonym)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Can G be transformed with at most s modification operations into a k-anonymous graph $G' = (V', E')$, that is, for each vertex in G' there are $k-1$ other vertices of the same degree?

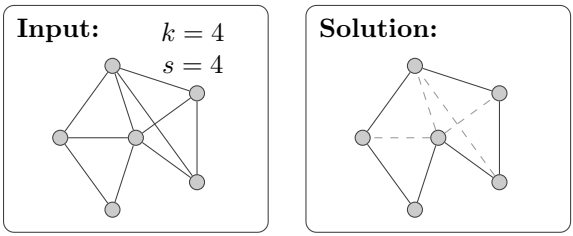


Observe that Degree Anonymity is, in contrast to DAG Realization, a *pure* degree-constrained graph modification problem, that is, besides the degree constraint "k-anonymity" no further restrictions are imposed on the resulting graph. As already mentioned in the beginning, we concentrate on the basic modification operations vertex deletion, edge deletion, and edge insertion. We consider only one of this three operations at once since the corresponding problems are already pretty hard in a computational sense. This yields the following three problems which we consider in Chapters 5 and 6. To this end, for a set S of vertices or edges, we denote by $G - S$ (by $G + S$) the graph that results from deleting (inserting) in G the vertices or edges contained in S.

Degree Anonymity by Vertex Deletion (Anonym V-Del)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Is there a vertex subset $S \subseteq V$ of size at most s such that $G - S$ is k-anonymous?

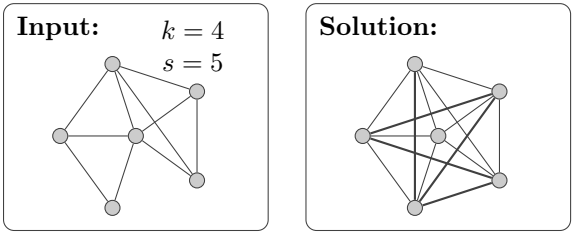


Degree Anonymity by Edge Deletion (Anonym E-Del)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Is there an edge subset $S \subseteq E$ of size at most s such that $G - S$ is k-anonymous?



In Chapter 5, we present strong intractability results for Anonym V-Del and for Anonym E-Del.

Degree Anonymity by Edge Insertion (Anonym E-Ins)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Is there an edge set $S \subseteq \binom{V}{2} \setminus E$ of size at most s such that $G+S$ is k-anonymous?



Notably, there is a very close connection between Anonym E-Del and Anonym E-Ins as a graph G is k-anonymous if and only if the complement graph $\overline{G}$ is k-anonymous. Hence, a given graph G can be made k-anonymous by inserting at most s edges if and only if $\overline{G}$ can be made k-anonymous by deleting at most s edges. Thus, most hardness results for Anonym E-Del transfer to Anonym E-Ins. However, since the complement graph operation does not preserve the maximum degree, the NP-completeness of Anonym E-Del on bounded-degree graphs (see Theorems 5.19 and 5.20) does not transfer to Anonym E-Ins. In fact, a central result Chapter 6 is that Anonym E-Ins is fixed-parameter tractable with respect to the parameter maximum degree.

Anonymization. Data anonymization is an active area of research with a considerable amount of published work. See, for example, the survey by Fung et al. [Fun+10]. The concept of k-anonymity was introduced for tabular data in databases [Fun+10, Sam01, SS98, Swe02]. While it is beyond the scope of this work to fully address all the potential weaknesses of the k-anonymity concept, we mention for the sake of completeness that it is known to be vulnerable against the attack models "attribute linkage", "table linkage", and "probabilistic attack" [Fun+10]. In particular, note that with "differential privacy" (cleverly adding some random noise) a "statistical model" has become very popular as well [Dwo11, Fun+10]; we do not study this here. Instead, we focus on a better understanding of the computational complexity and on tractable special cases of combinatorial data privacy problems in graphs.

Liu and Terzi [LT08] assumed in their model that an adversary (who wants to de-anonymize the network) knows only the degree of the vertex of a target

individual; this is a modest adversarial model. Clearly, there are stronger adversarial models which (in many cases very realistically) assume that the adversary has more knowledge, making it possible to breach the privacy provided by a "k-anonymized graph" [ALY11, NS09, Sal+11]. Notably, the differential privacy framework incurs other difficulties when applied to anonymizing graphs [NS09, Sal+11]. Moreover, it has been argued that graph anonymization has fundamental theoretical barriers which prevent a fully effective solution [ALY11]. In conclusion, given the generality of background knowledge an adversary may or may not have, graph anonymization remains a chimerical target [LSB12] and, thus, a universally best model for graph anonymization is not available. Degree Anonymity, however, provides the perhaps most basic and still practically relevant model; it is the subject of active research [Bre+14b, Che+12, Che+13a, CHT13, LSB12].

Related work. Our most important point of reference is Liu and Terzi's work [LT08] where the basic model of graph anonymization (by edge insertions) was introduced and sophisticated (heuristic) algorithms (also using algorithms to determine the realizability of degree sequences) have been developed and validated on experimental data. Somewhat more general models have been considered by Zhou and Pei [ZP11] (studying the neighborhood of vertices instead of only the degree) and by Chester et al. [Che+12] (anonymizing a *subset* of the vertices of the input). Lu et al. [LSB12] and Casas-Roma et al. [CHT13] proposed enhanced algorithms for Anonym E-Ins. Again, these algorithms are heuristic in nature. Today, the field of graph anonymization has grown tremendously with numerous surveys and research directions. We only discuss some directly related work.

Chester et al. [Che+13b] were among the few authors having performed formal computational complexity studies of Anonym E-Ins and edge-labeled variants. On the positive side, they showed a polynomial-time algorithm for the unlabeled case on bipartite graphs. In particular, they asked for effective polynomial-time approximation algorithms (for the optimization versions of the underlying decision problems) for the NP-hard variants and complain the lack of complexity investigations and theoretical research.

Chester et al. [Che+13a] and Bredereck et al. [Bre+14b] investigated the vertex insertion variant of Degree Anonymity. Here, the question is whether at most s vertices can be inserted to the input graph to make it k-anonymous; the inserted vertices can be made adjacent to all other (original and newly inserted)

Table 1.1.: Overview on the computational complexity classification of Anonym V-Del, Anonym E-Del, and Anonym E-Ins.

Parameter	Anonym V-Del	Anonym E-Del	Anonym E-Ins
k	NP-complete for $k = 2$	NP-complete for $k = 2$	NP-complete for $k = 2$
(s, k)	W[2]-hard	W[1]-hard	W[1]-hard
Δ	NP-complete for $\Delta = 3$	NP-complete for $\Delta = 7$	FPT
(s, Δ)	FPT	FPT	FPT
(k, Δ)	FPT	FPT	FPT

vertices. In this basic variant, the actual graph structure is not important, only the degrees of the input graph matter. This gives the problem a significantly different flavor when compared to the Degree Anonymity variants that we study in this thesis. Chester et al. [Che+13a] provided NP-completeness for the vertex labeled variant, and for the unlabeled variant they gave an algorithm that computes in $\mathcal{O}(nk)$ time a solution containing at most k vertices more than an optimal solution. Bredereck et al. [Bre+14b] showed NP-completeness for several restricted variants and presented fixed-parameter tractability results for the basic variant. However, the complexity status of the basic vertex insertion variant of Degree Anonymity remains open.

Finally, we mention in passing that there is recent work on studying the parameterized complexity of k-Anonymity on tabular data with numerous tractability and intractability results [BDVD11, Bon+13, Bre+14c, EWC09].

Our contributions. The field of graph anonymization is young and under strong development; there is very little research on its theoretical foundations, particularly concerning computational complexity and algorithms with provable performance guarantees [Che+12]. The following results are a first step towards closing the gap between theoretical and practical results.

In Chapter 5, we show that Anonym V-Del and Anonym E-Del are NP-complete, even on very special graph classes such as trees or bounded-degree graphs. We then analyze the parameterized complexity of Anonym V-Del and Anonym E-Del. Once again, both problems show a computationally difficult and challenging characteristic, see Table 1.1 for an overview: They

Table 1.2.: Overview on the inapproximability of the optimization variants associated with Anonym V-Del and Anonym E-Del. The results for Anonym E-Del also hold for Anonym E-Ins due to their strong connection via the complement graph.

vertex deletion running time	Anonym Min-V-Del (fixed k, minimize s)	Max-Anonym V-Del (fixed s, maximize k)
polynomial time	no $n^{1-\varepsilon}$-approximation	no $n^{1/2-\varepsilon}$-approximation
$f(s) \cdot n^{O(1)}$	open	no $n^{1/2-\varepsilon}$-approximation
edge deletion running time	Anonym Min-E-Del (fixed k, minimize s)	Max-Anonym E-Del (fixed s, maximize k)
polynomial time	no $n^{1-\varepsilon}$-approximation	no $n^{1-\varepsilon}$-approximation
$f(s) \cdot n^{O(1)}$	open	no $n^{1-\varepsilon}$-approximation

are W[1]-hard with respect to each of the three (single) parameters solution size s, anonymity level k, and maximum degree Δ, and with respect to the combined parameter (s, k). The only positive parameterized results come with the combined parameters (Δ, s) and (Δ, k). We then study the approximability of natural optimization problems associated with Anonym E-Del or Anonym V-Del. Partially answering an open question of Chester et al. [Che+13b], we give negative results showing that in polynomial time none of the considered problems can be approximated better than a factor of $n^{1/2}$. Here, n denotes the number of vertices in the input graph. Furthermore, for the optimization variants where the solution size s is given and the task is to maximize the anonymity level k, this inapproximability even holds if we allow a running time of $f(s)n^{\mathcal{O}(1)}$ for any computable f. Again, this result holds for both the edge deletion variant and the vertex deletion variant, see Table 1.2 for an overview.

Our findings for Anonym E-Ins, presented in Chapter 6, are more positive, see Table 1.1 for an overview and comparison to the vertex and edge deletion variants. Here, the central result is that Anonym E-Ins has a polynomial-size problem kernel when parameterized by the maximum vertex degree Δ of the input graph. In other words, we prove that there is a polynomial-time algorithm that transforms any input instance of Anonym E-Ins into an equivalent instance with at most $\mathcal{O}(\Delta^7)$ vertices. Indeed, we encounter a "win-win" situation when proving this result: We show that Liu and Terzi's heuristic strategy [LT08] finds an optimal solution when the size s of a minimum solution is larger than $2\Delta^4$.

Hence, either we can solve the problem in polynomial time or the solution size is "small". As a consequence, we can bound s in $\mathcal{O}(\Delta^4)$ and, hence, a polynomial kernel we provide for the combined parameter (Δ, s) actually is also a polynomial kernel only for Δ. While our kernelization directly implies fixed-parameter tractability for Anonym E-Ins parameterized by Δ, we also develop a further fixed-parameter algorithm with an improved worst-case running time.

We generalize the ideas behind the kernelization to further graph completion problems where the task is to insert edges so that the degree sequence of the resulting graph fulfills some prescribed property Π. Furthermore, we experimentally evaluate the usefulness of our theoretical results on the "win-win" situation. We present an enhancement of the heuristic due to Liu and Terzi [LT08], 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 21 % of the real-world data we provide (provably) optimal solutions, whereas in the other cases our solutions significantly improve on known heuristic solutions.

Chapter 2.

Preliminaries and Notation

In this chapter, we provide our notation and explain the employed theoretical concepts. As this thesis focuses on graph problems, we start in Section 2.1 by introducing the relevant notation for graphs. In Section 2.2 we recall basic concepts from classical complexity theory with a focus on NP and NP-hard problems. In Sections 2.3 and 2.4 we give a brief overview on two common approaches to deal with NP-hard problems: parameterized and approximation algorithms.

2.1. Graphs

This section provides our notation for graphs. For a broader overview, we refer to Diestel [Die10]. All graphs studied in this thesis are simple, that is, they contain no self-loops and no multi-edges. We study undirected as well as directed graphs. First, we give some basic definitions. Let $\mathbb{N} := \{0, 1, 2, \ldots\}$. For a set S we denote by 2^S the *power set* of S, that is the set of all subsets of S. We further denote by $\binom{S}{2}$ the set of all size-two subsets of S.

Undirected graphs. An undirected graph is a pair $G = (V, E)$. In this context, we denote by

V the *vertex set* of G;

E the *edge set* of G with $E \subseteq \binom{V}{2}$; for an edge $e = \{u, v\} \in E$ the two vertices u and v are called *endpoints* of e;

n_G the number $|V|$ of *vertices*;

m_G the number $|E|$ of *edges*;

$N_G(v)$	the *(open) neighborhood* of v, formally, $N_G(v) := \{u \in V \mid \{u, v\} \in E\}$;
$N_G[v]$	the *closed neighborhood* of v, formally, $N_G[v] := N_G(v) \cup \{v\}$;
$N_G(V')$	the set $\bigcup_{u \in V'} N_G(u)$ for $V' \subseteq V$;
$N_G[V']$	the set $N_G(V') \cup V'$ for $V' \subseteq V$;
$N_G^i[v]$	for $i \geq 1$ the *closed i^{th} neighborhood* of v, formally, $N_G^1[v] := N_G[v]$ and $N_G^i[v] := N_G(N_G^{i-1}[v])$ for $i \geq 2$;
$N_G^i(v)$	for $i \geq 1$ the *(open) i^{th} neighborhood* of v, formally, $N_G^1(v) := N_G(v)$ and $N_G^i(v) := N_G^i[v] \setminus N_G^{i-1}[v]$ for $i \geq 2$;
$\deg_G(v)$	the *degree* of v, formally, $\deg_G(v) := \lvert N_G(v) \rvert$;
Δ_G	the *maximum degree* of G, formally, $\Delta_G := \max_{v \in V}\{\deg_G(v)\}$;
δ_G	the *minimum degree* of G, formally, $\delta_G := \min_{v \in V}\{\deg_G(v)\}$;
$\mathcal{S}_G$	the *degree sequence* of G, that is, the multiset $\{\deg_G(v) \mid v \in V\}$;
$B_G(d)$	the *block* of degree d, that is, the set of all vertices with degree d in G, formally, $B_G(d) := \{v \in V \mid \deg_G(v) = d\}$;
$\mathcal{B}_G$	the *block sequence* of G, formally, $\mathcal{B}_G := (\lvert B_G(0) \rvert, \lvert B_G(1) \rvert, \lvert B_G(2) \rvert, \ldots, \lvert B_G(\Delta_G) \rvert)$;
$V(E')$	the set of of all endpoints of edges in $E' \subseteq \binom{V}{2}$, formally, $V(E') := \{v \in V \mid \{u, v\} \in E'\}$;
$E(V')$	the set of edges having both endpoints in $V' \subseteq V$, formally, $E(V') := E \cap \binom{V'}{2}$;
$\overline{G}$	the *complement graph* of G, formally, $\overline{G} := (V, \binom{V}{2} \setminus E)$;
$G[V']$	the subgraph of G induced by $V' \subseteq V$, formally, $G[V'] := (V', E(V'))$;
$G[E']$	the graph induced by $E' \subseteq \binom{V}{2}$, formally, $G[E'] := (V(E'), E')$;
$G + E'$	the graph obtained from G by inserting the edges $E' \subseteq \binom{V}{2} \setminus E$, formally, $G + E' := (V, E \cup E')$;

$G - E'$ the graph obtained from G by deleting the edges $E' \subseteq E$, formally, $G - E' := (V, E \setminus E')$;

$G - V'$ the graph obtained from G by deleting the vertices $V' \subseteq V$, formally, $G - V' := G[V \setminus V']$.

If the graph G is clear from the context, then we will omit the subscript G.

Directed graphs. A directed graph, also called *digraph*, is a pair $D = (V, A)$. We denote by

V the *vertex set* of D;

A the *arc set* of D with $A \subseteq V \times V$; for an arc $a = (u, v) \in E$ the vertex u is called the *tail* and v is called the *head* of a;

n_D the number $|V|$ of vertices;

m_D the number $|A|$ of arcs;

$D[V']$ the *induced subgraph* of D on $V' \subseteq V$, formally, $D[V'] := (V', E \cap (V' \times V'))$;

$\deg_D^+(v)$ the *outdegree* of v, formally, $\deg_D^+(v) := |\{(v, u) \in A\}|$;

$\deg_D^-(v)$ the *indegree* of v, formally, $\deg_D^-(v) := |\{(u, v) \in A\}|$;

$\mathcal{S}_D$ the *degree sequence* of D, that is, the multiset $\left\{ \binom{\deg_D^-(v)}{\deg_D^+(v)} \mid v \in V \right\}$;

Δ_D the *maximum degree* of D, formally, $\Delta_D := \max_{v \in V} \{\deg_D^+(v), \deg_D^-(v)\}$.

Remark: This is nonstandard notation. Usually the maximum degree in directed graphs is defined as the maximum over the sum of indegree and outdegree of each vertex.

Again, if the digraph D is clear from the context, then we will omit the subscript D.

Graph relations. Let $G = (V, E)$ be an undirected graph and $D = (V, A)$ be a digraph. The graph $G' = (V', E')$, respectively the digraph $D' = (V', A')$, is

a *supergraph* of G, respectively D, if $V' \supseteq V$ and $E' \supseteq E$, respectively $A' \supseteq A$;

a *subgraph* of G, respectively D, if $V' \subseteq V$ and $E' \subseteq E \cap \binom{V'}{2}$, respectively $A' \subseteq A \cap (V' \times V')$;

a *spanning subgraph* of G, respectively D, if $V' = V$ and $E' \subseteq E \cap \binom{V'}{2}$, respectively $A' \subseteq A \cap (V' \times V')$;

an *induced subgraph* of G, respectively D, if $V' \subseteq V$ and $E' = E \cap \binom{V'}{2}$, respectively $A' = A \cap (V' \times V')$;

isomorphic to G, respectively D, if there exists a bijection $f\colon V \to V'$ such that $\{u, v\} \in E$ if and only if $\{f(u), f(v)\} \in E'$, respectively $(u, v) \in A$ if and only if $(f(u), f(v)) \in A'$.

A path is a graph $G = (V, E)$ with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and edge set $E = \{\{v_i, v_{i+1}\} \mid 1 \leq i < n\}$. The vertices v_1 and v_n are called *endpoints*. A path on n vertices is denoted by P_n and has *length* $n - 1$. Two vertices u and v in the graph G are *connected* if G contains a path with endpoints u and v. A graph G is *connected* if every pair of vertices $u, v \in V$ is connected. A *connected component* is a maximal set of pairwise connected vertices.

The *H-index* of a graph G is the maximum integer h such that G has at least h vertices with degree at least h. As a consequence, if G has H-index h, then it has at most h vertices of degree larger than h.

The *underlying undirected graph* of a directed graph $D = (V, A)$ is the graph $G = (V, E)$ with $E := \{\{u, v\} \mid (u, v) \in A \vee (v, u) \in A\}$. Two vertices in a directed graph $D = (V, A)$ are called *connected* if they are connected in the underlying undirected graph. This is also called *weakly connected*. A *connected component* is a maximal set of pairwise connected vertices. This is also called *weakly connected component*. For two vertices $u, v \in V$ with $(u, v) \in A$, we call u an *inneighbor* of v and v an *outneighbor* of u. A vertex $v \in V$ is called a *source* if $\deg^-(v) = 0$ and it is called a *sink* if $\deg^+(v) = 0$.

Special graphs. A graph $G = (V, E)$ with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ is a

cycle if G is connected and each vertex has degree two, that is, the block sequence of G is $(0, 0, n)$. A cycle on n vertices is denoted by C_n.

three-regular graph	if the block sequence of G is $(0, 0, 0, n)$;
independent set	if G contains no edge, that is, the block sequence of G is (n);
clique	if the complement graph of G is an independent set, that is, the degree sequence of G is $\{n-1, n-1, \ldots, n-1\}$;
bipartite graph	if V can be partitioned into two independent sets;
three-colorable graph	if V can be partitioned into three independent sets;
planar	if G can be embedded into the plane (drawn with points for vertices and curves for edges) without crossing edges;
forest	if G contains no cycle;
tree	if G is connected and a forest;
caterpillar	if G is a tree that has a dominating path. A *dominating path* is a path in G such that each vertex not contained in the path has a neighbor in the path;
$\mathcal{F}$*-free*	if G does not contain any graph of the set $\mathcal{F}$ as induced subgraph;
cluster graph	if every connected component is a clique, or equivalently, if G is $\{P_3\}$-free;
cograph	if G is $\{P_4\}$-free;
split graph	if V can be partitioned into a clique and an independent set;
pseudo-split graph	if G is $\{2P_2, C_4\}$-free;
trivially perfect graph	if G is $\{P_4, C_4\}$-free;
threshold graph	if there are positive real vertex weights $w(v)$ for $v \in V$ such that $\{v_i, v_j\} \in E$ if and only if $w(v_i) + w(v_j) \geq 1$;
interval graph	if there are real intervals $\{I_v \mid v \in V\}$ such that $\{v_i, v_j\} \in E$ if and only if $I_{v_i} \cap I_{v_j} \neq \emptyset$;

unit interval graph if G is an interval graph where every interval has unit length;

permutation graph if there is a permutation σ of the numbers $1, 2, \ldots, n$, such that $\{v_i, v_j\} \in E$ if and only if $i < j$ and $\sigma(i) > \sigma(j)$;

bipartite permutation if G is a bipartite graph and a permutation graph.

Equivalent definition: A bipartite permutation graph is a bipartite graph not containing an asteroidal triple (in other words, it is AT-free). Three vertices of a graph form an *asteroidal triple* if every two of them are connected by a path avoiding the neighborhood of the third.

unigraph if G is determined by its degree sequence up to isomorphism.

A digraph $D = (V, A)$ with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ is a

directed path if $A = \{(v_i, v_{i+1}) \mid 1 \leq i < n\}$;

directed cycle if $A = \{(v_i, v_{i+1}) \mid 1 \leq i < n\} \cup \{v_n, v_1\}$;

independent set if D contains no edge;

directed acyclic graph (DAG) if D does not contain a directed cycle.

Each DAG D admits a *topological ordering*, that is, an ordering of all its vertices $v_1, v_2, \ldots, v_n$ such that for all arcs $(v_i, v_j) \in A$ it holds that $i < j$.

tournament if the underlying undirected graph of D is a clique;

transitive tournament if D does not contain a directed cycle and is a tournament.

Equivalent definition: a tournament is a digraph with degree sequence $\left\{\binom{0}{n-1}, \binom{1}{n-2}, \ldots, \binom{n-1}{0}\right\}$.

Degree factors and degree sequences. A *degree factor* in a graph $G = (V, E)$ is a spanning subgraph of G satisfying some predefined degree conditions. Given a function $f\colon V \to \mathbb{N}$, an *f-factor* in a graph G is a spanning subgraph G' of G such that $\deg_{G'}(v) = f(v)$ for all $v \in V$.

A *degree sequence* is a multiset of nonnegative integers or a multiset of nonnegative integer pairs, depending on whether we search for undirected or directed graphs. Although in a multiset no ordering is given, we stick to the term degree *sequence* as it is commonly used in the literature. The elements of the degree sequence are called *degrees*. If there exists for a degree sequence $\mathcal{S}$ a graph G (digraph D) with degree sequence $\mathcal{S}$, then $\mathcal{S}$ is called *realizable* and G (D) is called *realization of* $\mathcal{S}$ or *realizing graph*. For a graph property Π a degree sequence $\mathcal{S}$ is called *(potentially)* Π*-realizable* if there exists a realization of $\mathcal{S}$ having property Π. A degree sequence $\mathcal{S}$ is called *forcibly* Π*-realizable* if $\mathcal{S}$ is realizable and all realizations of $\mathcal{S}$ have property Π.

A *block sequence* is a tuple of nonnegative integers (we define block sequences only for undirected graphs). The block sequence $\mathcal{B}_{\mathcal{S}}$ of a degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ with maximum degree $\Delta := \max\{d_i\}$ is defined as $\mathcal{B}_{\mathcal{S}} := (|\{i \mid d_i = 0\}|, |\{i \mid d_i = 1\}|, \ldots, |\{i \mid d_i = \Delta\}|)$. Observe that if the degree sequence $\mathcal{S}$ is realizable and G is one of its realizations, then $\mathcal{B}_{\mathcal{S}}$ is the block sequence of G. Conversely, the degree sequence $\mathcal{S}_{\mathcal{B}}$ of a block sequence $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ is the multiset that contains b_d times the number d for each $1 \leq d \leq \Delta$. If $\mathcal{S}$ respectively $\mathcal{B}$ is clear from the context, then we omit the subscript. Let $\mathcal{S} = \{d_1, \ldots, d_n\}$ and $\mathcal{S}' = \{d'_1, \ldots, d'_n\}$ be two degree sequences with corresponding block sequences $\mathcal{B}$ and $\mathcal{B}'$. We define $\|\mathcal{B}\| := |\mathcal{S}| := \sum_{i=1}^{n} d_i$. We write $\mathcal{S}' \geq \mathcal{S}$ 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 "$\mathcal{S}'$ can be obtained from $\mathcal{S}$ by increasing some values". If $\mathcal{S}' \geq \mathcal{S}$, then (for sorted degree sequences) we define the degree sequence $\mathcal{S}' - \mathcal{S} = \{d'_1 - d_1, \ldots, d'_n - d_n\}$ and set $\mathcal{B}' \ominus \mathcal{B}$ to be its block sequence.

k-anonymous graphs. A graph $G = (V, E)$ is *k-anonymous* if all blocks of G contain zero or at least k vertices, that is, for each vertex $v \in V$ there are $k - 1$ other vertices having the same degree as v. We call a set of edges $S \subseteq \binom{V}{2} \setminus E$ an *edge insertion set* for G. If $G + S$ is k-anonymous, then S is called *k-insertion set* for G. Analogously, a vertex subset $S \subseteq V$ (an edge subset $S \subseteq E$) is called *k-deletion set* if $G - S$ is k-anonymous.

2.2. Computational Complexity

In this section, we provide a brief overview on classical complexity theory with focus on P, NP, and NP-complete problems. The purpose of this theory is to get a better understanding of the worst-case-*difficulty* to solve certain *decision problems*. Given a finite alphabet Σ, the decision problem associated with a language $L \subseteq \Sigma^*$ is to decide whether a given word $x \in \Sigma^*$ belongs to L. If the answer is yes, then we refer to x as *yes-instance*, otherwise as *no-instance*. Usually, we use the binary alphabet $\Sigma = \{0, 1\}$.

Example 1. Consider the DAG Realization problem.

DAG Realization
Input: A degree sequence $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n} \right\}$.
Question: Is there a directed acyclic graph realizing $\mathcal{S}$?

Denoting the binary encoding of a multiset S of nonnegative integer pairs by $\langle S \rangle$, the associated language L is defined by

$$L := \{\langle \mathcal{S} \rangle \mid \mathcal{S} \text{ is a realizable as dag}\} \subseteq \{0, 1\}^*.$$

To distinguish between tractable and intractable problems one typically uses the two complexity classes P and NP. Formally, the class P contains all problems that can be solved in *polynomial time* on a *deterministic* Turing machine. All problems that can be solved on a *nondeterministic* Turing machine in polynomial time are contained in the class NP. Although the question "P = NP?" is probably the most famous open problem in computer science, it is widely conjectured that the answer is P $\neq$ NP. In this way, P is considered to contain the *tractable* problems whereas the "hardest" problems in NP are considered *intractable*. To formally define what a "hardest" problem in NP is, we need the concept of *polynomial-time many-one reductions*.

Definition 2.1. Let $A, B \subseteq \Sigma^*$ be two problems. A *polynomial-time many-one reduction* from A to B is a polynomial-time computable function $f \colon \Sigma^* \to \Sigma^*$ such that for all $x \in \Sigma^*$ it holds that $x \in A \iff f(x) \in B$.

A problem B is called *NP-hard* if for each problem A in NP there is a polynomial-time many-one reduction from A to B. If it further holds that $B \in$ NP, then B is called *NP-complete*. Thus, all NP-complete problems are considered to be intractable. We refer to Garey and Johnson [GJ79] for a list of NP-complete problems and more details on P, NP, and NP-completeness.

To cope with NP-hard problems several approaches have been developed. In the next two sections, we describe two of them: parameterized and approximation algorithms.

2.3. Parameterized Complexity

The concept of parameterized complexity was pioneered by Downey and Fellows [DF13] (see Flum and Grohe [FG06] and [Nie06] for further monographs on parameterized complexity). The basic idea herein is to obtain a more "fine-grained" view on the problems. The motivation is that in practice the input is often restricted in the sense that it has *some* structure. For example, when anonymizing graphs, the input is usually some kind of *social network*. Social networks typically have small diameter ("small world" phenomenon) and admit a power-law degree distribution [New10]. Exploiting such structural properties is the goal of fixed-parameter algorithms.

To formally capture the additional structure, usually measured as a positive integer, we extend the input by one dimension.

Definition 2.2. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \Sigma^*$. For an instance $(I, p) \in \Sigma^* \times \Sigma^*$ the part I is called *input* and p is called the *parameter*.

This definition is general enough to allow the specification of *combined parameters* such that more than one structure of the input can be analyzed at once; this is a more recent development called multivariate complexity analysis [FJR13, KN12, Nie10]. For simplicity, one can think in the following of the parameter as a positive integer. However, all concepts work with combined parameters as well.

Given the additional structure(s) in the parameter, we classify problems in terms of the parameter and the input size. The counterpart of polynomial-time solvability in classical complexity theory is *fixed-parameter tractability* defined as follows.

Definition 2.3. A parameterized problem is called *fixed-parameter tractable* if there exists an algorithm that decides any instance (I, p) in $f(p) \cdot |I|^{\mathcal{O}(1)}$ time, for some computable function f solely depending on p.

The analog of the complexity P is the class FPT containing all fixed-parameter tractable problems. Observe that for constant parameter value p fixed-parameter

tractability implies polynomial-time solvability where the degree of the polynomial is *independent* of p (e. g. $\mathcal{O}(2^p \cdot n^2)$). If the degree of the polynomial depends on the parameter p (e. g. $\mathcal{O}(n^p)$), then the problem is contained in the complexity class XP.

Parameterization. When considering a new problem from the parameterized complexity perspective, a first question is: What is the parameter? The "standard parameter" is the size of a sought solution and we also consider this parameter in our study of Degree Anonymity. In the DAG Realization problem, however, the solution size (the size of the realizing DAG) is the same as the input size. Since the purpose of parameterized complexity is to consider more than just the input size, different parameters have to be considered. In this thesis, the parameter maximum degree plays a central role and the main fixed-parameter tractability results are obtained with respect to this parameter.

Kernelization. A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by data reduction, called *kernelization*. It is well-known that a decidable parameterized problem is fixed-parameter tractable if and only if it has a kernelization [Bod09, GN07, Kra14]. Here, the goal is to transform a given problem instance (I, p) in polynomial time into an equivalent instance (I', p') whose size is upper-bounded by a function of p.

Definition 2.4. Let $A \subseteq \Sigma^* \times \Sigma^*$ be a parameterized problem. A *problem kernelization* is an algorithm that takes as input an instance $(I, p) \in \Sigma^* \times \Sigma^*$ and outputs in time polynomially bounded in $|I| + p$ an instance (I', p') such that

(i) $|I'| \leq f(p)$ and $p' \leq f(p)$ for some computable function f, and

(ii) $(I, p) \in A \iff (I', p') \in A$.

The instance (I', p') is called a *problem kernel* and f denotes the *size* of the problem kernel. If f is a polynomial, then the instance (I', p') is called a *polynomial problem kernel*. We call the Property (ii) the *correctness* of the data reduction and say that (I, p) and (I', p') are *equivalent* instances.

Parameterized intractability. Parameterized complexity theory also provides a hardness program that allows to show presumable fixed-parameter *intractability*. The counterpart to NP is the W-hierarchy:

$$\mathrm{FPT} \subseteq \mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \ldots \mathrm{W}[t] \subseteq \ldots \subseteq \mathrm{W}[P] \subseteq \mathrm{XP}.$$

As in the classical setting, it is not known whether FPT = W[1] but it is believed that the above inclusions are strict.

For $t \geq 1$, one can show W[t]-hardness of a parameterized problem A by providing a *parameterized reduction* from a W[t]-hard problem to A.

Definition 2.5. Let $A, A' \subseteq \Sigma^* \times \Sigma^*$ be two parameterized problems. A *parameterized reduction* from A to A' is a function $f\colon \Sigma^* \times \Sigma^* \to \Sigma^* \times \Sigma^*$ such that for all $(I, p) \in \Sigma^* \times \Sigma^*$ it holds that

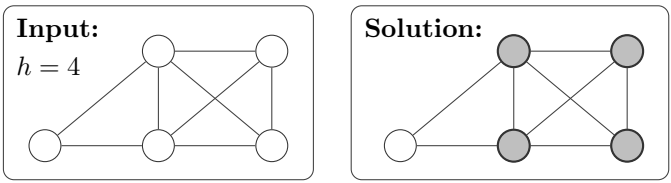
(i) $f((I,p)) = (I', p')$ can be computed in $g(p) \cdot |I|^{\mathcal{O}(1)}$ time for some computable function g,

(ii) $p' \leq h(p)$ for some computable function h, and

(iii) $(I,p) \in A \iff f((I,p)) \in A'$.

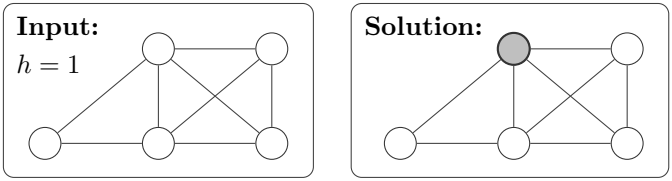
Clique is W[1]-complete with respect to the parameter solution size h.

Clique [GJ79, GT19]
Input: An undirected graph $G = (V, E)$ and an integer $h \in \mathbb{N}$.
Question: Is there a subset $V' \subseteq V$ of at least h pairwise adjacent vertices?



Dominating Set is W[2]-complete with respect to the parameter solution size h.

Dominating Set [GJ79, GT2]
Input: An undirected graph $G = (V, E)$ and an integer $h \in \mathbb{N}$.
Question: Is there a subset $V' \subseteq V$ of size at most h such that $N[V'] = V$?



2.4. Approximation

Another approach of dealing with NP-hard problems are approximation algorithms. Instead of relaxing the time-constraints as in parameterized algorithmics the idea is to find solutions that are not necessarily optimal but *close* to being optimal. This approach requires to consider *optimization problems* instead of decision problems. For example, one optimization variant for Anonym E-Del is the following.

Anonym Min-E-Del
Input: An undirected graph $G = (V, E)$ and an integer $k \in \mathbb{N}$.
Task: Compute a minimum-size k-deletion set $E' \subseteq E$.

Another optimization variant would be to give a graph G and an integer $s \in \mathbb{N}$ and to ask for a set of edge deletions that maximizes the anonymity level in the resulting graph.

Following Ausiello et al. [Aus+99] and Marx [Mar08] we formalize this concept as follows.

Definition 2.6. An optimization problem Q is a quadruple $(\mathcal{I}, \mathrm{sol}, \mathrm{cost}, \mathrm{goal})$ where

1. $\mathcal{I} \subseteq \Sigma^*$ is a set of instances;
2. for an instance $I \in \mathcal{I}$, $\mathrm{sol}(I)$ is the set of *feasible solutions* of I. The length of each $x \in \mathrm{sol}(I)$ is polynomially bounded in $|I|$, and it can be decided in polynomial time whether $x \in \mathrm{sol}(I)$ holds for given I and x;
3. given an instance I and a feasible solution x, $\mathrm{cost}(I, x)$ is a polynomial-time computable positive integer;
4. $\mathrm{goal} \in \{\min, \max\}$.

For a given instance I the task is to find a feasible solution x that achieves the best objective value, that is, we want x to satisfy

$$\mathrm{cost}(I, x) = \mathrm{goal}\{\mathrm{cost}(I, x') \mid x' \in \mathrm{sol}(I)\}.$$

Observe that any optimization problem Q has an associated decision problem P_Q. If goal = min, then the decision problem is given the instance I and an integer h to decide whether there exists a feasible solution in $\mathrm{sol}(I)$ of size at most h. If P_Q is NP-complete, then we cannot find an optimal feasible solution in polynomial time, unless P = NP. Thus we seek for feasible solutions that are not far away from the optimum solution. To this end, let $\mathrm{opt}(I)$ denote the cost of an optimal solution for the instance I. For a feasible solution $x \in \mathrm{sol}(I)$ the *performance ratio* (or *approximation factor*) R is defined as

$$R(I, x) := \begin{cases} \mathrm{cost}(I, x)/\,\mathrm{opt}(I), & \text{if goal} = \min \\ \mathrm{opt}(I)/\,\mathrm{cost}(I, x), & \text{if goal} = \max\,. \end{cases}$$

Note that $R(I, x) \geq 1$ for any instance I and feasible solution x and the closer $R(I, x)$ is to one, the better is the solution x. For a function $\rho\colon \mathbb{N} \to \mathbb{N}$, a $\rho(|I|)$-*approximation algorithm* returns for every instance $I \in \mathcal{I}$ a solution x such that $R(I, x) \leq \rho(|I|)$.

Note that this notion of approximability imposes no requirements on the running time so far. Of course the focus is mostly on polynomial time, see the textbooks of Vazirani [Vaz01] and Williamson and Shmoys [WS11] for a thorough overview on polynomial-time approximation algorithms. There is, however, also the relatively new approach of combining approximation and fixed-parameter algorithms [Cai08, CGG06, DFM06, Mar08]. Here, parameterized optimization problems are a natural extension of optimization problems that are obtained by replacing in Definition 2.6 (1) the term $\mathcal{I} \subseteq \Sigma^*$ by $\mathcal{I} \subseteq \Sigma^* \times \Sigma^*$ in order to incorporate the parameter. A *fixed-parameter approximation algorithm* for parameterized optimization problem is an approximation algorithm that runs for any instance (I, p) in $f(p) \cdot |I|^{\mathcal{O}(1)}$ time.

Inapproximability. This thesis contains besides exact algorithms (that have an approximation factor of one) no approximation algorithm but only inapproximability results. To show that a problem admits no $\rho(n)$-approximation algorithm, we employ the notion of a *gap*-reduction due to Arora and Lund [AL96].

Definition 2.7. A decision problem A is called *gap-reducible* to a maximization problem Q with gap $\rho \geq 1$ if there exists a polynomial-time computable function that maps any instance I of A to an instance I' of Q, while satisfying the following properties:

(i) if I is a yes-instance, then $\mathrm{opt}(I') \geq \xi(|I'|) \cdot \rho(|I'|)$, and

(ii) if I is a no-instance, then $\mathrm{opt}(I') < \xi(|I'|)$,

where ξ and ρ are two computable functions.

The definition for minimization problems is completely analogous. The purpose of gap-reductions is to transfer intractability results from classical complexity theory to approximation complexity.

Lemma 2.1 (Arora and Lund [AL96])**.** *If a decision problem A is NP-complete and gap-reducible to an optimization problem Q with gap ρ, then Q is not ρ-approximable in polynomial time, unless $P = NP$.*

Applying gap-reductions to fixed-parameter approximation algorithms requires a slight modification of the definition since we have to take care of the parameter.

Definition 2.8. A parameterized problem A is called *fpt gap-reducible* to a parameterized maximization problem Q with gap $\rho \geq 1$ if any instance (I, p) of A can be mapped to an instance (I', p') of Q in $f(p) \cdot |I|^{\mathcal{O}(1)}$ time while satisfying the following properties:

(i) $p' \leq g(p)$ for some computable function g,

(ii) if I is a yes-instance, then $\mathrm{opt}(I') \geq \xi(|I'|) \cdot \rho(|I'|)$, and

(iii) if I is a no-instance, then $\mathrm{opt}(I') < \xi(|I'|)$,

where ξ and ρ are two computable functions.

Again, the definition for minimization problems is completely analogous. With Definition 2.8, we can transfer the intractability results from parameterized complexity theory to fixed-parameter approximation algorithms.

Lemma 2.2. *If a parameterized problem A is $\mathcal{C}$-hard, fpt gap-reducible to a parameterized optimization problem Q with gap ρ, and Q is ρ-approximable in fpt-time, then $FPT = \mathcal{C}$, where $\mathcal{C}$ is any class of the W-hierarchy.*

Proof. We give a fixed-parameter algorithm for the parameterized problem A as follows: Since A is fpt gap-reducible to $Q = (\mathcal{I}, \mathrm{sol}, \mathrm{cost}, \max)$ with gap ρ, there exists an algorithm mapping the input (I, p) of A to an instance $(I', p') \in \mathcal{I}$ of Q in $f(p) \cdot |I|^{\mathcal{O}(1)}$ time such that the properties (i) to (iii) of Definition 2.8 are satisfied. We then apply the fixed-parameter ρ-approximation algorithm for Q on the instance (I', p'). Due to property (i), this algorithm runs in $g(p) \cdot |I|^{\mathcal{O}(1)}$ time for some computable function g. Let $x \in \mathrm{sol}(I')$ be the solution produced by the fixed-parameter ρ-approximation algorithm for Q. Assume that (I, p) was a no-instance. Hence, we have $\mathrm{cost}(x) \leq \mathrm{opt}(I')$ and by property (iii) it follows that $\mathrm{cost}(x) < \xi(I')$. Now assume that (I, p) was a yes-instance. Hence, we have $\mathrm{opt}(I')/\mathrm{cost}(x) \leq \rho(I')$ and thus $\mathrm{cost}(x) \geq \mathrm{opt}(I')/\rho(I')$. By property (ii) it follows that $\mathrm{cost}(x) \geq \mathrm{opt}(I')/\rho(I') \geq (\xi(I') \cdot \rho(I'))/\rho(I') = \xi(I')$. Hence, by distinguishing the two cases $\mathrm{cost}(x) < \xi(I')$ and $\mathrm{cost}(x) \geq \xi(I')$ we can decide the instance (I, p) of A in $(g(p) + f(p)) \cdot |I|^{\mathcal{O}(1)}$ time. Thus A is fixed-parameter tractable, and since A is $\mathcal{C}$-hard it follows that $\mathrm{FPT} = \mathcal{C}$. $\square$

Chapter 3.

Basics on Degree Factors

Most results obtained in this thesis as well as some of the employed core tools are strongly related to degree factors and degree sequences. Surveying these connections, in this chapter we briefly overview the rich literature of degree factors and degree sequences. This chapter is, however, no precondition to understand the rest of this thesis, and in later chapters we give explicit reference when we make use of results mentioned in this chapter.

Let us recall some useful definitions. A *degree factor* in a graph $G = (V, E)$ is a spanning subgraph of G satisfying some predefined degree conditions. Here, $G' = (V', E')$ is a *spanning subgraph* of G if $V = V'$ and $E' \subseteq E$. A very prominent—and for this thesis important—example of degree factors are *f-factors*. Given a function $f\colon V \to \mathbb{N}$, an *f-factor* in a graph G is a spanning subgraph $G' = (V, E')$ of G such that $\deg_{G'}(v) = f(v)$ for all $v \in V$, see Figure 3.1 for an illustration. In Section 3.2 we survey results about f-factors; these play an important role for a kernelization algorithm presented in Chapter 6. The special case of f-factors in complete graphs is known as *realizable degree sequences*. In Chapter 4 we provide results about realizable degree sequences of directed acyclic graphs, and in Chapter 6 we use results about realizable degree sequences of undirected graphs. Section 3.1 lists the related results from the literature.

With the focus of this work being on finding efficient algorithms or ruling out their existence under standard assumptions from complexity theory, we concentrate in this chapter on showing the borderline between tractability and intractability. This means that we mostly consider the decision problem of whether a given graph contains a certain degree factor; f-Factor is the corresponding problem for f-factors, see Section 3.2.1 for the formal definition.

While the f-Factor problem is polynomial-time solvable, we present in Section 3.2 several generalizations, some of which are still polynomial-time

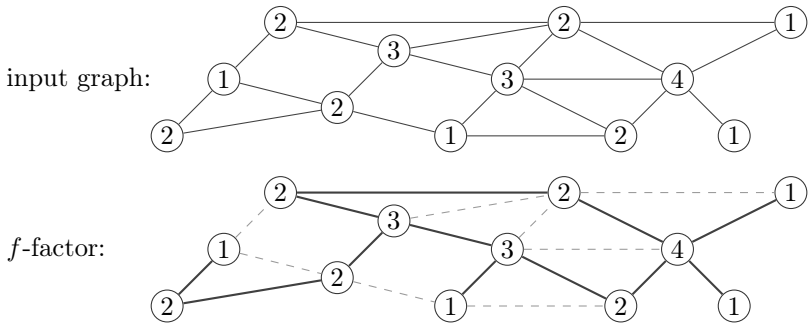


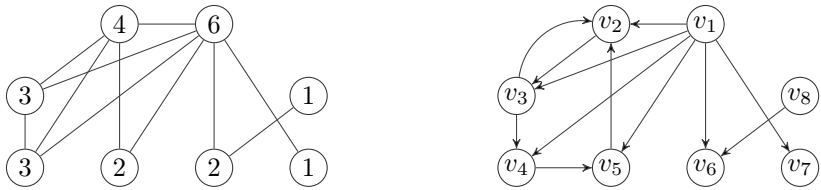
Figure 3.1.: *Top*: the input graph G with the numbers inside the vertices specifying the degree constraint f. *Bottom*: an f-factor in G. The dashed edges are in the input graph but not part of the f-factor.

solvable while others are NP-complete. We start in Section 3.1 with realizable degree sequences which are a special case of f-factors.

3.1. Realizable Degree Sequences

The problem of deciding whether there exists a graph with a prescribed degree sequence is quite old [EG60, Hak62, Hav55]. In this section, we survey some basic results for this problem on undirected graphs (Section 3.1.1) and on directed graphs (Section 3.1.2). Before doing so, we recall some necessary notation and definitions. A degree sequence is a multiset of nonnegative integers or a multiset of nonnegative integer tuples, depending on whether we search for undirected or directed graphs. Although in a multiset no ordering is given, we stick to the term degree *sequence* as it is commonly used in the literature. Naturally, we call the elements of the degree sequence *degrees*. The degree sequence $\mathcal{S}$ of a graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ is the multiset $\{\deg(v_1), \deg(v_2), \ldots, \deg(v_n)\}$. If there exists for a degree sequence $\mathcal{S}$ a graph G with degree sequence $\mathcal{S}$, then $\mathcal{S}$ is called *realizable*[1] and G is called *realization of* $\mathcal{S}$ or *realizing graph*, see Figure 3.2 for an illustrated example. Since in this work n_G denotes the number of vertices of some graph G and the number of degrees in a degree sequence $\mathcal{S}$ is equal to the number of vertices of a realizing graph, we set $n_{\mathcal{S}} := |\mathcal{S}|$ and omit,

[1] In the literature also the term *graphical* is used.



$\mathcal{S} = \{6, 4, 3, 3, 2, 2, 1, 1\}$ $\qquad$ $\mathcal{S} = \{\binom{0}{6}, \binom{3}{1}, \binom{2}{2}, \binom{2}{1}, \binom{2}{1}, \binom{2}{0}, \binom{1}{0}, \binom{0}{1}\}$

Figure 3.2.: *Left:* An undirected graph with degree sequence $\mathcal{S} = \{6, 4, 3, 3, 2, 2, 1, 1\}$. The numbers inside the vertices denote their degrees. *Right:* A digraph with degree sequence $\mathcal{S} = \{\binom{0}{6}, \binom{3}{1}, \binom{2}{2}, \binom{2}{1}, \binom{2}{1}, \binom{2}{0}, \binom{1}{0}, \binom{0}{1}\}$. Here, the vertex v_1 corresponds to the first degree $\binom{0}{6}$, the vertex v_2 corresponds to the second degree $\binom{3}{1}$, and so on.

as in the graph setting, the subscript $\mathcal{S}$ if $\mathcal{S}$ is clear from the context. We assume that the maximum degree (undirected case) as well as the maximum indegree and the maximum outdegree (directed case) are upper-bounded by $n - 1$, as otherwise no realization exists. For convenience we also assume throughout this work that all numbers in the degree sequences are given explicitly, that is, the overall input size of a degree sequence instance is $\mathcal{O}(n \log n)$ (both in the directed and undirected case).

3.1.1. Realizing Undirected Graphs

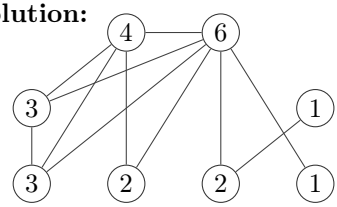
The basic problem considered in this section is the following.

Graph Realization [LP86, Chapter 10]
Input: A degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$.
Question: Is there an undirected graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ such that for all $i \in \{1, 2, \ldots, n\}$ the degree of v_i is $\deg(v_i) = d_i$?

Input:

$\mathcal{S} = \{6, 4, 3, 3, 2, 2, 1, 1\}$

Solution:



Algorithm 3.1: Pseudocode of the Havel-Hakimi algorithm.

Input: A degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$.
Output: An undirected graph G realizing $\mathcal{S}$ if it exists, 'NO' otherwise.

```
V ← {v_1, v_2, ..., v_n}                    // the realizing graph will have n vertices
E ← ∅                                       // edges will be added during the algorithm
for i ← 1 to n do
    d_i^r ← d_i                             // initializing the residual degrees
while ∃i ∈ {1, 2, ..., n} : d_i^r > 0 do
    v_i ← vertex with residual degree d_i^r > 0
    V^i ← the d_i^r vertices with the highest residual degree in V \ {v_i}
    foreach v_j ∈ V^i do                    // add edges between v_i and all vertices in V^i
        E ← E ∪ {v_i, v_j}
        d_j^r ← d_j^r − 1
        if d_j^r < 0 then return 'NO'       // S is not realizable
    d_i^r ← 0
return G ← (V, E).
```

Havel [Hav55] and Hakimi [Hak62] independently discovered a simple greedy algorithm known as *Havel-Hakimi algorithm* for Graph Realization. This algorithm starts with an edgeless n-vertex graph G and repeatedly inserts edges in the following way: It picks an arbitrary vertex v_i whose *residual degree* $d_i - \deg(v_i)$ is at least one, that is, v_i needs at least one further neighbor. Then, it adds edges from v_i to the $d_i - \deg(v_i)$ vertices in G with the highest residual degree. When all residual degrees are zero, then a realization of $\mathcal{S}$ is found. The pseudocode is given in Algorithm 3.1. For an illustration of the algorithm obtaining for the degree sequence $\mathcal{S} = \{6, 4, 3, 3, 2, 2, 1, 1\}$ the realizing graph displayed in Figure 3.2 (left) we refer to Figure 3.3. Carefully managing the residual degrees, the running time is $\mathcal{O}(\Delta n)$ as for each vertex up to Δ edges are inserted. The correctness of this recursive algorithm relies on the following theorem.

Theorem 3.1 (Havel [Hav55], Hakimi [Hak62])**.** *Let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ be a degree sequence, where $d_1 \geq d_2 \geq \ldots \geq d_{n-1}$ and $d_n > 0$. Then, $\mathcal{S}$ is realizable if and only if $\{d_1 - 1, d_2 - 1, \ldots, d_{d_n} - 1, d_{d_n+1}, d_{d_n+2}, \ldots, d_{n-1}\}$ is realizable.*

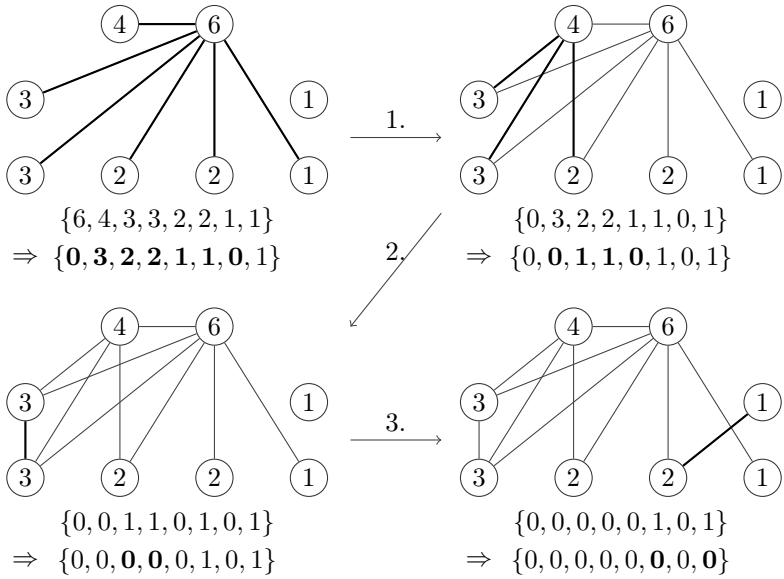


Figure 3.3.: The Havel-Hakimi algorithm realizing the degree sequence $\mathcal{S} = \{6,4,3,3,2,2,1,1\}$. Each displayed graph corresponds to one iteration of the loop in Line 5 of Algorithm 3.1. Below each graph the residual degrees before and after the insertion of the edges are displayed.

Note that there is no prescribed ordering between d_n and the rest of the degree sequence. This allows us in Figure 3.3 to simply pick the first vertex with non-zero residual degree. While we use this algorithm in our implementation in Chapter 6 to construct a graph for a degree sequence, we also use a faster test deciding whether a given degree sequence is realizable. This test is based on the following characterization of realizable degree sequences.

Theorem 3.2 (Erdős and Gallai [EG60])**.** *Let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ be a degree sequence, where $d_1 \geq d_2 \geq \ldots \geq d_n$. Then, $\mathcal{S}$ is realizable if and only if $\sum_{i=1}^{n} d_i$ is even and for all $r \in \{1, 2, \ldots, n-1\}$ it holds that*

$$\sum_{i=1}^{r} d_i \leq r(r-1) + \sum_{i=r+1}^{n} \min\{d_i, r\}. \tag{3.1}$$

While this characterization only gives a test running in $\mathcal{O}(n^2)$ time (checking n inequalities with n variables each), Tripathi and Vijay [TV03] showed that it is sufficient to check Inequality (3.1) only for values of r where $d_r > d_{r+1}$ and for $r = n$. With an additional linear-time preprocessing step counting the frequencies of each degree, this allows us to test in $\mathcal{O}(n + \Delta^2)$ time whether a given degree sequence $\mathcal{S}$ is realizable.

In the experiments in Chapter 6, we also face the following task: Given a degree sequence $\mathcal{S}$ that is not realizable, how many 1's have to be added to $\mathcal{S}$ in order to obtain a realizable degree sequence? We use Theorem 3.2 to answer this question. To this end, let us briefly discuss the necessity of the given conditions given in Theorem 3.2. Let $G = (V, E)$ be a realization of $\mathcal{S}$. Since each edge contributes one to the degree of two vertices, it follows that the sum of the degrees has to be even. Furthermore, consider the subset $V_r \subseteq V$ of r vertices realizing the r highest degrees. Clearly there can be at most $\binom{r}{2}$ edges with both endpoints in V_r. This yields the summand $r(r-1) = 2\binom{r}{2}$ in Inequality (3.1). Each vertex $v_i \in V \setminus V_r$ has at most $\min\{\deg(v_i) = d_i, r\}$ neighbors in V^r. Hence, the sum of the degrees of the vertices in V^r has to satisfy Inequality (3.1). Thus, in order to accomplish the above mentioned task, we denote by $\operatorname{diff}(r)$ the difference between the left-hand side and the right-hand side of Inequality (3.1), that is,

$$\operatorname{diff}(r) := \sum_{i=1}^{r} d_i - \left(r(r-1) + \sum_{i=r+1}^{n} \min\{d_i, r\} \right).$$

If $\max\{\operatorname{diff}(r) \mid 1 \leq r \leq n-1\}$ is even, then the number of 1's that need to be added to $\mathcal{S}$ is $\max\{\operatorname{diff}(r) \mid 1 \leq r \leq n\}$, otherwise this number is $\max\{\operatorname{diff}(r) \mid 1 \leq r \leq n\} + 1$.

Generalizations. Besides deciding whether $\mathcal{S}$ is realizable, also the question whether there exists a realizing graph having some specific property P is studied in the literature. Examples for P that have been studied are ℓ-edge connected and ℓ-vertex connected for some given integer ℓ, Hamiltonian, self-complementary, or planar, to mention just a few. We refer to the surveys of Hakimi and Schmeichel [HS78] and Rao [Rao81] for further details and a list of results. In this thesis, we deal with such a question in the directed case, namely asking for an acyclic realization. Indeed, we show in Chapter 4 that deciding whether a given degree sequence is realizable by a directed *acyclic* graph is NP-complete. The corresponding problem in the undirected setting is,

in contrast to the directed case, polynomial-time solvable due to the following characterization.

Theorem 3.3 (Lovász and Plummer [LP86, Exercise 10.3.5], Edmonds [Edm64])**.** *Let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ be a degree sequence, where $d_1 \geq d_2 \geq \ldots \geq d_n$. Then, $\mathcal{S}$ is realizable by*

(i) a tree if and only if $\sum_{i=1}^{n} d_i = 2n - 2$ and $d_n > 0$;

(ii) a connected graph if and only if $\mathcal{S}$ is realizable, $\sum_{i=1}^{n} d_i \geq 2n-2$, and $d_n > 0$.

Finally, we remark that for the more general f-Factor problem, the corresponding variants asking for an acyclic or a connected f-factor are NP-complete: there are simple reductions from the Hamilton Cycle problem [Gol14a].

3.1.2. Realizing Directed Graphs

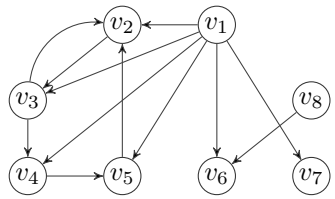
The basic problem considered in this section is the following.

Digraph Realization
Input: A degree sequence $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2} \ldots, \binom{a_n}{b_n} \right\}$.
Question: Is there a directed graph $D = (V, A)$ with $V = \{v_1, v_2, \ldots, v_n\}$ such that for all $i \in \{1, 2, \ldots, n\}$ the indegree of v_i is $\deg^-(v_i) = a_i$ and the outdegree is $\deg^+(v_i) = b_i$?

Input:

$$\mathcal{S} = \left\{ \binom{0}{6}, \binom{3}{1}, \binom{2}{2}, \binom{2}{1}, \binom{2}{1}, \binom{2}{0}, \binom{1}{0}, \binom{0}{1} \right\}$$

Solution:



Many results from the undirected setting carry over to the directed setting; this is also true for a greedy algorithm realizing a given degree sequence as digraph. This algorithm due to Kleitman and Wang [KW73] was rediscovered by Erdős et al. [EMT10] and works similarly as the Havel-Hakimi algorithm. For its

Algorithm 3.2: Pseudocode of the algorithm due to Kleitman and Wang.

Input: A degree sequence $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n} \right\}$.
Output: A digraph D realizing $\mathcal{S}$ if it exists, 'NO' otherwise.

```
if ∑_{i=1}^n a_i ≠ ∑_{i=1}^n b_i then
    return 'NO'                                   // S is not realizable
V ← {v_1, v_2, ..., v_n}                          // the realizing graph will have n vertices
A ← ∅                                             // arcs will be added during the algorithm
for i ← 1 to n do
    a_i^r ← a_i
    b_i^r ← b_i                                   // initializing the residual degrees
while ∃i ∈ {1, 2, ..., n} : b_i^r > 0 do
    v_i ← vertex with residual outdegree b_i^r > 0
    arrange the vertices in V \ {v_i} in lexicographic order
    V^i ← first b_i^r vertices in this order
    foreach v_j ∈ V^i do                          // add arcs between v_i and all vertices in V^i
        A ← A ∪ (v_i, v_j)
        a_j^r ← a_j^r − 1
        if d_j^r < 0 then return 'NO'             // S is not realizable
    b_i^r ← 0
return G ← (V, A).
```

description, we need a further definition. The integer tuples $\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ are in *lexicographic* order if $a_i > a_{i+1}$ or $a_i = a_{i+1}$ and $b_i \geq b_{i+1}$ for all $1 \leq i < n$.

The algorithm starts with an arcless n-vertex graph G and repeatedly inserts arcs in the following way: It picks an arbitrary vertex v_i whose *residual outdegree* $b_i - \deg^+(v_i)$ is at least one, that is, v_i needs at least one further neighbor. Then, it adds arcs from v_i to the $b_i - \deg^+(v_i)$ vertices in G with the highest residual degree according the lexicographic order. When all residual indegrees and outdegrees are zero, then a realization of $\mathcal{S}$ is found. The pseudocode is given in Algorithm 3.2. For an illustration of the algorithm obtaining for the degree sequence $\mathcal{S} = \left\{ \binom{0}{6}, \binom{3}{1}, \binom{2}{2}, \binom{2}{1}, \binom{2}{1}, \binom{2}{0}, \binom{1}{0}, \binom{0}{1} \right\}$ a realizing graph displayed in Figure 3.2 (right) we refer to Figure 3.4. Carefully managing the residual in- and outdegrees, the running time is $\mathcal{O}(\Delta n)$ as for each vertex up to Δ arcs are inserted.

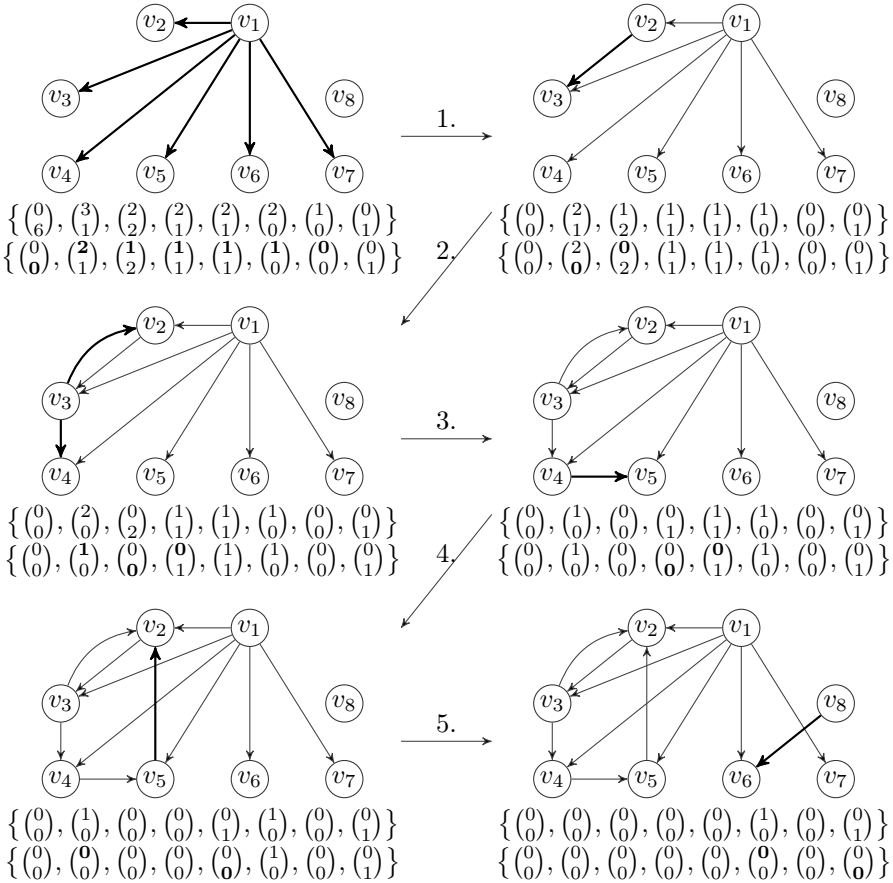


Figure 3.4.: The algorithm due to Kleitman and Wang [KW73] realizing the degree sequence $\mathcal{S} = \left\{\binom{0}{6}, \binom{3}{1}, \binom{2}{2}, \binom{2}{1}, \binom{2}{1}, \binom{2}{0}, \binom{1}{0}, \binom{0}{1}\right\}$. Each displayed graph corresponds to one iteration of the loop in Line 8 of Algorithm 3.2. Below each graph the residual degrees before and after the insertion of the arcs are displayed.

The correctness of the algorithm relies on the next result corresponding to Theorem 3.1.

Theorem 3.4 (Erdős et al. [EMT10], Kleitman and Wang [KW73]). *Let* $\mathcal{S} = \left\{\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}\right\}$ *be a degree sequence where* $b_n > 0$ *and* $\binom{a_1}{b_1}$, $\binom{a_2}{b_2}$, ..., $\binom{a_{n-1}}{b_{n-1}}$ *are in lexicographic order. Then,* $\mathcal{S}$ *is realizable if and only if*

$$\left\{\binom{a_1-1}{b_1}, \binom{a_2-1}{b_2}, \ldots, \binom{a_{b_n}-1}{b_{b_n}}, \binom{a_{b_n+1}}{b_{b_n+1}}, \binom{a_{b_n+2}}{b_{b_n+2}}, \ldots, \binom{a_{n-1}}{b_{n-1}}\right\}$$

is a realizable degree sequence.

In Chapter 4 we use a slightly modified variant of Algorithm 3.2 to realize a given degree sequence by a directed acyclic graph with prescribed topological order, see Lemma 4.10.

For the sake of completeness we mention the characterization of realizable degree sequences in the directed setting that corresponds to Theorem 3.2, although we not do use it.

Theorem 3.5 (Fulkerson [Ful60], Gale [Gal57], Ryser [Rys57]). *Let* $\mathcal{S} = \left\{\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}\right\}$ *be a degree sequence where* $a_1 \geq a_2 \geq \ldots \geq a_n$. *Then,* $\mathcal{S}$ *is realizable if and only if* $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i$ *and for all* $r \in \{1, 2, \ldots, n-1\}$ *it holds that*

$$\sum_{i=1}^{r} a_i \leq \sum_{i=1}^{r} \min\{b_i, r-1\} + \sum_{i=r+1}^{n} \min\{b_i, r\}. \tag{3.2}$$

Compared to the characterization for the undirected case (see Theorem 3.2) we have the following differences. In the undirected case each edge contributes one to the degree of two vertices. In the directed case, however, each added arc contributes one to one indegree and to one outdegree. Thus, the condition that the sum of the degrees is even is replaced by the condition that the sum of the indegrees is equal to the sum of the outdegrees. Similarities can be seen between Inequalities (3.1) and (3.2): In both inequalities an upper bound on the sum of the r highest (in)degrees is checked. The only difference is that the summand $r(r-1)$ in Inequality (3.1) is in Inequality (3.2) replaced by $\sum_{i=1}^{r} \min\{b_i, r-1\}$. Note that $\sum_{i=1}^{r} \min\{b_i, r-1\} \leq r(r-1)$, hence the inequality is tightened in the directed case. The reason is that if each of the

outdegrees $b_1, b_2, \ldots, b_r$ is smaller than $r-1$, then the vertices V_r realizing the degrees $\binom{a_1}{b_1}, \binom{a_2}{b_2} \ldots, \binom{a_r}{b_r}$ cannot form a clique in the realization of $\mathcal{S}$. Thus, this difference covers the case that the indegrees $a_1, a_1, \ldots, a_r$ are "large" and the outdegrees $b_1, b_2 \ldots, b_r$ are "small".

Analogously to the undirected case, it is sufficient to check Inequality (3.2) for all values of r where $a_r > a_{r+1}$ and for $r = n$ [Ber14].

Generalizations. Similarly to the undirected case, it has been studied in the literature whether a given degree sequence can be realized by a digraph with some specific property P. We refer to the survey of Rao [Rao81] for an overview. If P is acylicity, then the corresponding problem is NP-complete as proven in Chapter 4. Hence, there is no simple characterization similar to Theorem 3.3(i), unless P = NP. Considering connectivity as required property yields polynomial-time solvable problems [BH65]. For completeness, we mention the two corresponding results; one result for each of two different notions of connectedness of digraphs. First, we consider *weakly connected* digraphs, that is, digraphs where the underlying undirected graph is connected. Here, the result is very similar to Theorem 3.3 (ii).

Theorem 3.6 (Beineke and Harary [BH65])**.** *A realizable degree sequence* $\mathcal{S} = \left\{\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}\right\}$ *is realizable by a weakly connected digraph if and only if* $\binom{0}{0} \notin \mathcal{S}$ *and* $\sum_{i=1}^{n} a_i \geq n-1$.

Second, we consider strongly connected digraphs. A digraph $D = (V, A)$ is *strongly connected* if for every pair of vertices $u, v \in V$ there is a directed path from u to v and a directed path from v to u. Surprisingly, the corresponding characterization has more in common with Theorem 3.5 than with Theorem 3.6.

Theorem 3.7 (Beineke and Harary [BH65])**.** *Let* $\mathcal{S} = \left\{\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}\right\}$ *be a realizable degree sequence, where* $a_1 \geq a_2 \geq \ldots \geq a_n$. *Then* $\mathcal{S}$ *is realizable by a strongly connected digraph if and only if* $a_i > 0$ *and* $b_i > 0$ *for all* $1 \leq i \leq n$ *and for each* $r \in \{1, 2, \ldots, n-1\}$

$$\sum_{i=1}^{r} a_i < \sum_{i=1}^{r} b_i + \sum_{i=r+1}^{n} \min\{r, b_i\}.$$

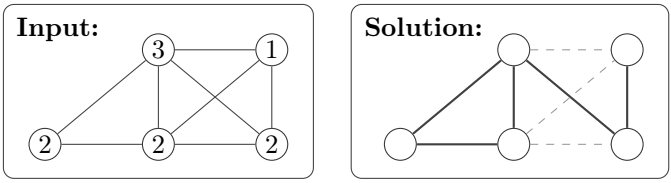
3.2. f-Factors and More General Degree Factors

In this section, we give a brief overview on f-factors—the core tool used in the kernelization algorithm we are presenting in Chapter 6. As in Sections 3.1.1 and 3.1.2, we start in Section 3.2.1 with the algorithmic results and then continue with structural results. Since the corresponding algorithms are rather complicated and we only need them as black boxes, we just list the corresponding results and refrain from further explanation like it is done in Section 3.1. In Section 3.2.2, we discuss generalizations of the f-Factor problem and show connections to Degree Anonymity.

3.2.1. f-Factors

The f-Factor problem is formally defined as follows.

f-Factor [LP86, Chapter 10]
Input: An undirected graph $G = (V, E)$ and a function $f\colon V \to \mathbb{N}$.
Question: Does G contain an f-factor?



The first polynomial-time algorithm for the f-Factor problem is due to Tutte [Tut54] and Edmonds [Edm65]. Tutte [Tut54] gave a polynomial-time reduction from the f-Factor problem to the 1-Factor problem (where $f \equiv 1$) and Edmonds [Edm65] showed with his *blossom algorithm* that one can find a maximum matching in $\mathcal{O}(n^4)$ time. Here, a matching in a graph G is a nonoverlapping subset of its edge set and a matching is called *perfect* if it covers all vertices, that is, a perfect matching is a 1-factor. So far, the fastest algorithm to find a maximum (weighted) matching is due to Micali and Vazirani [MV80] and runs in $\mathcal{O}(m\sqrt{n})$ time. Using this algorithm, Gabow [Gab83] showed that the f-Factor problem can be solved in $\mathcal{O}(m\sqrt{f(V)})$ time, where $f(V) := \sum_{v \in V} f(v)$ and V is the set of vertices in the input graph. Recently, Gabow and Sankowski [GS13] presented an algebraic approach solving the f-Factor problem in time $\mathcal{O}(f(V)^\omega)$ time, where $\mathcal{O}(x^\omega)$ is the time needed to multiply two $x \times x$ matrices; currently the

best bound is $\omega \approx 2.3727$ [Vas12]. This algorithm is faster if $f(V)$ is small and the input graph is dense.

There exists a rich structure theory behind f-factors, see Akiyama and Kano [AK85] and Plummer [Plu07] for two listings of numerous results in this direction. Here, we refrain from giving an overview but mention a result of Katerinis and Tsikopoulos [KT00] which we will use in Chapter 6.

Lemma 3.8 (Katerinis and Tsikopoulos [KT00])**.** *Let $G = (V, E)$ be an undirected graph with minimum vertex degree δ and let $a \leq b$ be two positive integers. Suppose further that*

$$\delta \geq \frac{b}{a+b}|V| \text{ and } |V| > \frac{a+b}{a}(a+b-3).$$

Then, for any function $f\colon V \to \{a, a+1, ..., b\}$ where $\sum_{v \in V} f(v)$ is even, G has an f-factor.

3.2.2. Generalizations of f-Factors

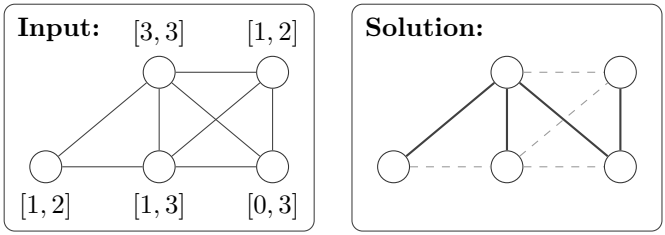
Several generalizations of f-factors have been looked at. They can be categorized into two types. The problems of the first type change the degree constraints. The problems of the second type change the condition that the sought graph needs to be a spanning subgraph. Instead, they ask for solutions being "close" to the given graph, where close is usually measured by the edit distance with respect to some graph modification operation, for example, vertex deletions or edge insertions. This leads to graph modification problems as, for example, Degree Anonymity. The f-Factor problem can be seen as a graph modification problem where only edge deletions are allowed. Furthermore, also the degree sequence realization problems considered in Section 3.1 are graph modification problems where the input graph is edgeless and only edge insertions are allowed.

Generalized degree constraints. Generalizing the f-Factor problem such that for each vertex a range of allowed degrees is given instead of just one degree leads to the (g, f)-Factor problem, where g gives a lower bound and f an upper bound on the degrees.

(g, f)-Factor [LP86, Chapter 10]

Input: An undirected graph $G = (V, E)$ and a two functions $g, f \colon V \to \mathbb{N}$ with $g(v) \leq f(v)$ for all $v \in V$.

Question: Does G contain an (g, f)-*factor*, that is, a spanning subgraph $G' = (V, E')$ of G such that $g(v) \leq \deg_{G'}(v) \leq f(v)$ for all $v \in V$?



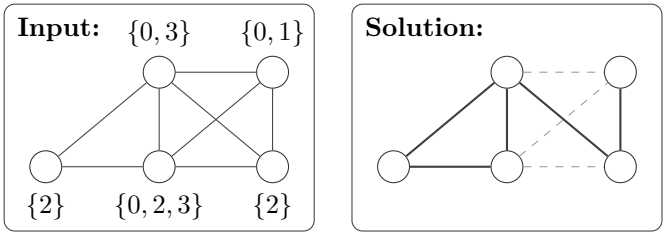
Notably, the above mentioned algorithm of Gabow [Gab83] returns a (g, f)-factor with the maximum number of edges. Furthermore, Gabow generalized this result to the weighted variant, where as additional input real-valued edge-weights are provided and the task is to find a maximum-weight (g, f)-factor. For this weighted (g, f)-Factor problem he provided an algorithm running in $\mathcal{O}(f(V) \cdot (n^2 + m \log n))$ time.

A natural generalization of (g, f)-factors is that, instead of giving upper and lower bounds for the degrees, a set of allowed degrees is provided for every vertex. This variant, introduced by Lovász [Lov72], is called General Factor and defined as follows. To this end, we denote the power set of a set $S$by 2^S.

General Factor [Lov72]

Input: An undirected graph $G = (V, E)$ and a function $\tau \colon V \to 2^{\{0,1,\ldots,n-1\}}$.

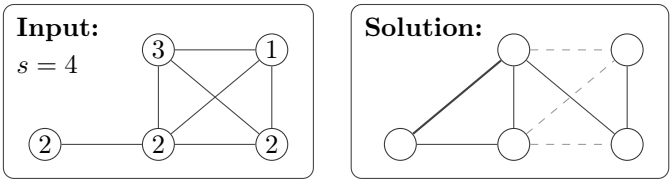
Question: Does G contain a *general factor*, that is, a spanning subgraph $G' = (V, E')$ of G such that $\deg_{G'}(v) \in \tau(v)$ for all $v \in V$?



While the general problem is NP-complete [Lov72], Cornuéjols [Cor88] proved that the problem variant where τ contains no gap of size at least two is polynomial-time solvable. Here, τ contains a gap of size two if for a vertex $v \in V$ and an integer $\ell \in \mathbb{N}$ it holds that $\ell, \ell+3 \in \tau(v)$, but $\ell+1, \ell+2 \notin \tau(v)$.

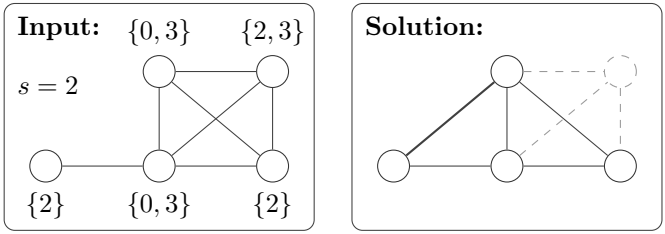
Degree-constrained graph modification problems. Another type of generalization of the f-factor problem is when instead of searching for a subgraph one searches for a graph that is "close" to the input graph, turning the problem into a graph modification problem where vertex deletions as well as edge deletions and insertions are allowed. In this line, Mathieson and Szeider [MS12] considered the probably simplest generalization of f-Factor by allowing edge editings, that is, edge deletions and insertions.

f-Factor Editing [MS12]
Input: An undirected graph $G = (V, E)$, an integer $s \in \mathbb{N}$, and a function $f\colon V \to \mathbb{N}$.
Question: Can G be transformed by at s edge editing operations into a graph $G' = (V, E')$ such that $\deg_{G'}(v) = f(v)$ for all $v \in V$?



Mathieson and Szeider showed that f-Factor Editing is polynomial-time solvable by a reduction to the weighted 1-Factor problem. We remark that Mathieson and Szeider did not introduce a name for f-Factor Editing but simply treated it as a polynomial-time solvable special case of their much more general Degree Constraint Editing problem, which is defined as follows.

Degree Constraint Editing [MS12]
Input: An undirected graph $G = (V, E)$, two integers $s, r \in \mathbb{N}$, and a function $\tau\colon V \to 2^{\{0,1,\ldots,r\}}$.
Question: Is it possible to obtain a graph $G' = (V', E')$ from G using at most s editing operations such that $\deg_{G'}(v) \in \tau(v)$ for all $v \in V'$?



Degree Constraint Editing can be interpreted as the graph modification variant of the General Factor problem. Mathieson and Szeider [MS12] considered the modification operations edge insertion, edge deletion, and vertex deletion and showed that Degree Constraint Editing is for each combination of the three operations W[1]-hard with respect to the parameter s and fixed-parameter tractable with respect to the combined parameter (s, r). The fixed-parameter tractability is achieved by a general meta-theorem due to Frick and Grohe [FG01], Golovach [Gol14b] presented direct combinatorial algorithms. Interestingly, Mathieson and Szeider also studied kernelization, where they left as most challenging open problem to extend their kernelization results to cases that include vertex deletion and edge insertion, emphasizing that the presence of edge insertions is making their approach inapplicable. In Chapter 6, we show a kernelization approach that can be applied to a broad subclass of degree-constrained completion problems. Furthermore, we recently used the presented ideas to provide a polynomial problem kernel for Degree Constraint Editing when only edge insertions are allowed [FNN14].

Degree Constraint Editing naturally generalizes Bounded-Degree Deletion and Regular-Degree-d Vertex Deletion. The Bounded-Degree Deletion problem asks, given an undirected graph G and two integers $d, s \in \mathbb{N}$, whether at most s vertices can be deleted such that the maximum degree in the resulting graph is at most d. Fellows et al. [Fel+11] showed a polynomial size problem kernel for Bounded Degree Deletion with respect to the combined parameter (s, d). They further proved W[2]-hardness with respect to the single parameter s. Betzler et al. [Bet+12] proved that Bounded-Degree Deletion is W[1]-hard with respect to the parameter treewidth, but becomes fixed-parameter tractable with respect to the combined parameter treewidth and solution size s. Closely related to Bounded-Degree Deletion is the Regular-Degree-d Vertex Deletion problem, where given an undirected graph G and an integer $s \in \mathbb{N}$, the task is to decide whether G can be made d-regular by at most s vertex deletions. Moser and Thilikos [MT09] showed that

Regular-Degree-d Vertex Deletion can be solved in $\mathcal{O}(n(s+d)+(d+2)^s)$ time and presented a problem kernel of size $\mathcal{O}(sd(d+s)^2)$. Degree Constraint Editing also generalizes many well-studied NP-complete problems like Vertex Cover (s is the size of the sought vertex cover, $\tau(v) := \{0\}$ for all $v \in V$, and only vertex deletions are allowed) or Dominating Set (s is the size of the sought dominating set, $\tau(v) := \{0, 1, \ldots, \deg_G(v) - 1\}$ for all $v \in V$, and only vertex deletions are allowed).

Note that Degree Constraint Editing is, however, neither a special case nor a generalization of Degree Anonymity: The degree constraints in the Degree Constraint Editing problem, in the f-Factor problem and in all the presented generalizations of f-Factor are *local*: independent from the rest of the graph, one only needs to know the degree of a vertex to decide whether it fulfills the degree constraints. Degree Anonymity differs from the degree factors presented in this section as the degree property is *global*. To this end, recall that the task in Degree Anonymity is, given an undirected graph G and two integers $k, s \in \mathbb{N}$, to decide whether at most s graph modification operations can transform G into a k-anonymous graph. Here, a graph is called k-anonymous if for each vertex there are at least $k-1$ other vertices of same degree (see page 7 for further details). Thus, whether one particular vertex v fulfills the degree constraints depends on the degrees of other vertices that may be far away from v. These *global* degree constraints cannot be expressed in the Degree Constraint Editing problem. Furthermore, the global nature of the degree constraints is one explanation for the very strong intractability results for the edge and vertex deletion variants of Degree Anonymity, see Chapter 5.

Chapter 4.

DAG Realization

In this chapter, we investigate the computational complexity of the DAG Realization problem. It asks, given a degree sequence $\mathcal{S}$, whether $\mathcal{S}$ can be realized by a directed acyclic graph. Answering an open question of Berger and Müller-Hannemann [BM11], we show that DAG Realization is NP-complete. On the positive side, we classify DAG Realization as fixed-parameter tractable with respect to the parameter maximum degree.

4.1. Introduction

It has been known for a long time that it is decidable in polynomial time whether a given degree sequence is realizable by an undirected graph. There are characterizations for realizable degree sequences due to Erdős and Gallai [EG60] and algorithms by Havel [Hav55] and Hakimi [Hak62]. The problem variant where one asks whether there is a directed graph realizing the given degree sequence has also been intensively studied; see Chen [Che66], Fulkerson [Ful60], Gale [Gal57], and Ryser [Rys57] for characterizations of realizable degree sequences and Kleitman and Wang [KW73] for polynomial-time algorithms. Further research then concentrated on realizing degree sequences as graphs with certain properties, for example requiring that the realizing graph shall be connected. While for the connectedness Edmonds [Edm64] and Beineke and Harary [BH65] gave characterizations for the directed as well as undirected case, we consider in this chapter another important graph property, namely acyclicity. Here, in the undirected case there exists a simple characterization, see Lovász and Plummer [LP86, Exercise 10.3.5.]. We show that the directed case, however, is much more complicated. The corresponding graph realization problem, introduced by Berger and Müller-Hannemann [BM11], is defined as follows:

DAG Realization

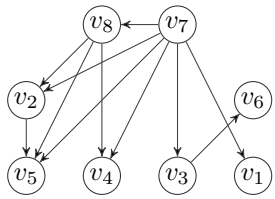
Input: A degree sequence $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n} \right\}$.

Question: Is there a directed acyclic graph with vertex set $\{v_1, v_2, \ldots, v_n\}$ such that for every $v_i \in V$, $1 \leq i \leq n$, the indegree is a_i and the outdegree is b_i?

Input:

$$\mathcal{S} = \left\{ \binom{1}{0}, \binom{2}{1}, \binom{1}{1}, \binom{2}{0}, \binom{3}{0}, \binom{1}{0}, \binom{0}{6}, \binom{1}{3} \right\}$$

Solution:



This problem arises in the context of randomly generating DAGs satisfying some prespecified degree constraints [Ber11]. Berger and Müller-Hannemann [Ber11, BM11, BM12] investigated restricted variants of DAG Realization that are polynomial-time solvable and performed an extensive experimental study on the general problem. We refer to Section 3.1 for a detailed overview on known results about realizable degree sequences for the undirected as well as the directed case.

Our contributions. Berger and Müller-Hannemann [BM11] left the computational complexity of the general problem as their main open question. We answer this question by showing that DAG Realization is NP-complete. Moreover, on the positive side we classify DAG Realization as fixed-parameter tractable with respect to the parameter maximum degree $\Delta := \max\{a_1, b_1, a_2, b_2, \ldots, a_n, b_n\}$. Denoting by n the number of vertices and by m the number of arcs in a realizing DAG and assuming $\binom{0}{0} \notin \mathcal{S}$, Berger and Müller-Hannemann [BM12] showed that if $m < n$, then DAG Realization is polynomial-time solvable. We extend this result to less sparse and to dense settings by considering the four parameters $m - n + 1$, m/n, $\binom{n}{2} - m$, and $\binom{n}{2}/m$. For the first and the third parameter $m - n + 1$ and $\binom{n}{2} - m$ we prove fixed-parameter tractability, whereas for the second and fourth parameter m/n and $\binom{n}{2}/m$ we show NP-completeness for any constant parameter value greater than one, see Table 4.1 for an overview.

Table 4.1.: Overview on the computational complexity classification of DAG Realization.

Parameter	Result
-	NP-complete (Theorem 4.6 on page 56)
Δ	FPT (Theorem 4.27 on page 83)
$m - n + 1$	FPT (Theorem 4.31 on page 89)
$\binom{n}{2} - m$	FPT (Theorem 4.28 on page 84)
m/n $\binom{n}{2}/m$	NP-complete for each constant > 1 (Theorem 4.7 on page 57)

Preliminaries. We denote with $\uplus$ the multiset sum (e. g. $\{1,1\} \uplus \{1,2\} = \{1,1,1,2\}$). For a degree sequence $\mathcal{S} = \left\{\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}\right\}$ we set $m := \sum_{i=1}^{n} a_i$ and we assume throughout this chapter that $\binom{0}{0} \notin \mathcal{S}$, that the maximum occurring value is at most $n-1$, and that $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i$. If a *degree sequence* $\mathcal{S}$ is a yes-instance of DAG Realization, then $\mathcal{S}$ is called *realizable* and the corresponding directed acyclic graph (DAG for short) D is called a *realizing DAG* for $\mathcal{S}$. For a degree sequence $\mathcal{S}$ and a tuple $s \in \mathcal{S}$ with $s = \binom{a}{b}$, we set $\deg^-(s) := a$ and $\deg^+(s) := b$. A vertex $v \in V$ or a tuple $\binom{a}{b} \in \mathcal{S}$ is called a *source* if $\deg^-(v) = 0 = a$ and it is called a *sink* if $\deg^+(v) = 0 = b$. Each DAG D admits a *topological ordering*, that is, an ordering of all its vertices $v_1, v_2, \ldots, v_n$ such that for all arcs $(v_i, v_j) \in A$ it holds that $i < j$. Next, we define a central notion for this chapter.

Definition 4.1. An ordered degree sequence $\sigma = \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ is called a *realizable degree ordering* if there is a realizing DAG D for σ admitting a topological ordering $v_1, v_2, \ldots, v_n$ such that $\deg^-(v_i) = a_i$ and $\deg^+(v_i) = b_i$ for all $1 \leq i \leq n$.

In the following, a realizable degree ordering *of a degree sequence* of $\mathcal{S}$ refers to an ordering $\mathcal{S}$ such that it fulfills Definition 4.1. Berger and Müller-Hannemann [BM12] proved that one can check in polynomial time whether a given ordering of a degree sequence is a realizable degree ordering.

Organization. This chapter is structured as follows: Section 4.2 contains the proof of the NP-completeness of DAG Realization. Furthermore, we show

that DAG Realization remains NP-complete in case of any constant ratios $m/n > 1$ and $\binom{n}{2}/m > 1$. In Section 4.3, we show that DAG Realization is fixed-parameter tractable with respect to the parameter maximum degree Δ. Finally, in Section 4.4, we prove that DAG Realization is fixed-parameter tractable with respect to each of the parameters $m - n + 1$ and $\binom{n}{2} - m$.

4.2. NP-Completeness

In this section, we first describe the construction of our reduction proving NP-hardness of DAG Realization and explain the high-level idea of how it works. Then, we prove its correctness. Finally, we show that DAG Realization remains NP-hard in sparse as well as dense setting. More precisely we prove NP-hardness of DAG Realization for any constant ratio of m/n and of $\binom{n}{2}/m$. Observe that containment in NP is easy to see: Guessing an n-vertex DAG and checking whether or not it is a realization for the degree sequence $\mathcal{S}$ is clearly doable in polynomial time. Hence we focus in the following on showing NP-hardness.

We prove NP-hardness for DAG Realization by giving a polynomial-time many-to-one reduction from the strongly NP-hard 3-Partition problem [GJ79, SP15]:

3-Partition
Input: A multiset $\mathcal{A} = \{a_1, a_2, \ldots, a_{3p}\}$ of $3p$ positive integers and an integer B with $\sum_{i=1}^{3p} a_i = pB$ and $\forall i\colon B/4 < a_i < B/2$.
Question: Is there a partition of the $3p$ integers from $\mathcal{A}$ into p disjoint triples such that in every triple the three elements add up to B?

Input:
$$B = 40$$
$$\mathcal{A} = \left\{ \begin{array}{c} 11,\ 11,\ 11,\ 12,\ 12,\ 12,\ 12,\ 12, \\ 13,\ 14,\ 14,\ 15,\ 16,\ 17,\ 18 \end{array} \right\}$$

Solution:
$(11, 11, 18), (12, 13, 15),$
$(12, 12, 16), (11, 12, 17),$
$(12, 14, 14)$

Construction. Throughout this section let $(\mathcal{A}, B)$ be an instance of 3-Partition and let $\mathcal{S}_{\mathcal{A},B}$ be the degree sequence constructed as follows:

$$\mathcal{S}_{\mathcal{A},B} := X_0 \uplus X_1 \uplus \ldots \uplus X_p \uplus \left\{ \binom{a_1}{a_1}, \binom{a_2}{a_2}, \ldots, \binom{a_{3p}}{a_{3p}} \right\}.$$

We formally define the multisets X_i, $0 \leq i \leq p$, after giving the idea of the construction. For the description of the idea we need some notation: We set $X := \biguplus_{i=0}^{p} X_i$, call a tuple in X an *x-tuple*, and call all tuples in $\mathcal{S}_{\mathcal{A},B} \setminus X$ *a-tuples*. In a realizing DAG the vertices realizing x-tuples are called *x-vertices* and the vertices realizing a-tuples are called *a-vertices*.

The intuition of the construction is that a DAG realizing $\mathcal{S}_{\mathcal{A},B}$ (if it exists) looks as follows: The tuples of a multiset X_i, $0 \leq i \leq p$, form a "module" in a realizable degree ordering. These modules are a skeletal structure in any realizable degree ordering and there are p "gaps" between them. The construction ensures that each gap is filled with a-vertices adjacent to the vertices in the modules bordering the gap and, moreover, the indegrees and outdegrees of all a-vertices in a gap sum up to B. Hence, these p gaps require to partition the a-vertices into p sets where the in- and outdegrees of the a-vertices in each set sum up to B. Thus, the p sets correspond to a solution for the 3-Partition instance that we reduce from. In the reverse direction, for each triple in a solution of a 3-Partition instance the corresponding a-vertices will be used to fill up one gap. See Figure 4.1 for an example of the construction.

To achieve the mentioned skeletal structure of X, we require the corresponding x-vertices to form a *transitive tournament*. This is a DAG with n vertices and $\binom{n}{2}$ arcs; hence it is the only realization of the degree sequence $\{\binom{0}{n-1}, \binom{1}{n-2}, \ldots, \binom{n-1}{0}\}$. Observe that there is only one DAG realizing such a sequence and, furthermore, a transitive tournament admits only one topological ordering.

Now, we complete the reduction. For the ease of argumentation we fix one DAG $D_{\mathcal{A},B}$ serving as a blueprint for any DAG realizing $\mathcal{S}_{\mathcal{A},B}$ and we explain the construction with $D_{\mathcal{A},B}$. We set $X = \{x_0, x_1, \ldots, x_{2(p+1)B-1}\}$. As indicated in Figure 4.1, each multiset X_i, $0 \leq i \leq p$, contains $2B$ tuples, more specifically, $X_i = \{x_{2iB}, x_{2iB+1}, \ldots, x_{2iB+2B-1}\}$. The corresponding set of x-vertices in $D_{\mathcal{A},B}$ is $V_i = \{v_{2iB}, v_{2iB+1}, \ldots, v_{2iB+2B-1}\}$ and it forms the ith module. Recall that the x-vertices are supposed to form a transitive tournament in $D_{\mathcal{A},B}$ and the topological ordering shall be $v_0, v_1, \ldots, v_{2(p+1)B-1}$. To achieve this, v_j has incoming arcs from all j preceding vertices and outgoing arcs to all succeeding vertices (these are $|X| - 1 - j = 2(p+1)B - 1 - j$ vertices). As mentioned before, there should be some gaps in the skeletal structure provided by the modules formed of the tuples of X. To this end, the first half of the x-vertices in V_i, $0 < i < p$, will have one incoming arc from an a-vertex and the second half of the x-vertices will have an outgoing to an a-vertex. Observe that some x-vertices corresponding to tuples in X_0 and X_p

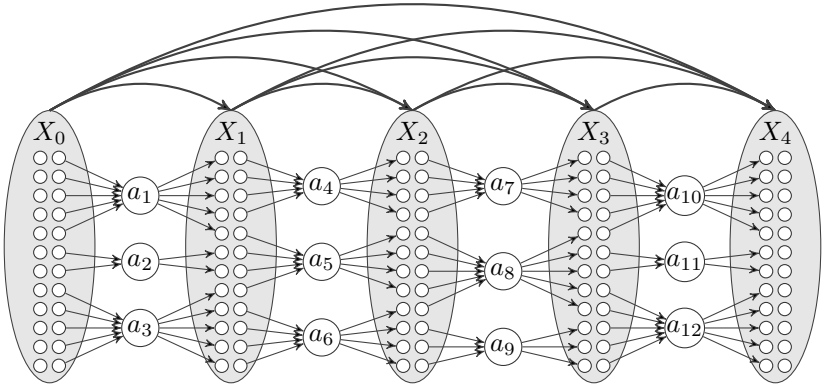


Figure 4.1.: A schematic representation of a DAG that realizes a degree sequence $\mathcal{S}_{\mathcal{A},B}$ that is constructed from a 3-Partition instance with $B = 12$ and $p = 4$. There are five modules marked by the gray ellipses and four gaps between them. In each gap there are three a-vertices, altogether having in- and outdegree B. The sets X_i, $0 \leq i \leq p$, are partitioned into two parts of size B each. The vertices in the left part (except for X_0) have B incoming arcs from the a-vertices that fill the gap between X_{i-1} and X_i. Correspondingly, the vertices in the right part (except in X_p) have B outgoing arcs to the a-vertices that fill the gap between X_i and X_{i+1}. The in- and outdegrees of the a-vertices in each triple sum up to B. The vertices in the gray ellipses form a big transitive tournament where the first vertex with no inneighbors is the top-leftmost vertex and the vertex without outneighbors is the bottom-rightmost vertex. Here, the bold arcs on top indicate that each vertex in an ellipse has outgoing arcs to all vertices in the proceeding gray ellipses.

have neither an incoming nor an outgoing arc to an a-vertex. We introduce the following notation for the ease of argumentation: For each $i \in \{0, 1, \ldots, p-1\}$ we denote by V_i^r the "right part of the ith module V_i" in $D_{\mathcal{A},B}$, formally $V_i^r := \{v_{2iB+B}, v_{2iB+B+1}, \ldots, v_{2iB+2B-1}\}$, and we denote by V_{i+1}^ℓ the "left part of the $(i+1)^{\text{th}}$ module", formally $V_{i+1}^\ell := \{v_{2(i+1)B}, v_{2(i+1)B+1}, \ldots, v_{2(i+1)B+B}\}$. To simplify the index handling we set $V_0^\ell := V_p^r := \emptyset$ and $V^\ell := \bigcup_{i=0}^p V_i^\ell$ and $V^r := \bigcup_{i=0}^p V_i^r$. Furthermore, X^r, X^ℓ, X_i^r, and X_i^ℓ, $0 \leq i \leq p$, contain the corresponding x-tuples. Now we can precisely describe the arcs between a-vertices and x-vertices: For each $0 \leq i \leq p$, each $v \in V_i^r$ has an outgoing arc

to an a-vertex and each vertex $v \in V_{i+1}^{\ell}$ has an incoming arc from an a-vertex. Summarizing, the corresponding x-tuple for v_j is

$$x_j := \begin{pmatrix} j + \mathrm{in}(j) \\ 2(p+1)B - 1 - j + \mathrm{out}(j) \end{pmatrix},$$

where $\mathrm{in}(j)$ and $\mathrm{out}(j)$ are defined as follows:

$$\mathrm{in}(j) := \begin{cases} 1, & \text{if } x_j \in X^{\ell} \\ 0, & \text{else,} \end{cases} \qquad \mathrm{out}(j) := \begin{cases} 1, & \text{if } x_j \in X^{r} \\ 0, & \text{else.} \end{cases}$$

Observe that the strong NP-hardness of 3-Partition is essential to prove the polynomial running time of the reduction: The size of the constructed DAG Realization instance is upper-bounded by a polynomial in the values of the integers in $\mathcal{A}$. Since 3-Partition is strongly NP-hard, it remains NP-hard even when the values of the integers in $\mathcal{A}$ are bounded by a polynomial in the input size. Hence, the size of the DAG Realization instance is polynomially bounded in the size of the 3-Partition instance. Thus, the construction can clearly be performed in polynomial time.

Correctness. Next, we prove the correctness of the construction given above. The main effort here is to prove that $D_{\mathcal{A},B}$ is indeed a blueprint of at least one realizing DAG. Before doing this, we show that if $(\mathcal{A}, B)$ is a yes-instance, then $D_{\mathcal{A},B}$ realizes the constructed degree sequence $\mathcal{S}_{\mathcal{A},B}$ as intended.

Lemma 4.1. *If $(\mathcal{A}, B)$ is a yes-instance of* 3-Partition*, then $\mathcal{S}_{\mathcal{A},B}$ is a yes-instance of* DAG Realization*.*

Proof. We prove that if $(\mathcal{A}, B)$ is a yes-instance, then there exists a realizing DAG for $\mathcal{S}_{\mathcal{A},B}$ as described above and depicted in Figure 4.1. Let π be a permutation of the sequence $\mathcal{A}$ such that $a_{\pi(3i+1)} + a_{\pi(3i+2)} + a_{\pi(3i+3)} = B$ for all $0 \leq i < p$. Since $(\mathcal{A}, B)$ is a yes-instance of 3-Partition such a permutation exists. We now give a complete description of the realizing DAG $D_{\mathcal{A},B}$. As mentioned above, we let the x-vertices in $D_{\mathcal{A},B}$ form a transitive tournament with the topological ordering $v_0, v_1, \ldots, v_{2(p+1)B-1}$. Furthermore, the a-vertices are denoted by $u_1, u_2, \ldots, u_{3p}$ where u_i realizes the a-tuple $\binom{a_i}{a_i}$. Then, the a-vertex $u_{\pi(3i+1)}$ ($u_{\pi(3i+2)}, u_{\pi(3i+3)}$ resp.) has an incoming arc from each of to the first $a_{\pi(3i+1)}$ (next $a_{\pi(3i+2)}$, last $a_{\pi(3i+3)}$) vertices in V_i^r and an outgoing arc to each of the first $a_{\pi(3i+1)}$ (next $a_{\pi(3i+2)}$, last $a_{\pi(3i+3)}$) vertices in V_{i+1}^{ℓ}.

In this way, each a-vertex u_i has an in- and outdegree of a_i. Furthermore, as $a_{\pi(3i+1)} + a_{\pi(3i+2)} + a_{\pi(3i+3)} = B$ for all $0 \leq i < p$, each vertex in V^r has an outgoing arc to an a-vertex and each vertex in V^ℓ has an incoming arc from an a-vertex. Hence each x-vertex v_j indeed realizes the x-tuple x_j. □

To show the reverse direction, we first need some lemmas.

Lemma 4.2. *In any DAG D realizing $\mathcal{S}_{\mathcal{A},B}$, the a-vertices form an independent set and the x-vertices form a transitive tournament.*

Proof. The number $\deg^-(X)$ of ingoing arcs to all x-vertices is

$$\begin{aligned}\deg^-(X) &= \sum_{j=0}^{2(p+1)B-1} \deg^-(x_j) = \sum_{j=0}^{2(p+1)B-1} j + \mathrm{in}(j) \\ &= pB + \sum_{j=0}^{2(p+1)B-1} j = pB + (p+1)B(2(p+1)B-1).\end{aligned}$$

Note that $\deg^-(X)$ is equal to the number $\deg^+(X)$ of outgoing arcs from all x-vertices. The number of a-vertices is $3p$ and the number of x-vertices is $2(p+1)B$. Hence, the number ξ of arcs between the x-vertices is at most:

$$\xi = \binom{2(p+1)B}{2} = (p+1)B(2(p+1)B-1).$$

As a consequence, there are at least $\deg^-(X) - \xi = pB$ arcs going from an a-vertex to an x-vertex. Since $pB = \sum_{i=1}^{3p} a_i$ is the number of outgoing arcs from the a-vertices, *all* outgoing arcs from a-vertices go to x-vertices. Thus, in any realizing DAG D the a-vertices form an independent set and the number of arcs between the x-vertices is exactly ξ. Hence, the x-vertices form a transitive tournament in D. □

We next show that for a realizable degree sequence $\mathcal{S}_{\mathcal{A},B}$ there exists a realizable degree ordering in which the x-vertices are ordered $v_0, v_1, \ldots, v_{2(p+1)B-1}$. To this end, we need some further notations: We use the *opposed order* $\leq_{\mathrm{opp}}$ for the tuples of a degree sequence $\mathcal{S}$ as introduced by Berger and Müller-Hannemann [BM11]: For two tuples $\binom{a}{b}, \binom{a'}{b'} \in \mathcal{S}$ it holds that

$$\binom{a}{b} \leq_{\mathrm{opp}} \binom{a'}{b'} \iff a \leq a' \wedge b \geq b'.$$

Note that there might be tuples in the degree sequence $\mathcal{S}$ that are not comparable with respect to the opposed order (for example $\binom{1}{1}$ and $\binom{2}{2}$). However, the next lemma due to Berger and Müller-Hannemann [BM11] implies that we can always assume that a realizable degree ordering does not collide with the opposed order. (Clearly, there can exist realizing degree orderings which are not in opposed order.)

Lemma 4.3 ([BM11, Corollary 3])**.** *Let $\mathcal{S}$ be a realizable degree sequence. Then, there exists a realizable degree ordering $\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ of $\mathcal{S}$ such that for all $1 \leq i, j \leq n$ with $\binom{a_i}{b_i} \leq_{\text{opp}} \binom{a_j}{b_j}$ and $\binom{a_i}{b_i} \neq \binom{a_j}{b_j}$, it holds that $i < j$.*

As a consequence of Lemma 4.3, if there are two tuples t_1, t_2 in a realizable degree sequence $\mathcal{S}_{\mathcal{A},B}$ such that $t_1 \leq_{\text{opp}} t_2$ and $t_1 \neq t_2$, then we can always assume that there is a realizable degree ordering where the tuple t_1 is ahead of t_2. Furthermore, observe that if $t_1 = t_2$ and there is a realizable degree ordering where t_1 is ahead of t_2, then there is also a realizable degree ordering where t_2 is ahead of t_1 (just exchange these two identical tuples). We use this fact to prove the next lemma.

Lemma 4.4. *If $\mathcal{S}_{\mathcal{A},B}$ is realizable, then there exists a realizable degree ordering σ such that in σ for all $0 \leq i < j < 2(p+1)B$ the tuple x_i is ahead of x_j.*

Proof. To prove that such a realizable degree ordering exists, by Lemma 4.3 and the above discussion, it is sufficient to show for all $0 \leq i < j < 2(p+1)B$ that $x_i \leq_{\text{opp}} x_j$. This can be verified easily as $\text{in}(k), \text{out}(k) \in \{0, 1\}$ for all $0 \leq k < 2(p+1)B$:

$$\begin{aligned}
\deg^-(x_i) - \deg^-(x_j) &= i + \text{in}(i) - j - \text{in}(j) \leq i + 1 - j \leq 0 \\
\deg^+(x_i) - \deg^+(x_j) &= 2(p+1)B - i - 1 + \text{out}(i) \\
&\quad - (2(p+1)B - j - 1 + \text{out}(j)) \\
&= j + \text{out}(i) - i - \text{out}(j) \geq j - i - 1 \geq 0.
\end{aligned}$$

□

With Lemmas 4.2 and 4.4 we can prove the next lemma, which completes the proof of the correctness of our reduction.

Lemma 4.5. *If $\mathcal{S}_{\mathcal{A},B}$ is a yes-instance of* DAG Realization*, then $(\mathcal{A}, B)$ is a yes-instance of* 3-Partition*.*

Proof. Let $D = (V, A)$ be a realizing DAG of the degree sequence $\mathcal{S}_{\mathcal{A},B}$ with a topological ordering ϕ. Let v_j be the x-vertex realizing x_j and let u_i be the a-vertex realizing $\binom{a_i}{a_i}$. Denote for each vertex v by $\mathrm{pos}_\phi(v)$ the position of v in ϕ. From Lemma 4.2 it follows that the x-vertices form a transitive tournament. Furthermore, we can assume by Lemma 4.4 that $\mathrm{pos}_\phi(v_j) < \mathrm{pos}_\phi(v_k)$ for all $0 \leq j < k < 2(p+1)B$. Hence, it follows from the in- and outdegrees of the x-vertices that each x-vertex in V_i^r, $0 \leq i < p$, has exactly one outgoing arc to an a-vertex and each x-vertex in V_i^ℓ, $1 \leq i \leq p$, has exactly one incoming arc from an a-vertex. Hence, we can assume that there are a-vertices $u_{i_1}, u_{i_2}, \ldots, u_{i_k}$ with $\mathrm{pos}_\phi(v) < \mathrm{pos}_\phi(u_{i_j}) < \mathrm{pos}_\phi(v')$ for all $v \in V_0^r$ and $v' \in V_1^\ell$. As B vertices from V_0^r have an outgoing arc to these a-vertices, it follows that $\sum_{j=1}^k a_{i_j} \leq B$. As each a-vertex has an indegree equal to its outdegree and each of the B vertices in V_1^ℓ requires one incoming arc from an a-vertex, it follows that $\sum_{j=1}^k a_{i_j} \geq B$. Hence, $\sum_{j=1}^k a_{i_j} = B$. Since $B/4 < a_j < B/2$ for all $1 \leq j \leq 3p$, it follows that $k = 3$.

The vertices in V_1^r also have no incoming arc from an a-vertex but each of them has an outgoing arc to an a-vertex. Also, each of the vertices in V_2^ℓ needs one incoming arc from an a-vertex. Thus, we can assume that in the topological ordering ϕ of D there are three a-vertices between the vertices of V_1^r and V_2^ℓ such that their indegrees and also their outdegrees sum up to B. Analogously, it follows for all $1 \leq i < p$ that there are three a-vertices $u_{j_1^i}, u_{j_2^i}, u_{j_3^i}$ with

$$\mathrm{pos}_\phi(v) < \mathrm{pos}_\phi(u_{j_1^i}) < \mathrm{pos}_\phi(u_{j_2^i}) < \mathrm{pos}_\phi(u_{j_3^i}) < \mathrm{pos}_\phi(v')$$

for all $v \in V_i^r$ and $v' \in V_{i+1}^r$, and $\sum_{\ell=1}^3 a_{j_\ell^i} = B$. Thus, $(\mathcal{A}, B)$ is a yes-instance of 3-Partition. □

Our construction together with Lemmas 4.1 and 4.5 yields the NP-hardness of DAG Realization. As mentioned in the beginning of this section containment in NP is easy to obtain. Hence, we arrive at the following theorem.

Theorem 4.6. DAG Realization *is NP-complete.*

Theorem 4.6 proves that, unless P = NP, DAG Realization is not solvable in polynomial time. However, Berger and Müller-Hannemann [BM11] identified special cases of DAG Realization that can be solved in polynomial time; for instance, this is the case when the degree sequence can be linearly ordered with respect to the opposed order. Moreover, Berger and Müller-Hannemann

[BM12] showed that DAG Realization is polynomial-time solvable when the number m of arcs is less than the number n of vertices. The positive results together with the general NP-hardness of DAG Realization motivate a parameterized complexity analysis of the problem meaning to perform a more fine-grained complexity analysis with respect to various parameters [FJR13, KNU11, Nie10]. Thereby, the general target is to identify certain "quantities" or parameters whose restriction allows the problem to be solved efficiently. To this end, observe that in our NP-hardness proof construction the resulting DAG Realization instance contains $\Theta(n^2)$ arcs and that the problem is polynomial-time solvable if $m < n$. Thus, we consider the "sparseness" of the instance, measured with the "quantities" m/n or $m - n + 1$. Unfortunately, the next theorem proves that, unless P = NP, the parameter m/n is not helpful. Furthermore, it shows that although the problem is easy if $m = n(n-1)/2$ (the only realizing DAG is a transitive tournament), the problem becomes hard if m is "almost" $n(n-1)/2$.

Theorem 4.7. *For every constant $\ell > 1$,* DAG Realization *remains NP-complete even if $m < \ell n$ or if $m > n(n-1)/(2\ell)$.*

Proof. We prove both results with simple padding arguments. To this end, let $\ell > 1$ be some constant and let $\mathcal{S} = \left\{ \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n} \right\}$ be an arbitrary instance of DAG Realization. We will modify $\mathcal{S}$ in order to obtain an instance with the desired ratios between m and n. We denote the modified instance with $\mathcal{S}'$ which is initialized as a copy of $\mathcal{S}$.

First, we describe the modifications for the case $m > n(n-1)/(2\ell)$: We repeatedly add universal sources one after the other to $\mathcal{S}'$, that is, we add a tuple $\binom{0}{n'}$ to $\mathcal{S}'$ and increase the first component of the other tuples of $\mathcal{S}'$ by one, that is, replace $\binom{a}{b} \in \mathcal{S}'$ by $\binom{a+1}{b}$. Notice that once a universal source is added to $\mathcal{S}'$ all former sources become non-sources; in particular, if a second universal source is added, then the first added universal source is no longer a source. We keep adding universal sources until $m' > n'(n'-1)/(2\ell)$.

Observe that adding a universal source to a DAG Realization instance results in an equivalent instance: The vertex realizing the universal source has to have outgoing arcs to each other vertex in any realizing DAG. Thus, for every realizing DAG for the new instance the subgraph induced by all vertices not realizing the universal source forms a realizing DAG for the original instance. Conversely, any realizing DAG for the old instance can be easily extended to a realizing DAG for the new instance. Thus, the constructed instance $\mathcal{S}'$ is

a yes-instance if and only if $\mathcal{S}$ is a yes-instance. Hence, DAG Realization remains NP-complete even if $m > n(n-1)/(2\ell)$.

Second, for the case $m < \ell n$ we do the following: We add some number of $\binom{0}{0}$-tuples to $\mathcal{S}'$ and then add one universal source turning the $\binom{0}{0}$-tuples into $\binom{1}{0}$-sinks. By choosing the appropriate amount of $\binom{0}{0}$-tuples the modified instance $\mathcal{S}'$ satisfies $m' < \ell n'$. Furthermore, as adding a $\binom{0}{0}$-tuple results in an equivalent instance, it follows that the constructed instance $\mathcal{S}'$ is a yes-instance if and only if $\mathcal{S}$ is a yes-instance. Hence, DAG Realization remains NP-complete even if $m < \ell n$. □

Complementing Theorem 4.7, we show in Section 4.4 that DAG Realization is fixed-parameter tractable with respect to each of the two parameters $m-n+1$ and $\binom{n}{2} - m$. Besides the above mentioned parameters, the maximum degree Δ is unbounded in our NP-hardness proof. Hence it is a good candidate for further parameterized investigations. Indeed, we show in the next section that DAG Realization is linear-time solvable for constant maximum degree.

4.3. Fixed-Parameter Tractability with Respect to Maximum Degree

Let $\Delta := \max\{a_1, b_1, a_2, b_2, \ldots, a_n, b_n\}$ denote the maximum degree in a degree sequence. In this section, we show that DAG Realization is fixed-parameter tractable with respect to the parameter Δ. To this end, we assume that Δ is some fixed value and all degree sequences considered in this section have a maximum degree of Δ.

A high-level description of our approach is as follows: First, we show that we can reorder any realizable degree ordering for our given input instance $\mathcal{S}$ so that it has a certain structure. Second, our algorithm branches into all possible orderings of the tuples in $\mathcal{S}$ satisfying this structure and returns "yes" if in at least one branch the considered ordering is indeed a realizable degree ordering, otherwise it returns "no". Herein, we will distinguish two cases for the sought structures. In both cases, however, we use the same reordering operations and their description makes up the major part of Section 4.3.1. To describe how we reorder realizable degree orderings, we need the following central definition.

Definition 4.2. Let $\phi = v_1, v_2, \ldots, v_n$ be a topological ordering of a DAG D. For all $0 \leq i \leq n$, the *potential* at position i is a vector $p_i^\phi \in \mathbb{N}^\Delta$ where $p_i^\phi[\ell]$ for

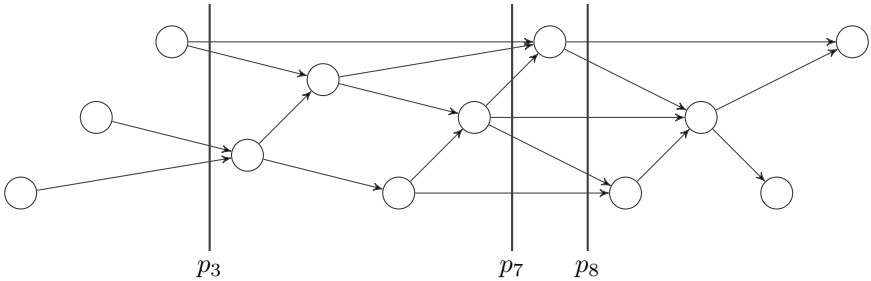


Figure 4.2.: A realizing DAG for the degree sequence $\mathcal{S} = \{\binom{0}{1}, \binom{0}{1}, \binom{0}{2}, \binom{2}{2}, \binom{2}{2}, \binom{1}{2}, \binom{2}{3}, \binom{3}{2}, \binom{2}{1}, \binom{3}{2}, \binom{2}{0}, \binom{1}{0}\}$. Here we have $\Delta = 3$. The highlighted potentials are as follows: $p_3 = (3, 1, 0)$, $p_7 = (4, 1, 1)$, and $p_8 = (3, 2, 0)$. Their values (number of arcs crossing the line indicating the potential) are: $\omega(p_3) = 4$, $\omega(p_7) = 6$, and $\omega(p_8) = 5$.

$1 \leq \ell \leq \Delta$ is the number of vertices in the subsequence $v_1, v_2, \ldots, v_i$ that have in D at least ℓ outneighbors in the subsequence $v_{i+1}, v_{i+2}, \ldots, v_n$. The *value* of a potential p_i^ϕ is $\omega(p_i^\phi) := \sum_{\ell=1}^{\Delta} p_i^\phi[\ell]$.

See Figure 4.2 for an example. If the DAG D and the topological ordering ϕ are clear from the context, then we write p_i instead of p_i^ϕ. We denote with 0^Δ the potential of value zero, for example, it holds that $p_0 = p_n = 0^\Delta$. To understand the role of potentials in the reordering operation consider a topological ordering $\phi = v_1, v_2, \ldots, v_n$ where at two positions $0 < i < j < n$ the potentials are equal, that is $p_i^\sigma = p_j^\sigma$. We will show in Section 4.3.1 that we can cut out the vertices $v_{i+1}, v_{i+2}, \ldots, v_j$, that is, $\phi' = v_1, v_2, \ldots, v_i, v_{j+1}, v_{j+2}, \ldots, v_n$ is also a topological ordering.

In order to give some intuition about potentials we now make a few general observations. First, observe that the value of a potential $\omega(p_i)$ is just the number of arcs with tail in $\{v_1, v_2, \ldots, v_i\}$ and head in $\{v_{i+1}, v_{i+2}, \ldots, v_n\}$; for example, in Figure 4.2 four arcs "cross" the third position and hence $\omega(p_3) = 4$. Since the number of arcs is determined by the degrees of the vertices, the value of the potential at position $i + 1$ can be determined by the potential at position i and the vertex v_{i+1} at position $i + 1$: As v_{i+1} "absorbs" $\deg^-(v_{i+1})$ arcs and "contributes" $\deg^+(v_{i+1})$ arcs to the following vertices, the value of the potential at position $i + 1$ is $\omega(p_{i+1}) = \omega(p_i) - \deg^-(v_{i+1}) + \deg^+(v_{i+1})$. Generalizing this yields the following.

Observation 4.8. *Let $\phi = v_1, v_2, \ldots, v_n$ be a topological ordering of a DAG D and let $1 \leq i < j \leq n$ be two integers. Then it holds that $\omega(p_j) = \omega(p_i) + \sum_{\ell=i+1}^{j}(\deg^+(v_\ell) - \deg^-(v_\ell))$.*

Second, we remark that a potential stores more information than just the number of arcs "crossing" some position: The potential p_i stores *all* information about *how many* vertices from $\{v_1, v_2, \ldots, v_i\}$ have *how many* outgoing arcs to $\{v_{i+1}, v_{i+2}, \ldots, v_n\}$: In particular, for each $1 \leq j < \Delta$ there are $p_i[j] - p_i[j+1]$ vertices in $\{v_1, v_2, \ldots, v_i\}$ that have exactly j neighbors in $\{v_{i+1}, v_{i+2}, \ldots, v_n\}$. Thus, for any potential $p_i \in \mathbb{N}^\Delta$ at any position $i \in \mathbb{N}$, it holds that $p_i[j] \geq p_i[j+1]$ for all $1 \leq j < \Delta$.

Algorithm outline. Our algorithm consists of two parts. First, as described in Section 4.3.2, the algorithm checks whether the DAG Realization instance admits a "high-potential realization" where at any position the value of the potential is at least Δ^2. If no high-potential realization is found, then, by exploiting the fact that the value of all potentials is upper-bounded, the algorithm checks whether a "low-potential realization" exists; see Section 4.3.3 for the description.

4.3.1. General Terms and Observations

In this section, we provide some general notations, observations, and lemmas leading to the reordering operation we use in the algorithms to find high-potential as well as low-potential realizations.

Notation. For a topological ordering $\phi = v_1, v_2, \ldots, v_n$ and two indices $1 \leq i \leq j \leq n$ we set $\phi[i,j] := v_i, v_{i+1}, \ldots, v_j$. The vertex set $\{v_i, v_{i+1}, \ldots, v_j\}$ is denoted by $\phi\{i,j\}$. Analogously, for an ordering $\sigma = \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ of a degree sequence $\mathcal{S}$ we set $\sigma[i,j] := \binom{a_i}{b_i}, \binom{a_{i+1}}{b_{i+1}}, \ldots, \binom{a_j}{b_j}$ and we denote the multiset $\left\{\binom{a_i}{b_i}, \binom{a_{i+1}}{b_{i+1}}, \ldots, \binom{a_j}{b_j}\right\}$ by $\sigma\{i,j\}$. For two topological orderings $\phi = v_1, v_2, \ldots, v_n$ and $\phi' = v'_1, v'_2, \ldots, v'_{n'}$ we set $\phi\phi' := v_1, v_2, \ldots, v_n, v'_1, v'_2, \ldots, v'_{n'}$. Similarly, for two orderings $\sigma = \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ and $\sigma' = \binom{a'_1}{b'_1}, \binom{a'_2}{b'_2}, \ldots, \binom{a'_{n'}}{b'_{n'}}$ we set $\sigma\sigma' := \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}, \binom{a'_1}{b'_1}, \binom{a'_2}{b'_2}, \ldots, \binom{a'_{n'}}{b'_{n'}}$.

Definition 4.3. Two tuples $\binom{a}{b}$ and $\binom{a'}{b'}$ are of the same *type* if $\binom{a}{b} = \binom{a'}{b'}$. Furthermore, $\binom{a}{b}$ is a *good type* tuple if $a \leq b$ and otherwise it is a *bad type* tuple. Two vertices are of the same type, if they have the same in- and the same outdegrees. Correspondingly, a vertex v is a *good type* vertex if $\deg^-(v) \leq \deg^+(v)$ and otherwise it is a *bad type* vertex.

Note that the input degree sequence contains at most $(\Delta + 1)^2$ different types.

Well-connected DAGs. Berger and Müller-Hannemann [BM12] showed that, given an ordering of a degree sequence, one can check in polynomial time whether this ordering is a realizable degree ordering. To prove this, the main observation is that for any realizable degree ordering one can construct at least one corresponding DAG by *well-connecting* consecutive vertices in a corresponding topological ordering.

Definition 4.4. Let D be a DAG with a topological ordering $\phi[1, n]$. The *remaining outdegree* of vertex v_i at position j, $1 \leq i \leq j < n$, is the number of v_i's neighbors in the subsequence $\phi[j + 1, n]$. Furthermore, D is *well-connected* with respect to ϕ if for all vertices $v_i \in \phi\{1, n\}$ it holds that v_i's inneighbors are the $\deg^-(v_i)$ vertices in $\phi[1, i - 1]$ that have the highest remaining outdegree at position $i - 1$.

Observe that the potential p_i^ϕ at position i stores the remaining outdegrees of all vertices $v_1, v_2, \ldots, v_i$ at position i. In the following, we omit ϕ and just write that D is well-connected when the corresponding topological ordering ϕ is clear from the context or implicitly given by a realizable degree ordering corresponding to D. Furthermore, during the construction of a DAG corresponding to a realizable degree ordering we write that we well-connect the vertex v_i as an abbreviation for making the $\deg^-(v_i)$ vertices with the highest remaining outdegree at position $i - 1$ inneighbors of v_i. Berger and Müller-Hannemann [BM12] showed how to construct a well-connected DAG realizing a given realizable degree ordering; see Berger [Ber11] for the complete proof of correctness.

Lemma 4.9 ([Ber11, Theorem 4.1] and [BM12, Lemma 1])**.** *Let σ be a realizable degree ordering. Then, there exists a realizing well-connected DAG D that admits a topological ordering ϕ corresponding to σ.*

We use Lemma 4.9 as follows: It paves the way to a simple algorithm checking whether a given ordering of a degree sequence $\mathcal{S}$ is indeed a realizable degree

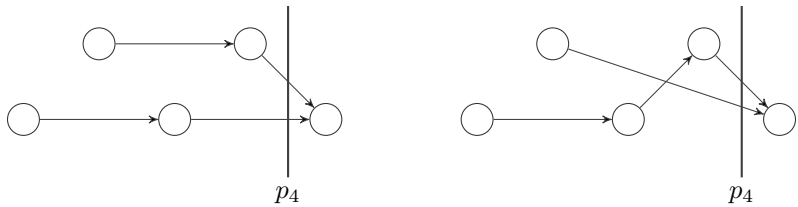


Figure 4.3.: Two non-isomorphic well-connected DAGs realizing the realizable degree ordering $\binom{0}{1}, \binom{0}{1}, \binom{1}{1}, \binom{1}{1}, \binom{2}{0}$. Observe that for each position the two potentials at this position are the same, for example, in both graphs the potential at position four is $p_4 = (2, 0)$.

ordering: The algorithm iteratively adds the vertices according to the given ordering and well-connects each added vertex, thus working similar to the algorithm of Kleitman and Wang [KW73], see Algorithm 3.2 on page 36.

Lemma 4.10. *Given an ordered degree sequence σ, one can decide in $\mathcal{O}(\Delta n)$ time whether σ is a realizable degree ordering.*

Proof. As mentioned above, the algorithm constructs the DAG stepwise by iterating over σ, adding a vertex v for the currently considered tuple, and well-connecting v. To this end, the algorithm uses Δ lists, where the ith list stores all vertices having remaining outdegree i at the currently considered position. By virtually concatenating the Δ lists in $\mathcal{O}(\Delta)$ time and then iterating over the first Δ elements one can determine in $\mathcal{O}(\Delta)$ time up to Δ vertices with highest outdegrees. Hence, well-connecting a vertex v can be done in $\mathcal{O}(\Delta)$ time and decreasing the remaining outdegree of the $\deg^-(v) \leq \Delta$ inneighbors of v by one can also be done in $\mathcal{O}(\Delta)$ time. Furthermore, inserting v into these lists can be done in constant time. Hence, with n elements in the given realizable degree ordering σ one can decide in $\mathcal{O}(\Delta n)$ time whether σ is a realizable degree ordering. □

In the following it will be important that Lemma 4.9 allows us to assume that, given a realizable degree ordering σ, the corresponding realizing DAG is well-connected if not explicitly stated otherwise. This allows us to define the potential p_i^σ of σ at position i as the potential at position i in a topological ordering of a well-connected DAG corresponding to σ. Note that there might be more than one well-connected DAG realizing σ as there might be multiple

vertices with the highest outdegree at some position, see Figure 4.3 for an example. However, the potential p_i^σ is indeed well-defined as for each $x \in \mathbb{N}$ the number of vertices with remaining outdegree x at position i is the same for all well-connected DAGs realizing σ.

Reordering realizable degree orderings. Given a realizable degree ordering, cutting out subsequences and reinserting them appropriately is the main operation that we perform to reorder the degree sequence such that we can exploit the resulting regular structure in our algorithm. Basically, if a certain potential p occurs twice in a realizable degree ordering, then removing the subsequence between them results also in a realizable degree ordering. Furthermore, this subsequence can be reinserted at any position where the potential p occurs. In the following we give a formal description of this operation. To this end, we link subsequences to the two potentials that appear at the cut-positions in the realizable degree ordering as these potentials obviously do not depend solely on the subsequences but on the whole realizable degree ordering. In this way, the following definition formalizes the potentials that may fit to a subsequence if the rest of the realizable degree ordering is chosen accordingly.

Definition 4.5. Let $\sigma[1, n]$ be a realizable degree ordering and let $1 \leq i < j \leq n$ be two integers. Then $\sigma[i, j]$ is a *partial realizable degree ordering* with *input potential* p_{i-1}^σ and *output potential* p_j^σ.

Observe that a partial realizable degree ordering with input and output potential 0^Δ is also a realizable degree ordering. A first observation towards cutting and merging partial realizable degree orderings is that we can "append" a tuple $\binom{a}{b}$ to a partial realizable degree orderings σ when at the end of σ at least a vertices require at least one more outneighbor. Furthermore, the information about the number of vertices with remaining outdegree at least one is stored in the first entry of the corresponding potential. Combining this with Observation 4.8 we arrive at the following.

Observation 4.11. *Let $\sigma = \binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}$ be a partial realizable degree ordering with input potential p_0^σ and output potential p_n^σ, and let $a, b \in \mathbb{N}$. If $a \leq p_n^\sigma[1]$, then there exists a potential $p \in \mathbb{N}^\Delta$ with $\omega(p) = \omega(p_n^\sigma) - a + b$ such that $\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}, \binom{a}{b}$ is a partial realizable degree ordering with input potential p_0^σ and output potential p.*

Proof. Let $\widetilde{\sigma}[1, \widetilde{n}]$ be a realizable degree ordering corresponding to σ, that is, there are integers $1 \leq i < j \leq \widetilde{n}$ such that $\sigma = \widetilde{\sigma}_1[i, j]$, $p_0^\sigma = p_{i-1}^{\widetilde{\sigma}}$, and $p_n^\sigma = p_j^{\widetilde{\sigma}}$.

Note that, by Definition 4.5, $\widetilde{\sigma}$, i, and j exist. Let D be a realizing DAG for $\widetilde{\sigma}$. Now, remove all vertices in D that correspond to tuples in $\widetilde{\sigma}[j+1, \widetilde{n}]$. Next, add to D the vertex v corresponding to $\binom{a}{b}$ and well-connect v. This is possible since $a \leq p_n^{\sigma}[1]$. Finally, repeatedly add sinks of type $\binom{1}{0}$ as long as D contains vertices whose out-degree does not fit with the corresponding tuples. The resulting DAG proves that $\binom{a_1}{b_1}, \binom{a_2}{b_2}, \ldots, \binom{a_n}{b_n}, \binom{a}{b}$ is indeed a partial realizable degree ordering. The claimed value of the output potential follows from Observation 4.8. □

Observation 4.11 indicates when we can add a tuple $\binom{a}{b}$ and that the value of the output potential changes by $b - a$. However, in order to append complete partial realizable degree orderings, we have to impose some restriction on the potential. To this end, we show that we can "merge" two partial realizable degree orderings σ_1 and σ_2 to $\sigma_1\sigma_2$ when the output potential of σ_1 is "better" than the input potential of σ_2. To formalize what it means to be better we introduce a partial order $\succeq$ for potentials.

Definition 4.6. For $p, p' \in \mathbb{N}^{\Delta}$, $p \succeq p'$ if $\forall 1 \leq j \leq \Delta\colon \sum_{i=1}^{j} p[i] \geq \sum_{i=1}^{j} p'[i]$.

Intuitively, a "bad" potential value represents "few" vertices with "high" outdegree. These vertices can only be connected to many vertices with low indegree. On the contrary, a "good" potential represents "many" vertices with "low" outdegree. These vertices can be connected to many vertices with low indegree *or* to few vertices with high indegree. So a good potential at some position indicates a high freedom for connecting the succeeding vertices. Indeed, a potential p that is better than a potential p' guarantees that all subsequent vertices that can be appended with potential p' can also be appended with potential p, as shown in the next proposition.

Proposition 4.12. *Let $\sigma_1[1, n_1]$ be a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1}^{\sigma_1} \in \mathbb{N}^{\Delta}$ and let $\sigma_2[1, n_2]$ be a partial realizable degree ordering with input potential $p_0^{\sigma_2}$ and output potential $p_{n_2}^{\sigma_2}$ such that $p_{n_1}^{\sigma_1} \succeq p_0^{\sigma_2}$. Then, $\sigma = \sigma_1\sigma_2$ is a partial realizable degree ordering with input potential $p_0^{\sigma} = p_0^{\sigma_1}$ and output potential $p_{n_1+n_2}^{\sigma} \succeq p_{n_2}^{\sigma_2}$ such that $p_i^{\sigma} = p_i^{\sigma_1}$ for all $1 \leq i \leq n_1$ and $p_i^{\sigma} \succeq p_{i-n_1}^{\sigma_2}$ for all $n_1 + 1 \leq i \leq n_2 + n_1$. If additionally $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2})$ and $p_0^{\sigma_1} = p_{n_2}^{\sigma_2} = 0^{\Delta}$, then σ is also a realizable degree ordering.*

The proof of Proposition 4.12 is based on the following lemma dealing with the case $n_2 = 1$.

Lemma 4.13. *Let $\sigma_1[1, n_1]$ be a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1}^{\sigma_1} \in \mathbb{N}^\Delta$ and let $\sigma_2 = \binom{a}{b}$, $a, b \in \mathbb{N}$, be a partial realizable degree ordering with input potential $p_0^{\sigma_2}$ and output potential $p_1^{\sigma_2}$ such that $p_{n_1}^{\sigma_1} \succeq p_0^{\sigma_2}$. Then, $\sigma = \sigma_1\sigma_2$ is a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1+1}^{\sigma} \succeq p_1^{\sigma_2}$ such that $p_i^{\sigma} = p_i^{\sigma_1}$ for all $1 \leq i \leq n_1$.*

Proof. Let $\widetilde{\sigma}_1[1, \widetilde{n}_1]$ be the realizable degree ordering corresponding to σ_1, that is, there are integers $1 \leq i < j \leq \widetilde{n}_1$ such that $\sigma_1 = \widetilde{\sigma}_1[i, j]$, $p_0^{\sigma_1} = p_{i-1}^{\widetilde{\sigma}_1}$, and $p_{n_1}^{\sigma_1} = p_j^{\widetilde{\sigma}_1}$. Note that, by Definition 4.5, $\widetilde{\sigma}_1$, i, and j exist. Furthermore, let $\hat{\sigma}$ be an ordering of $\omega(p_{n_1}^{\sigma_1}) - a + b$ sinks of type $\binom{1}{0}$. In order to prove the lemma we will show that $\widetilde{\sigma} = \widetilde{\sigma}_1[1, j]\sigma_2\hat{\sigma}$ is a realizable degree ordering such that $p_{i-1}^{\widetilde{\sigma}} = p_0^{\sigma_1}$ and $p_{j+1}^{\widetilde{\sigma}} \succeq p_1^{\sigma_2}$.

First, we show that $\widetilde{\sigma} = \widetilde{\sigma}_1[1, j]\sigma_2\hat{\sigma}$ is indeed a realizable degree ordering. To this end, note that, since $p_{n_1}^{\sigma_1} \succeq p_0^{\sigma_2}$, it holds that $p_{n_1}^{\sigma_1}[1] \geq a$. Hence, by Observation 4.11, $\widetilde{\sigma}_1[1, j]\sigma_2$ is a partial realizable degree ordering with input potential 0^Δ and output potential $p \in \mathbb{N}^\Delta$ such that $\omega(p) = \omega(p_{n_1}^{\sigma_1}) - a + b$. As $\hat{\sigma}$ contains exactly $\omega(p)$ sinks of type $\binom{0}{1}$, it follows that $\widetilde{\sigma}$ is a realizable degree ordering. Furthermore, since the first part remains unchanged it follows that $p_{i-1+\ell}^{\widetilde{\sigma}} = p_\ell^{\sigma_1}$ for all $1 \leq \ell \leq n_1$.

We now complete the proof by showing that $p_{j+1}^{\widetilde{\sigma}} \succeq p_1^{\sigma_2}$. To this end, we assume that $b = 0$. In the case that $b > 0$, we denote by $c \in \mathbb{N}^\Delta$ the vector having ones in the first b entries and zeros in the remaining entries. Intuitively, c is the "emission" the vertex v realizing $\binom{a}{b}$ adds to both potentials $p_{j+1}^{\widetilde{\sigma}}$ and $p_1^{\sigma_2}$. As this emission is the same for both potentials, we subtract c from $p_{j+1}^{\widetilde{\sigma}}$ and from $p_1^{\sigma_2}$ to obtain the potentials only containing the "absorption" of v. To measure the effect of the "absorption" we now prove for every $\ell \in \{1, 2, \ldots, \Delta\}$ that

$$\sum_{i=1}^{\ell} p_j^{\widetilde{\sigma}}[i] - p_{j+1}^{\widetilde{\sigma}}[i] = \max\{0, a - p_j^{\widetilde{\sigma}}[\ell + 1]\}, \tag{4.1}$$

where in slight abuse of the notation of potentials we set $p_j^{\widetilde{\sigma}}[\Delta + 1] := 0$. Note that this fits the intuition as no vertex has more than Δ outneighbors. We now show the correctness of Equation (4.1): The vertex v has the a vertices with highest remaining outdegree at position j as inneighbors. Thus, for $\ell = \Delta$ the difference on the left-hand side is exactly a. Furthermore, $p_j^{\widetilde{\sigma}}[\ell + 1]$ denotes the

number of vertices that have at position j a remaining outdegree of at least $\ell+1$. If $a \leq p_j^{\widetilde{\sigma}}[\ell+1]$, then clearly the vertices with remaining outdegree of at most ℓ at position j are not adjacent to v and, therefore, the difference on the left-hand side remains unchanged. Conversely, if $a > p_j^{\widetilde{\sigma}}[\ell+1]$, then $a - p_j^{\widetilde{\sigma}}[\ell+1]$ of the vertices with remaining outdegree at most ℓ at position j are adjacent to v. Thus, the difference on the left-hand side is in this case exactly $a - p_j^{\widetilde{\sigma}}[\ell+1]$.

We now use Equation (4.1) to show $p_{j+1}^{\widetilde{\sigma}} \succeq p_1^{\sigma_2}$. First, for every $\ell \in \{1, 2, \ldots, \Delta\}$ we have

$$\sum_{i=1}^{\ell} p_{j+1}^{\widetilde{\sigma}}[i] \overset{(4.1)}{=} \sum_{i=1}^{\ell} p_j^{\widetilde{\sigma}}[i] - \max\{0, a - p_j^{\widetilde{\sigma}}[\ell+1]\} \tag{4.2}$$

If $p_j^{\widetilde{\sigma}}[\ell+1] \geq a$, then we obtain

$$\sum_{i=1}^{\ell} p_{j+1}^{\widetilde{\sigma}}[i] = \sum_{i=1}^{\ell} p_j^{\widetilde{\sigma}}[i] \geq \sum_{i=1}^{\ell} p_0^{\sigma_2}[i] \geq \sum_{i=1}^{\ell} p_1^{\sigma_2}[i], \tag{4.3}$$

as by assumption we have $p_j^{\widetilde{\sigma}} \succeq p_0^{\sigma_2}$ and clearly $p_0^{\sigma_2} \succeq p_1^{\sigma_2}$. In the remaining case of $p_j^{\widetilde{\sigma}}[\ell+1] < a$, we obtain

$$\begin{aligned}
\sum_{i=1}^{\ell} p_{j+1}^{\widetilde{\sigma}}[i] &\overset{(4.2)}{=} \sum_{i=1}^{\ell} p_j^{\widetilde{\sigma}}[i] - (a - p_j^{\widetilde{\sigma}}[\ell+1]) = \sum_{i=1}^{\ell+1} p_j^{\widetilde{\sigma}}[i] - a \geq \sum_{i=1}^{\ell+1} p_0^{\sigma_2}[i] - a \\
&= \sum_{i=1}^{\ell} p_0^{\sigma_2}[i] - (a - p_0^{\sigma_2}[\ell+1]) \geq \sum_{i=1}^{\ell} p_0^{\sigma_2}[i] - \max\{0, a - p_0^{\sigma_2}[\ell+1]\} \\
&\overset{(4.1)}{=} \sum_{i=1}^{\ell} p_1^{\sigma_2}[i].
\end{aligned} \tag{4.4}$$

Observe that in Equation (4.4) we require Equation (4.1) to also work with $p_0^{\sigma_2}$ and $p_1^{\sigma_2}$, which is true by an analogous argument as for $p_j^{\widetilde{\sigma}}$ and $p_{j+1}^{\widetilde{\sigma}}[i]$. Since Inequality (4.3) and Equation (4.4) hold for every $\ell \in \{1, 2, \ldots, \Delta\}$, it follows that $p_{j+1}^{\widetilde{\sigma}} \succeq p_1^{\sigma_2}$. □

Now, we prove Proposition 4.12 by repeatedly invoking Lemma 4.13.

Proof. (of Proposition 4.12) We prove the first statement of the proposition with an induction invoking Lemma 4.13 in the induction step. For the base case

observe that by Lemma 4.13, $\sigma[1, n_1+1] = \sigma_1[1, n_1]\sigma_2[1, 1]$ is a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1+1}^{\sigma} \succeq p_1^{\sigma_2}$. For the induction step, let $j \in \{1, 2, \ldots, n_2 - 1\}$. If $\sigma[1, n_1 + j]$ is a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1+j}^{\sigma} \succeq p_j^{\sigma_2}$ such that $p_i^{\sigma} = p_i^{\sigma_1}$ for all $1 \leq i \leq n_1$ and $p_i^{\sigma} \succeq p_{i-n_1}^{\sigma_2}$ for all $n_1 + 1 \leq i \leq n_1 + j$, then, by Lemma 4.13, $\sigma[1, n_1 + j + 1]$ is a partial realizable degree ordering with input potential $p_0^{\sigma_1}$ and output potential $p_{n_1+j+1}^{\sigma} \succeq p_{j+1}^{\sigma_2}$ such that $p_i^{\sigma} = p_i^{\sigma_1}$ for all $1 \leq i \leq n_1$ and $p_i^{\sigma} \succeq p_{i-n_1}^{\sigma_2}$ for all $n_1 + 1 \leq i \leq n_1 + j + 1$. This proves the first statement of the proposition.

For the second statement observe that if $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2})$, then $\omega(p_{n_1+n_2}^{\sigma}) = \omega(p_{n_2}^{\sigma_2})$. Thus, if $p_0^{\sigma_1} = p_{n_2}^{\sigma_2} = 0^{\Delta}$, then σ is a partial realizable degree ordering with input and output potential 0^{Δ}. Hence, σ is in this case a realizable degree ordering. □

Proposition 4.12 shows that we can "merge" two partial realizable degree orderings σ_1 and σ_2 to $\sigma_1\sigma_2$ if for the output potential $p_{n_1}^{\sigma_1}$ of σ_1 and the input potential $p_0^{\sigma_2}$ it holds that $p_{n_1}^{\sigma_1} \succeq p_0^{\sigma_2}$ and $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2})$. This provides the basis for cutting out a subsequence in a realizable degree ordering and reinserting it at another position. First, consider the cutting out of subsequences.

Lemma 4.14. *Let $\sigma[1, n]$ be a realizable degree ordering. If there are two indices $1 \leq i < j \leq n$ such that $\omega(p_i^{\sigma}) = \omega(p_j^{\sigma})$ and $p_i^{\sigma} \succeq p_j^{\sigma}$, then $\sigma' = \sigma[1, i]\sigma[j + 1, n]$ is a realizable degree ordering with $p_{i+\ell}^{\sigma'} \succeq p_{j+\ell}^{\sigma}$ for all $1 \leq \ell \leq n - j$.*

Proof. Let $\sigma[1, n]$ be a realizable degree ordering and let $1 \leq i < j \leq n$ be two indices such that $\omega(p_i^{\sigma}) = \omega(p_j^{\sigma})$ and $p_i^{\sigma} \succeq p_j^{\sigma}$. First, by definition, $\sigma[1, i]$ is a partial realizable degree ordering with input potential 0^{Δ} and output potential p_i^{σ}. Furthermore, $\sigma[j + 1, n]$ is a partial realizable degree ordering with input potential p_j^{σ} and output potential 0^{Δ}. Hence, since $\omega(p_i^{\sigma}) = \omega(p_j^{\sigma})$ and $p_i^{\sigma} \succeq p_j^{\sigma}$, it follows from Proposition 4.12 that $\sigma' = \sigma[1, i]\sigma[j + 1, n]$ is a realizable degree ordering and $p_{i+\ell}^{\sigma'} \succeq p_{j+\ell}^{\sigma}$ for all $1 \leq \ell \leq n - j$. □

Lemma 4.14 shows that from a realizable degree ordering σ we can cut out a subsequence $\sigma[i + 1, j]$ whenever $p_i^{\sigma} = p_j^{\sigma}$. The next observation shows that we can reinsert this subsequence in the remaining realizable degree ordering σ' at any position ℓ with $p_{\ell}^{\sigma'} = p_i^{\sigma} = p_j^{\sigma}$.

Lemma 4.15. *Let $\sigma_1[1, n_1]$ be a realizable degree ordering. Furthermore, let $\sigma_2[1, n_2]$ be a partial realizable degree ordering with input and output potential p. Then, for all indices $1 \leq i \leq n_1$ where $p_i^{\sigma_1} = p$, the ordering*

$\sigma = \sigma_1[1,i]\sigma_2[1,n_2]\sigma[i+1,n_1]$ *is a realizable degree ordering with* $p_j^{\sigma_1} \succeq p_{j+n_2}^{\sigma}$ *for all* $i < j \leq n_1$.

Proof. Let $\sigma_1[1,n_1]$ be a realizable degree ordering and let $\sigma_2[1,n_2]$ be a partial realizable degree ordering with input and output potential $p \in \mathbb{N}^{\Delta}$. Furthermore, let i be a position in σ_1 such that $p_i^{\sigma_1} = p$. Then, by Proposition 4.12, $\sigma_1[1,i]\sigma_2[1,n_2]$ is a partial realizable degree ordering with input potential 0^{Δ} and output potential $p' \succeq p$. Since the input and output potential of σ_2 are equal, it follows that $\omega(p') = \omega(p_i^{\sigma_1}) = \omega(p)$. Hence, again applying Proposition 4.12, it follows that $\sigma = \sigma_1[1,i]\sigma_2[1,n_2]\sigma_1[i+1,n_1]$ is a realizable degree ordering with $p_j^{\sigma_1} \succeq p_{j+n_2}^{\sigma}$ for all $i < j \leq n_1$. □

Given a realizable degree ordering σ where at the three positions i, j, k the same potential occurs, we can now cut out (Lemma 4.14) the part between positions i and j and then insert (Lemma 4.15) it at position k. The following proposition formalizes this "reordering operation".

Proposition 4.16. *Let* $\sigma[1,n]$ *be a realizable degree ordering and let* $1 \leq i < j < k \leq n$ *be three positions with* $p_i = p_j = p_k$. *Then* $\sigma[1,i]\sigma[j+1,k]\sigma[i+1,j]\sigma[k+1,n]$ *is a realizable degree ordering.*

4.3.2. High-Potential Sequences

In this section, we show that if a realizable degree sequence admits a realizable degree ordering where at some position the value of the potential is at least Δ^2, a so-called *high-potential realizable degree ordering*, then there is also a realizable degree ordering σ that is of the following "pattern" (see Figure 4.4 for an illustration): The ordering σ can be partitioned into four subsequences I, G, B, E (here I stands for initialization, G for good types, B for bad types, and E for end). It starts with sequence I "establishing" a potential of value at least Δ^2, a so-called *high potential*. Correspondingly, at the end there is a sequence E that reduces the value of the potential from a value that is at least Δ^2 to zero. Furthermore, I and E are of length at most $\Delta^{2\Delta}$. The subsequence G, which is of arbitrary length, only consists of good type tuples in arbitrary order and, correspondingly, B is of arbitrary length but only consists of bad type tuples in arbitrary order. Recall that a tuple $\binom{a}{b}$ is a good type tuple if $a \leq b$, otherwise it is a bad type tuple.

This characterization allows us to check whether there is a high-potential realizable degree ordering as follows: First, branch into all possibilities to

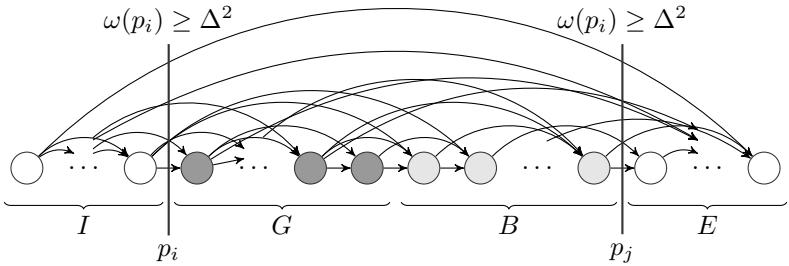


Figure 4.4.: A schematic illustration of a realizing high-potential DAG that corresponds to the pattern $I\,G\,B\,E$. Thereby, I is a subsequence of length at most $\Delta^{2\Delta}$ such that the first high potential occurs at position i. Correspondingly, j is the last position with high potential and E is a sequence of length at most $\Delta^{2\Delta}$. The sequence G (respectively B) consists of only good (bad) type vertices but is of arbitrary length. All high-potential realizations can be reordered to fit into this pattern.

choose I and E. Second, insert in each branch the remaining vertices sorted by good and bad types between I and E and, third, check whether this ordering is a realizable degree ordering. There are at most $((\Delta+1)^2)^{2\Delta^{2\Delta}}) = \Delta^{\Delta^{\mathcal{O}(\Delta)}}$ possibilities for choosing I and E. Furthermore, the insertion and checking can be done in polynomial time, see Lemma 4.10. Hence, this branching algorithm yields fixed-parameter tractability with respect to Δ for the high-potential case.

Our strategy to prove that there is indeed a high-potential realizable degree ordering with the pattern $I\,G\,B\,E$ is as follows. Let $\sigma[1,n]$ be an arbitrary high-potential realizable degree ordering and let $1 \leq i \leq n$ be the first position with a high potential and, symmetrically, let j be the last position with a high potential. In the first part of this section (see Proposition 4.20), we show that σ can be restructured such that $i \leq \Delta^{2\Delta}$ and $j \geq n - \Delta^{2\Delta}$. To prove this the main argument is that if $i > \Delta^{2\Delta}$, then, since there are $\mathcal{O}(\Delta^{2\Delta})$ different potentials with value less than Δ^2, there have to be two positions $1 \leq \ell_1 < \ell_2 < i$ with $p_{\ell_1} = p_{\ell_2}$. Then, by Lemma 4.14, we can cut out $\sigma[\ell_1+1,\ell_2]$ from σ and we will show (see Lemma 4.19) that we can reinsert it right behind i, resulting in a realizable degree ordering $\sigma[1,\ell_1]\sigma[\ell_2+1,i]\sigma[\ell_1+1,\ell_2]\sigma[i+1,n]$. By iteratively applying this operation, we end up with a realizable degree ordering where the

first position with high potential is at most $\Delta^{2\Delta}$. A symmetric argument holds for the last position j with high potential.

In the second part of this section, we show that we can arbitrarily sort the vertices in $\sigma[i+1,j]$ under the constraint that at first vertices of good type (indegree at most outdegree) occur in any order, and then they are followed by the bad type (indegree larger than outdegree) vertices (see Proposition 4.21). The basic idea herein is that if the value of a potential at some position ℓ is at least Δ^2, then there are at least Δ vertices with remaining outdegree at least one at position ℓ. Hence, one can always connect the next vertex to the preceding vertices. Here, the sorting such that in $\sigma[i+1,j]$ first the good type vertices occur ensures that at each position $\ell \in \{i+1, i+2, \ldots, j-1\}$ the value of the potential is at least Δ^2.

Bounding the length of I and E. With the next lemmas and observations we show that the subsequences I and E of the above pattern can be assumed to be of length at most $\Delta^{2\Delta}$. As already mentioned above, if $i > \Delta^{2\Delta}$, then there have to be two positions $1 \leq \ell_1 < \ell_2 < i$ such that $p^\sigma_{\ell_1} = p^\sigma_{\ell_2}$. Hence, by Lemma 4.14, $\sigma' = \sigma[1,\ell_1]\sigma[\ell_2+1,n]$ is a realizable degree ordering with $p^{\sigma'}_{\ell_1+\ell} \succeq p^\sigma_{\ell_2+\ell}$ for all $1 \leq \ell \leq n-\ell_2$. Next, we show that $\sigma[\ell_1+1,\ell_2]$ can be reinserted behind $\sigma[i,i]$, meaning that $\sigma[1,\ell_1]\sigma[\ell_2+1,i]\sigma[\ell_1+1,\ell_2]\sigma[i+1,n]$ is a realizable degree ordering with a high potential at position $i-(\ell_2-\ell_1+1)$. However, observe that to prove this, Lemma 4.15 cannot be used since $\omega(p^{\sigma'}_{i-(\ell_2-\ell_1+1)}) \geq \Delta^2 > \omega(p^\sigma_{\ell_1})$. Thus, in the following we prove that we can reinsert the cut out subsequence in the high-potential part (Lemma 4.19). Before that, we formalize the observation that among potentials with the same value there is one that is minimum concerning the ordering introduced in Definition 4.6.

Lemma 4.17. *For a fixed positive integer x let $p(x) \in \mathbb{N}^\Delta$ be the potential with*

$$p(x)[j] = \begin{cases} \left\lceil \frac{x}{\Delta} \right\rceil, & \text{if } j \leq x \text{ modulo } \Delta \\ \left\lfloor \frac{x}{\Delta} \right\rfloor, & \text{otherwise} \end{cases}$$

for all $1 \leq j \leq \Delta$. Then, for all potentials $p' \in \mathbb{N}^\Delta$ with $x = \omega(p') = \omega(p(x))$ it holds that $p' \succeq p(x)$.

Proof. Let $p(x) \in \mathbb{N}^\Delta$ be the potential as defined in Lemma 4.17 and let $p' \in \mathbb{N}^\Delta$ be a potential with $\omega(p') = \omega(p(x))$. Clearly, by definition it holds that $\omega(p(x)) = x$. Towards a contradiction assume that $p' \succeq p(x)$ does *not* hold.

Then, there is a position $1 \leq j \leq \Delta$ with $\sum_{\ell=1}^{j} p(x)[\ell] > \sum_{\ell=1}^{j} p'[\ell]$. From this it follows that there is a position $1 \leq t \leq j$ such that $p(x)[t] > p'[t]$ and since $p(x)[t] \leq \lceil x/\Delta \rceil$ it follows that $p'[t] \leq \lfloor x/\Delta \rfloor$. Recall that for any potential p it holds that $p[\ell_1] \geq p[\ell_2]$ for all $1 \leq \ell_1 \leq \ell_2 \leq \Delta$ (see remark after Definition 4.2). Thus, from $p'[t] \leq \lfloor x/\Delta \rfloor$ it follows that

$$\sum_{\ell=j+1}^{\Delta} p'[\ell] \leq (\Delta - j)\lfloor x/\Delta \rfloor \leq \sum_{\ell=j+1}^{\Delta} p(x)[\ell].$$

Together with $\sum_{\ell=1}^{j} p(x)[\ell] > \sum_{\ell=1}^{j} p'[\ell]$ this yields a contradiction to $\omega(p(x)) = \omega(p')$. □

Lemma 4.18. *Let $\sigma_1[1, n_1]$ be a partial realizable degree ordering with input potential 0^Δ and output potential $p_{n_1}^{\sigma_1}$. Furthermore, let σ_2 be a partial realizable degree ordering with input potential $p_0^{\sigma_2}$ and output potential 0^Δ such that $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2}) \geq \Delta^2$. Then, $\sigma = \sigma_1\sigma_2$ is a realizable degree ordering.*

Proof. Let $\sigma_1[1, n_1]$ be a partial realizable degree ordering with input potential 0^Δ and output potential $p_{n_1}^{\sigma_1}$. Furthermore, let $\sigma_2[1, n_2]$ be a partial realizable degree ordering with input potential $p_0^{\sigma_2}$ and output potential 0^Δ such that $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2}) \geq \Delta^2$. For $x = \omega(p_0^{\sigma_2})$, we prove that $\sigma = \sigma_1\sigma_2$ is a realizable degree ordering in case of $p_{n_1}^{\sigma_1} = p(x)$, that is,

$$p_{n_1}^{\sigma_1}[j] = \begin{cases} \left\lceil \frac{x}{\Delta} \right\rceil, & \text{if } j \leq x \text{ modulo } \Delta \\ \left\lfloor \frac{x}{\Delta} \right\rfloor, & \text{otherwise} \end{cases}$$

for all $1 \leq j \leq \Delta$. This proves that σ_2 is a partial realizable degree ordering with input potential $p_{n_1}^{\sigma_1}$ and output potential 0^Δ and thus Lemma 4.17 and Proposition 4.12 imply the correctness of Lemma 4.18 in the general case.

Consider now the special case with $p_{n_1}^{\sigma_1}$ defined as above. Denote with D_1 and D_2 the two realizing DAGs corresponding to two realizable degree orderings containing σ_1 respectively σ_2 as subsequences. In the following, we describe how to construct a DAG with a topological ordering that corresponds to σ: First, copy all arcs between two vertices that correspond to two tuples in σ_1 resp. σ_2 from D_1 resp. D_2 into the DAG for σ. It remains to specify the arcs that start in a vertex that corresponds to some tuple in σ_1 and ends in a vertex that corresponds to some tuple in σ_2. For a more convenient construction of these arcs, assume that there are no arcs between two vertices that correspond to

tuples in σ_1 or σ_2, respectively. Because in the following we only add arcs having one endpoint in tuples corresponding to σ_1 and the other endpoint in vertices that correspond to tuples in σ_2, these arcs can be removed and the integers in the tuples of σ_1 and σ_2 can be decreased correspondingly. Afterwards, the arcs can be reinserted. Hence, we assume in the following that σ_1 consists of sources and σ_2 consists of sinks. We remark that the potential $p_{n_1}^{\sigma_1}$ is not changed by these simplifications.

We now inductively prove for each $r \in \{0, 1, \ldots, n_2\}$ that $\sigma[1, n_1 + r] = \sigma_1[1, n_1]\sigma_2[1, r]$ (with $\sigma[1, n_1] = \sigma_1[1, n_1]$) is a partial realizable degree ordering. Observe that by the choice of $p_{n_1}^{\sigma_1}$ it holds that $p_{n_1}^{\sigma_1}[1] \geq \Delta$, that is, there are enough vertices that have outgoing arcs "left" to connect the next vertex. The key point will be to show that when adding the rth vertex and connecting it to the other vertices, we will decrease the remaining outdegree of any vertex below ℓ for any $1 \leq \ell < \Delta$ only if there is no vertex left with remaining outdegree $\ell + 1$. Intuitively, this means that the potential $p_{n_1}^{\sigma_1}$ is used "level-wise" (from high to small degree) in the process of adding vertices and therefore there are always enough vertices with remaining outdegree at least one for the next vertex to connect. Putting this into a formula, we show that for the output potential $p_{n_1+r}^{\sigma[1,n_1+r]}$ of $\sigma[1, n_1 + r]$ it holds that

$$\forall \ell \in \{1, 2, \ldots, \Delta - 1\} \colon p_{n_1+r}^{\sigma[1,n_1+r]}[\ell] < p_{n_1}^{\sigma_1}[\ell] \Rightarrow p_{n_1+r}^{\sigma[1,n_1+r]}[\ell + 1] = 0. \tag{4.5}$$

Note that this perfectly reflects the definition of well-connectedness. For the base case of the induction observe that $\sigma[1, n_1] = \sigma[1, n]$ is a partial realizable degree ordering and with an output potential satisfying the required property as $p_{n_1}^{\sigma[1,n_1]} = p_{n_1}^{\sigma_1}$. For the induction step assume that $\sigma[1, n_1 + r]$ is a partial realizable degree ordering with an output potential satisfying Property (4.5). Let $\binom{a}{b} = \sigma_2[r + 1, r + 1]$ be the tuple we want to add next. Observe that, by our choice of $p_{n_1}^{\sigma_1}$, it holds for any $1 \leq \ell \leq \Delta$ that $p_{n_1}^{\sigma_1}[\ell] \geq \Delta \geq a$. In particular, we have $p_{n_1}^{\sigma_1}[1] \geq \Delta \geq a$. Now, consider the two cases

(i) $p_{n_1+r}^{\sigma[1,n_1+r]}[2] = 0$ where the remaining outdegree of all vertices at position $n_1 + r$ is at most one and

(ii) $p_{n_1+r}^{\sigma[1,n_1+r]}[2] > 0$.

For case (i) observe that $\omega(p_{n_1}^{\sigma_1}) = \omega(p_0^{\sigma_2})$ is a precondition. As σ_2 is a partial realizable degree ordering with output potential 0^{Δ} it follows that $p_{n_1+r}^{\sigma[1,n_1+r]}[1] \geq a$.

Hence, by Observation 4.11, $\sigma[1, n_1 + r + 1]$ is indeed a partial realizable degree ordering where $p_{n_1+r+1}^{\sigma[1,n_1+r+1]}$ clearly satisfies Property (4.5) as $p_{n_1+r+1}^{\sigma[1,n_1+r+1]}[2] = 0$. Now, consider case (ii). As $p_{n_1+r}^{\sigma[1,n_1+r]}[2] > 0$ and $p_{n_1+r}^{\sigma[1,n_1+r]}$ satisfies Property (4.5), it follows that $p_{n_1+r}^{\sigma[1,n_1+r]}[1] = p_n^{\sigma}[1] \geq a$. Hence, by Observation 4.11, $\sigma[1, n_1+r+1]$ is indeed a partial realizable degree ordering. As we can assume by Lemma 4.9 that the corresponding realizing DAG is well-connected, the added vertex corresponding to $\sigma_2[r+1, r+1]$ has the vertices with the highest remaining outdegree at position $n_1 + r$ as inneighbors. Let d be the maximum occurring outdegree of any vertex at position $n_1 + r$, that is, $d = \Delta$ or $p_{n_1+r}^{\sigma[1,n_1+r]}[d+1] = 0$. As $p_{n_1+r}^{\sigma[1,n_1+r]}$ satisfies Property (4.5), it follows that for all $1 \leq \ell < d$ we have $p_{n_1+r}^{\sigma[1,n_1+r]}[\ell] = p_{n_1}^{\sigma_1}[\ell] \geq \Delta \geq a$. Hence, there are at least a different vertices with remaining outdegree between $d - 1$ and d at position $n_1 + r$. Furthermore, the added vertex is connected to a vertex with remaining outdegree $d - 1$ at position $n_1 + r$ if and only if there are less than a vertices with remaining outdegree d at position $n_1 + r$. Thus, $p_{n_1+r+1}^{\sigma[1,n_1+r+1]}[d - 1] < p_{n_1}^{\sigma_1}[d - 1]$ implies that $p_{n_1+r+1}^{\sigma[1,n_1+r+1]}[d] = 0$ and, hence, $p_{n_1+r+1}^{\sigma[1,n_1+r+1]}$ satisfies Property (4.5). □

While Lemma 4.14 shows that we can cut out a partial realizable degree ordering with equal input and output potential, the following lemma shows that we can reinsert it right behind a high potential in any realizable degree ordering.

Lemma 4.19. *Let $\sigma_1[1, n_1]$ be a realizable degree ordering and let $\sigma_2[1, n_2]$ be a partial realizable degree ordering with input and output potential p. Then, for any position $1 \leq i \leq n_1$ with $\omega(p_i^{\sigma_1}) \geq \max\{\Delta^2, \omega(p)\}$ it holds that $\sigma = \sigma_1[1, i]\sigma_2[1, n_2]\sigma[i + 1, n_1]$ is a realizable degree ordering.*

Proof. Let $\sigma_1[1, n_1]$ be a realizable degree ordering and let $\sigma_2[1, n_2]$ be a partial realizable degree ordering with input and output potential p. Furthermore, let $1 \leq i \leq n_1$ be a position with $\omega(p_i^{\sigma_1}) \geq \max\{\Delta^2, \omega(p)\}$. We prove that $\sigma = \sigma_1[1, i]\sigma_2[1, n_2]\sigma_1[i + 1, n_1]$ is a realizable degree ordering. To this end, we show that $\sigma_1[1, i]\sigma_2[1, n_2]$ is a partial realizable degree ordering with input potential 0^{Δ} and output potential p' where $\omega(p') = \omega(p_i^{\sigma_1})$. Then, from Lemma 4.18 it follows that σ is a realizable degree ordering.

By Definition 4.5 there exists a realizable degree ordering $\widetilde{\sigma}_2[1, \widetilde{n}_2]$ such that $\sigma_2 = \widetilde{\sigma}_2[i, j]$ for some $1 \leq i < j \leq \widetilde{n}_2$ and $p = p_{i-1}^{\widetilde{\sigma}_2} = p_j^{\widetilde{\sigma}_2}$. Now, if $\omega(p_i^{\sigma_1}) > \omega(p)$, then we add $\omega(p_i^{\sigma_1}) - \omega(p)$ tuples of type $\binom{1}{0}$ at the beginning of $\widetilde{\sigma}_2$ and the same number of tuples of type $\binom{0}{1}$ at the end of $\widetilde{\sigma}_2$. This shows

that σ_2 is a partial realizable degree ordering with input potential p' and output potential p' with $\omega(p') = \omega(p_i^{\sigma_1}) \geq \Delta^2$.

From this together with Lemma 4.18 it follows that $\sigma_1[1,i]\sigma_2[1,n_2]\widetilde{\sigma}_2[j+1,\widetilde{n}_2]$ is a realizable degree ordering. Since, by our assumption σ_2 is a partial realizable degree ordering with input and output potential p', it follows that $\sum_{v\in\sigma_2} \deg^-(v) = \sum_{v\in\sigma_2} \deg^+(v)$. Hence, from Observation 4.8 and the fact that the potential at position i in $\sigma_1[1,i]\sigma_2[1,n_2]\widetilde{\sigma}_2[j+1,\widetilde{n}_2]$ is $p_i^{\sigma_1}$, it follows that $\sigma_1[1,i]\sigma_2[1,n_2]$ is a partial realizable degree ordering with input potential 0^Δ and output potential p'' with $\omega(p'') = \omega(p_i^{\sigma_1}) \geq \Delta^2$. Thus, by Lemma 4.18 it follows that σ is a realizable degree ordering. □

With Lemma 4.19 we are able to bound the length of the parts I and E, that is, the first position of a high potential is "near" the start and the last position of a high potential is "near" the end.

Proposition 4.20. *If a* DAG Realization *instance consisting of n tuples admits a high-potential realization, then there is also a corresponding high-potential realizable degree ordering such that the first position with high potential is at most $\Delta^{2\Delta}$ and the last position with high potential is at least $n - \Delta^{2\Delta}$.*

Proof. Let $\sigma[1,n]$ be a high-potential realizable degree ordering and let $1 \leq i \leq n$ be the first position where $\omega(p_i) \geq \Delta^2$. Consider the case where $i > \Delta^{2\Delta}$. Thus, for all $1 \leq \ell < i$ it holds that $\omega(p_\ell) < \Delta^2$. There are $\Delta^{2\Delta}$ integer Δ-tuples with elements between 0 and $\Delta^2 - 1$ and not every Δ-tuple is a potential. Hence, there are less than $\Delta^{2\Delta}$ potentials with value less than Δ^2. Thus, there are two indices $1 \leq \ell_1 < \ell_2 < i$ with $p_{\ell_1} = p_{\ell_2}$. By Lemma 4.14, the ordering $\sigma[1,\ell_1]\sigma[\ell_2+1,n]$ is a realizable degree ordering where the potential at position $i-(\ell_2-\ell_1)$ is p_i. Moreover, by definition $\sigma[\ell_1+1,\ell_2]$ is a partial realizable degree ordering with input and output potential p_{ℓ_1} where $\omega(p_{\ell_1}) < \omega(p_i)$. Thus, by Lemma 4.19, it holds that $\sigma[1,\ell_1]\sigma[\ell_2+1,i]\sigma[\ell_1+1,\ell_2]\sigma[i+1,n]$ is a realizable degree ordering. Moreover, since $\sum_{v\in\sigma\{\ell_1+1,\ell_2\}} \deg^-(v) - \deg^+(v) = 0$ it follows from Observation 4.8 that in this realizable degree ordering the first position with high potential is $i-(\ell_2-\ell_1)$. Applying the same operation iteratively as long as there are two positions with equal potential before the first high potential results in a realizable degree ordering where the first position with high potential is at most $\Delta^{2\Delta}$.

Basically, the same argumentation can be applied for the last position j where a high potential occurs. If $j < n - \Delta^{2\Delta}$, then there have to be two indices $j < \ell_1 < \ell_2 \leq n$ where $p_{\ell_1} = p_{\ell_2}$. Then, by Lemma 4.14 the ordering

$\sigma[1,\ell_1]\sigma[\ell_2+1,n]$ is a realizable degree ordering and $\sigma[\ell_1+1,\ell_2]$ is a partial realizable degree ordering with input and output potential p_{ℓ_1} with $\omega(\ell_1) < \omega(p_j)$. Thus, by Lemma 4.19 the ordering $\sigma[1,j]\sigma[\ell_1+1,\ell_2]\sigma[j+1,\ell_1]\sigma[\ell_2+1,n]$ is a realizable degree ordering. Since $\sum_{v\in\sigma\{\ell_1+1,\ell_2\}} \deg^-(v) - \deg^+(v) = 0$ it follows from Observation 4.8 that the last position with high potential is $j+(\ell_2-\ell_1)$. Again, by applying this operation iteratively we get an ordering where the last position with high potential is at least $n-\Delta^{2\Delta}$. □

Sorting the remaining vertices. Having shown that we can assume that for the first position i and the last position j with high potential it holds that $i \leq \Delta^{2\Delta}$ and $j \geq n-\Delta^{2\Delta}$, we next prove that one can sort all vertices between i and j arbitrarily by good (indegree at most outdegree) and bad (indegree larger than outdegree) types.

Proposition 4.21. *Let $\sigma[1,n]$ be a high-potential realizable degree ordering and let $1 \leq i < j \leq n$ be two arbitrary positions such that $\omega(p_i) \geq \Delta^2$ and $\omega(p_j) \geq \Delta^2$. Furthermore, let $\sigma'[i+1,j]$ be a permutation of the tuples in $\sigma[i+1,j]$ such that there is a position $0 \leq \ell \leq j-i$ with the property that the first ℓ tuples in $\sigma'[i+1,j]$ are of good type and all subsequent tuples are of bad type. Then, the ordering $\sigma[1,i]\sigma'[i+1,j]\sigma[j+1,n]$ is a realizable degree ordering.*

Proof. Assume that there is a high-potential realizable degree ordering with two indices $1 \leq i \leq j \leq n$ such that $\omega(p_i) \geq \Delta^2$ and $\omega(p_j) \geq \Delta^2$. We prove that $\sigma[1,i]\sigma'[i+1,j]\sigma[j+1,n]$ is a realizable degree ordering for any reordering $\sigma'[i+1,j]$ of $\sigma[i+1,j]$ where the first ℓ tuples are of good types and the remaining ones of bad types.

To this end, by induction on h with $1 \leq h \leq j-i$ we show that the sequence $\sigma[1,i]\sigma'[i+1,i+h]$ is a partial realizable degree ordering with input potential 0^Δ and output potential p_h with $\omega(p_h) \geq \Delta^2$. First, by induction hypothesis the output potential p_{h-1} of $\sigma[1,i]\sigma'[i+1,i+h-1]$ is a high potential and hence $p_{h-1}[1] \geq \Delta$. Thus, by Observation 4.11 $\sigma[1,i]\sigma'[i+1,i+h-1]$ is a partial realizable degree ordering. It remains to show that the value of the output potential p_h of $\sigma[1,i]\sigma'[i+1,i+h]$ is at least Δ^2. Towards a contradiction suppose that it is not. This implies

$$\sum_{\binom{a}{b}\in\sigma'\{i+1,i+h\}} a-b > \omega(p_i) - \Delta^2. \tag{4.6}$$

Clearly, $\sigma'[i+h, i+h]$ has to be a bad type tuple, otherwise Inequality (4.6) cannot be true. However, it holds that

$$\omega(p_i) - \sum_{\binom{a}{b} \in \sigma\{i+1,j\}} a - b = \omega(p_j) \geq \Delta^2$$

and thus

$$\sum_{\binom{a}{b} \in \sigma\{i+1,j\}} a - b \leq \omega(p_i) - \Delta^2. \tag{4.7}$$

Since $\sigma'[i+1, j]$ is sorted by good and bad types and $\sigma'[i+h, i+h]$ is of bad type, all tuples in $\sigma'[i+h, j]$ are bad type tuples with $a-b < 0$. Thus, Inequality (4.7) yields a contradiction to Inequality (4.6). Hence, $\sigma[1, i]\sigma'[i+1, j]$ is a partial realizable degree ordering with input potential 0^Δ and output potential p_{j-i} with $p_{j-i} \geq \Delta^2$. Furthermore, by Observation 4.8, it holds that $\omega(p_{j-i}) = \omega(p_j^\sigma)$. Thus, by Lemma 4.18, $\sigma[1, i]\sigma'[i+1, j]\sigma[j+1, n]$ is indeed a realizable degree ordering. □

Propositions 4.20 and 4.21 lead to the central result of this section:

Theorem 4.22. *If a* DAG Realization *instance admits a high-potential realizable degree ordering, then it can be solved in* $\Delta^{\Delta^{\mathcal{O}(\Delta)}} \cdot n + \mathcal{O}(n \log n)$ *time.*

Proof. If an instance of DAG Realization admits a high-potential realizable degree ordering, then by Proposition 4.20 there is also a high-potential realizable degree ordering in which the occurrence of the first high potential is at most at position i with $i \leq \Delta^{2\Delta}$ and the last occurrence of a high potential is at least at position j with $j \geq n - \Delta^{2\Delta}$. Recall that there are at most $(\Delta+1)^2$ types of tuples in the given degree sequence, and thus the subsequences $\sigma[1, i]$ and $\sigma[j, n]$ of a realizable degree ordering $\sigma[1, n]$ can be found by exhaustive search in $\Delta^{\Delta^{\mathcal{O}(\Delta)}}$ time. Proposition 4.21 shows that the remaining tuples can be arbitrarily inserted between them, as long as they are sorted by good and bad types. This can be done in $\mathcal{O}(n)$ time. Finally, we check whether the produced ordering is indeed a realizable degree ordering, which can be done by Lemma 4.10 in $\mathcal{O}(\Delta^2 n)$ time. Since we have to read the input of size $\mathcal{O}(n \log n)$, we altogether arrive at a running time of $\Delta^{\Delta^{\mathcal{O}(\Delta)}} \cdot n + \mathcal{O}(n \log n)$. □

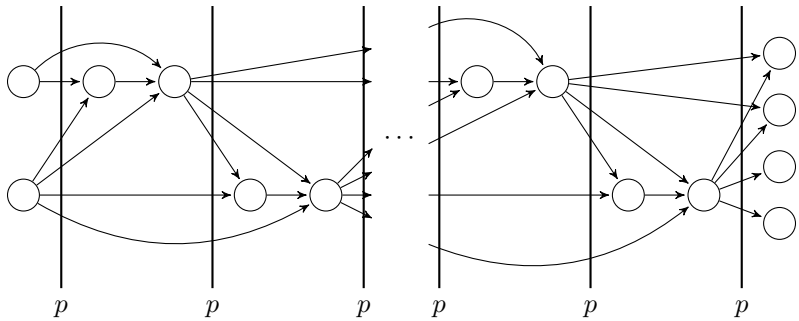


Figure 4.5.: Realization for the degree sequence $\binom{0}{2}, \binom{0}{4}, \binom{2}{1}, \binom{3}{4}, \binom{2}{1}, \binom{3}{4}, \ldots, \binom{2}{1}, \binom{3}{4}, \binom{2}{1}, \binom{3}{4}, \binom{2}{0}, \binom{2}{0}, \binom{1}{0}, \binom{1}{0}$. Since this sequence basically consists of only two different types (not regarding types with indegree or outdegree equal to zero), it is easy to check that the pictured low-potential realization (the highest occurring value of a potential is $\omega(p) = 6$) is the only one. The displayed potentials are all identical and they furthermore indicate the repetitions of the *super-type* $\binom{2}{1}, \binom{3}{4}$.

4.3.3. Low-Potential Sequences

In this section, we will provide an algorithm that finds a *low-potential realization* (if one exists) for a DAG Realization instance, that is, a realization such that the value of all potentials is strictly less than Δ^2. See Figure 4.5 for an example of such a realization.

As in the high-potential case, the main idea is to restrict the length of the parts in a realizable degree ordering that have to be guessed by brute force. In the low-potential case, we can exploit that there are at most $\Delta^{2\Delta}$ potentials with value less than Δ^2 and, thus, if the length of a realizable degree ordering is greater than $\Delta^{2\Delta}$, then there have to be two positions with equal potential. We call partial realizable degree orderings with equal potential at their start and end *super-types*. Then, by cutting out super-types and reinserting them appropriately, we can upper-bound the distance between two subsequent positions with the same potential, implying a restricted number of different super-types. Hence, if one removes in such a realizable degree ordering (almost) all repetitions of a super-type, then the length of the resulting *shortened realizable degree ordering* can be upper-bounded by a function solely depending on Δ.

Our algorithm works as follows: In the first step it branches into all possibilities to choose a shortened realizable degree ordering. In the second step, it checks via solving an ILP (integer linear program) whether this shortened realizable degree ordering can be extended by (re-)inserting repetitions of super-types to a realizable degree ordering of the input degree sequence.

In the following we introduce some notations to formalize the above concepts.

Definition 4.7. A *super-type* of potential $p \in \mathbb{N}^{\Delta}$ is a partial realizable degree ordering $\sigma[1, n]$ with input and output potential p where all potentials from position 1 to $n-1$ are different from p. A *k-repetition* of a super-type s in a realizable degree ordering σ is a subsequence ψ of σ with $\psi = s^k$, that is, k subsequent occurrences of s in σ. If k is maximal under this condition, then ψ is called a *maximal repetition* in σ.

In Figure 4.5 an example for a realizable degree sequence is given where the only existing realizable degree ordering consists basically of one maximal repetition of the super-type $\binom{2}{1}, \binom{3}{4}$ of potential $(2, 2, 1, 1)$. Removing this repetition indeed gives a very short ordering: $\binom{0}{2}, \binom{0}{4}, \binom{2}{1}, \binom{3}{4}, \binom{2}{0}, \binom{2}{0}, \binom{1}{0}, \binom{1}{0}$. We now formalize these shortened orderings.

Definition 4.8. A *repetition removal* in a realizable degree ordering is the replacement of one maximal repetition of a super-type s with $|s| \leq \Delta^{2\Delta}$ by s. An ordering σ' is a *shortened realizable degree ordering* of the realizable degree ordering σ if σ' can be obtained from σ by a series of repetition removals.

Observe that a shortened realizable degree ordering may still contain repetitions of super-types although removing these would yield an even shorter shortened realizable degree ordering. The reason for this is simplicity: The sole purpose of removing repetitions is to upper-bound the length of the shortened realizable degree ordering. Having achieved this upper bound, we do not care about further repetitions as the algorithm branching in all possibilities for the bounded-length shortened realizable degree ordering is not affected by possibly existing repetitions. To check in a branch whether a given sequence of integer tuples is indeed a shortened realizable degree ordering we use the following observation following from the definition of repetition removals and Lemma 4.14.

Observation 4.23. *Let σ be a realizable degree ordering and let σ' be a shortened realizable degree ordering of σ. Then σ' is also a realizable degree ordering.*

Using Observation 4.23 our algorithm branches into all sequences of integer tuples within the length-bound that we will provide. Then, we check in each

branch whether the sequence is realizable; by Lemma 4.10, this can be done in $\mathcal{O}(n\Delta)$ time. If yes, then we try to extend the sequence by inserting repetitions of super-types. Here, by Lemma 4.15 each insertion yields again a realizable degree ordering. Thus, the remaining problem is to "just" find a combination of super-types such that after inserting them the resulting sequence contains the very same tuples as our input degree sequence. We solve this problem with an integer linear program (ILP). Here, the restriction in Definition 4.8 that only repetitions of "short" super-types can be removed is crucial: Due to this bounded length also the total number of different super-types which have to be considered as candidates for (re)insertion is upper-bounded by a function of Δ. For each possible candidate super-type there is a variable representing the number k of repetitions of this super-type that have to be inserted to the shortened realizable degree ordering. The ILP will then return a solution if and only if we can insert the corresponding super-types and obtain a realizable degree ordering for our input instance.

We now prove the claim that there exists a shortened realizable degree ordering of length bounded by a function of Δ. To this end, our two core tools are the repetition removals and the reordering operation provided in Proposition 4.16.

Lemma 4.24. *If a* DAG Realization *instance $\mathcal{S}$ admits a low-potential realization, then there is also a corresponding low-potential realizable degree ordering σ for $\mathcal{S}$ and a shortened realizable degree ordering σ' of σ such that the length of σ' is bounded by $\Delta^{2\Delta}((\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta})$.*

Proof. Let $\sigma[1,n]$ be a low-potential realizable degree ordering for $\mathcal{S}$ and let $\mathcal{P}^\sigma = p_0^\sigma, p_1^\sigma, \ldots, p_n^\sigma$ be the corresponding sequence of potentials with values strictly less than Δ^2 with $p_0^\sigma = p_n^\sigma = 0^\Delta$. For a potential $p \in \mathcal{P}^\sigma$ we denote by $\text{first}_{\mathcal{P}}^\sigma(p)$ the position of the first occurrence of p in $\mathcal{P}^\sigma$.

Our strategy to construct a shortened realizable degree ordering σ' starting from σ is as follows: Iterate over the potentials occurring in $\mathcal{P}$ sorted by the values of $\text{first}_{\mathcal{P}}^\sigma$ in descending order, that is, the first considered potential p maximizes $\text{first}_{\mathcal{P}}^\sigma(p)$. In the ith iteration denote the considered potential by $p(i)$ and construct a shortened realizable degree ordering $\sigma(i)$ from $\sigma(i-1)$ where $\sigma(i)$ can be split into two parts $\sigma(i) = \sigma_1(i)\sigma_2(i)$ such that

1. all potentials considered until the ith iteration do not occur in the first part, that is, $p(1), p(2), \ldots, p(i) \notin \mathcal{P}^{\sigma_1(i)}$,

2. in iteration i only repetition removals that correspond to super-types of potential $p(i)$ are performed, and

3. the length of the second part is upper-bounded by $i((\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta})$.

For the initialization we set $\sigma(0) := \sigma_1(0) := \sigma$ and in the end the shortened realizable degree ordering σ' we are looking for will be $\sigma(\ell)$, where ℓ is the number of iterations. Since the value of any occurring potential is, by assumption, lower than Δ^2, there are less than $\Delta^{2\Delta}$ possible potentials: There are $\Delta^{2\Delta}$ integer Δ-tuples with elements between 0 and Δ^2-1 and not every Δ-tuple is a potential. Hence, the described algorithm terminates after at most $\Delta^{2\Delta}$ iterations. By Property 1 it follows that $\sigma(\ell)=\sigma_2(\ell)$ and by Property 3 the claimed length of $\Delta^{2\Delta}((\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta})$ follows.

We now show how to obtain $\sigma(i)$ from $\sigma(i-1)$ in the ith iteration. To this end we initialize $\sigma(i) := \sigma(i-1)$ and reorder it in the following. Let $j_1, j_2, \ldots, j_r$ be the occurrences of $p(i)$ in $\mathcal{P}^{\sigma_1(i-1)}$, that is, the occurrences in the first part of $\sigma(i-1)$. Assume that there is an $x \in \{2,3,\ldots,r\}$ such that $i_x - i_{x-1} > \Delta^{2\Delta}$. Then, there are two positions $j_{x-1} < h_1 < h_2 < j_x$ such that $p_{h_1}^{\sigma_1(i-1)} = p_{h_2}^{\sigma_1(i-1)} =: q$. As we consider the first part of $\sigma(i-1)$ it holds that $\text{first}_{\mathcal{P}}^{\sigma(i)}(p) > \text{first}_{\mathcal{P}}^{\sigma(i)}(q)$. Thus, there is a position k with $k < j_1$ with $p_k^{\sigma_1(i-1)} = q$. By Observation 4.23 and Proposition 4.16, the sequence $\sigma(i)[1,k]\sigma(i)[h_1+1,h_2]\sigma(i)[k+1,h_1]\sigma(i)[h_2+1,|\sigma(i)|]$ is also a realizable degree ordering. Since the second part $\sigma_2(i-1)$ remains unchanged by this ordering, the repetition removals performed to obtain $\sigma(i-1)$ can be reversed. Hence, the reordered sequence is also a shortened realizable degree ordering.

Our procedure to construct $\sigma(i)$ from $\sigma(i-1)$ is as follows: First, while there are two consecutive occurrences of the potential $p(i)$ having distance more than $\Delta^{2\Delta}$ in the sequence of potentials we reorder the sequence as described above. Second, while in the part between the last occurrence of $p(i)$ and the end of $\sigma_1(i-1)$ a potential occurs twice, we reorder the sequence in the same way by moving parts at the beginning of the sequence. Since this reordering only decreases the distance of consecutive occurrences of $p(i)$ it terminates at some point. Third, using again the reordering operation of Proposition 4.16, we sort all super-types of potential $p(i)$ such that there is at most one subsequent occurrence of each super-type of potential $p(i)$. Finally, we perform an exhaustive repetition removal for all super-types of potential $p(i)$. Then $\sigma(i) = \sigma_1(i)\sigma_2(i)$ is the resulting shortened realizable degree ordering where $\sigma_2(i)$ starts at $\text{first}_{\mathcal{P}}^{\sigma(i)}(p(i))$.

To verify the correctness observe that each repetition removal applies to a super-type of length at most $\Delta^{2\Delta}$ since the distance between two consecutive occurrences of $p(i)$ is at most $\Delta^{2\Delta}$. Thus, Properties 1 and 2 are satisfied. It remains to show that the length of $\sigma_2(i)$ is indeed bounded as required. To this end, observe that $\sigma_2(i)$ consists of super-types of potential $p(i)$, the part between the last occurrence of $p(i)$ and the beginning of $\sigma_2(i-1)$, and $\sigma_2(i-1)$. There are at most $(\Delta+1)^{2\Delta^{2\Delta}}$ different length $\Delta^{2\Delta}$ sequences of integer tuples with entries between zero and Δ. This is also an upper bound in the different super-types of potential $p(i)$. Due to the sorting and repetition removals, each of these super-types occurs at most once. Hence, the first part of $\sigma_2(i)$ stretching until the start of $\sigma_2(i-1)$ has length at most $(\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta}$. Due to Property 3, $\sigma_2(i-1)$ has length $(i-1)((\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta})$ which gives a total length of $i(\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+\Delta^{2\Delta}$ for $\sigma_2(i)$. □

Using Lemma 4.24, the algorithm branches into all possibilities to choose a shortened realizable degree ordering of length at most $\Delta^{2\Delta}((\Delta+1)^{2\Delta^{2\Delta}}\Delta^{2\Delta}+2\Delta^{2\Delta})<(2\Delta)^{3\Delta^{2\Delta}}$. This gives at most $(\Delta+1)^{2(2\Delta)^{3\Delta^{2\Delta}}}$ cases. For a shortened realizable degree ordering, the algorithm checks which of the $(\Delta+1)^{2\Delta^{2\Delta}}$ possibly repeating super-types occur in the sequence of the particular case and stores the occurring ones in a set $\mathcal{T}^S$. Then, given a shortened realizable degree ordering σ' for a DAG Realization instance $\mathcal{S}$ and the set $\mathcal{T}^S$ of super-types that may repeat, the problem of computing a realizable degree ordering that respects σ' is fixed-parameter tractable with respect to the number of super-types in $\mathcal{T}^S$. We give an ILP formulation for this problem. To this end, we formalize the problem and call it Sequence Filling.

Sequence Filling

Input: A multiset $\mathcal{S}=\left\{\binom{a_1}{b_1},\binom{a_2}{b_2},\ldots,\binom{a_n}{b_n}\right\}$, a shortened realizable degree ordering σ' and a set of all super-types $\mathcal{T}^S=\{s_1,s_2,\ldots,s_\ell\}$ of σ' that have length at most $\Delta^{2\Delta}$.

Question: Is there a realizable degree ordering σ for $\mathcal{S}$ such that σ' results from replacing for each $s\in\mathcal{T}^S$ all maximal repetitions of s in σ by one occurrence of s?

Next, we show fixed-parameter tractability of Sequence Filling with respect to the parameter $|\mathcal{T}^S|=\ell$. Since the number ℓ of super-types is bounded by a function only depending on Δ, this completes our algorithm for the case that an input of DAG Realization admits a low-potential realization.

Lemma 4.25. Sequence Filling *is fixed-parameter tractable with respect to the parameter* $|\mathcal{T}^S| = \ell$.

Proof. We show the fixed-parameter tractability result by giving an ILP-formulation of the problem with ℓ variables. It has been shown that an ILP with p variables can be solved in $\mathcal{O}(p^{2.5p+o(p)} \cdot L)$ time and space polynomial in L where L is the input size [FT87, Kan87, Len83]. To solve the Sequence Filling instance, we use the following ILP-formulation:

$$\forall 1 \leq i \leq \ell : \qquad f_i \geq 0 \tag{4.8}$$

$$\forall e \in \mathcal{S} : \qquad \sum_{i=1}^{\ell} f_i \cdot o(e, s_i) = o(e, \mathcal{S}) - o(e, \sigma') \tag{4.9}$$

Each of the ℓ integer variables $f_1, f_2, \ldots, f_\ell$ denotes how often the corresponding super-type will be inserted in σ' to obtain σ. The function $o(e, \mathcal{S})$ ($o(e, s_i)$) denotes the number of occurrences of the tuple e in $\mathcal{S}$ (in the super-type s_i). The ILP consists of $\ell + |\mathcal{S}|$ equations that contain together $\mathcal{O}(\ell \cdot |\mathcal{S}|)$ integers, each upper-bounded by $|\mathcal{S}|$. Hence, the ILP can be solved in $\mathcal{O}(\ell^{2.5\ell+o(\ell)} \cdot \ell \cdot |\mathcal{S}| \log |\mathcal{S}|)$ time.

Now we describe how we use the solution of the ILP-formulation to create a realizable degree ordering σ as described in the problem definition of Sequence Filling. For each super-type $s_i \in \mathcal{T}^S$ with $f_i > 0$ insert an f_i-repetition of s_i right after an occurrence of s_i in σ'. By Lemma 4.15, the ordering that results from adding the maximal repetitions results in a realizable degree ordering for $\sigma' \uplus \biguplus_{i, f_i > 0} \biguplus_{j=1}^{f_i} s_i = \mathcal{S}$.

Next, we show that if the Sequence Filling-instance is a yes-instance, then there exists a solution for our ILP-formulation. If there is a realizable degree ordering σ for $\mathcal{S}$ that respects σ', then there exists a set of super-types T_S such that replacing all maximal repetitions of super-types s_i in T_S in σ by one occurrence of the corresponding super-type results in σ'. Clearly, $T_S \subseteq \mathcal{T}^S$. Set $f_i = 0$ for each $s_i \in \mathcal{T}^S \backslash T_S$ and for all $s_i \in T_S$ set $f_i = k - 1$ where σ contains a maximal k-repetition of s_i. Thus, Inequality (4.8) is fulfilled for all $1 \leq i \leq \ell$. Since σ is a realizable degree ordering for $\mathcal{S}$, Equality (4.9) is fulfilled. □

Combining Lemma 4.24 and Lemma 4.25 shows fixed-parameter tractability for the low-potential case. As to the running time observe that there are $\Delta^{\Delta^{\Delta^{\mathcal{O}(\Delta)}}}$ possibilities for the shortened realizable degree ordering. To check whether a

possible shortened realizable degree ordering is realizable requires $\mathcal{O}(\Delta n)$ time (Lemma 4.10), the construction of the Sequence Filling instance $\mathcal{O}(\Delta^{\Delta^{\mathcal{O}(\Delta)}} \cdot n)$ time, and solving the ILP requires $\Delta^{\Delta^{\Delta^{\mathcal{O}(\Delta)}}} \cdot n \log n$ time. Altogether we have the following theorem.

Theorem 4.26. *If a degree sequence admits a low-potential realization, then it can be found in* $\Delta^{\Delta^{\Delta^{\mathcal{O}(\Delta)}}} \cdot n \log n$ *time.*

Theorems 4.22 and 4.26 together lead to the main result of this section.

Theorem 4.27. DAG Realization *is fixed-parameter tractable with respect to the parameter maximum degree* Δ.

Note that Theorem 4.27 is a mere classification result: The corresponding running time is $\Delta^{\Delta^{\Delta^{\mathcal{O}(\Delta)}}} \cdot n \log n$. It is dominated by the low-potential case.

4.4. Fixed-Parameter Tractability with Respect to $\binom{n}{2} - m$ and $m - n + 1$

In the end of Section 4.2 we showed that DAG Realization remains NP-complete even on sparse and on dense instances. More specifically, for every constant $\ell > 1$, DAG Realization remains NP-complete when restricted to instances with $m < \ell n$ (sparse setting) or $m > n(n-1)/(2\ell)$ (dense setting, see Theorem 4.7). In contrast, in this section we prove that if we measure the sparseness or denseness by the number of arcs that a realizing DAG is away from a tree or a tournament instead of using a fraction of m and n, then the problem becomes tractable. In particular, we show that DAG Realization is fixed-parameter tractable with respect to each of the parameters $\binom{n}{2} - m$ and $m - n + 1$. To this end, we first consider the (very) dense setting and then the (very) sparse setting.

Dense setting. Our algorithm for the dense setting relies on two simple observations: First, for each tuple $\binom{a}{b} \in \mathcal{S}$ with degree $a + b = n - 1$ the position in a realizable degree ordering σ of $\mathcal{S}$ is precisely defined: The a inneighbors of the vertex v realizing the tuple are ahead of v in the topological ordering ϕ corresponding to σ and the b outneighbors of v are behind v in ϕ. As $|\phi| = |\sigma| = n$ it follows that v is the $(a+1)^{\text{st}}$ vertex in ϕ, that is, $\binom{a}{b}$ occurs in any realizable

degree ordering at the $(a+1)^{\text{st}}$ position. Second, any realizing DAG can be obtained from a transitive tournament by exactly $k := \binom{n}{2} - m$ arc removals. Thus, all but $2k$ vertices in the realizing DAG have a degree (indegree plus outdegree) of $n-1$. Putting this together, there are at most $2k$ positions "left" in a realizable degree ordering. Hence, a simple search-tree algorithm tries all possibilities to insert the tuples $\binom{a}{b}$ with $a+b < n-1$ in these "free positions" and checks whether the resulting ordering is indeed a realizable degree ordering. As there are $(2k)!$ possibilities to insert the tuples and, by Lemma 4.10, the checking can be done in $\mathcal{O}(\Delta n)$ time, we arrive at the following.

Theorem 4.28. DAG Realization *is fixed-parameter tractable with respect to the parameter* $k := \binom{n}{2} - m$. *The corresponding running time is* $\mathcal{O}((2k)!\Delta n)$.

Sparse setting. The sparse setting requires more effort. Let $k := m - n + 1$. We develop a polynomial-time executable data reduction rule whose exhaustive application to a degree sequence $\mathcal{S}$ results in an instance which is equivalent to $\mathcal{S}$ and whose maximum degree is at most $2k$. Then, the claimed fixed-parameter tractability follows from Theorem 4.27.

We exploit the following observations: For an instance $\mathcal{S}$ with $m \leq n-1$, Berger and Müller-Hannemann [BM12] have shown that DAG Realization is polynomial-time solvable (recall that we assume $\binom{0}{0} \notin \mathcal{S}$). Moreover, they proved that if $m \geq n-1$ and $\mathcal{S}$ is realizable, then there is a realizing DAG for $\mathcal{S}$ that consists of only one connected component. (Recall that by "connectedness" in a directed graph we always refer to the connectivity in the underlying undirected graph.) Thus, if a degree sequence is realizable, then there is a connected realizing DAG D such that the parameter k denotes the size of a feedback edge set of the underlying undirected graph of D. A feedback edge set F of an undirected graph G is a subset of the edges whose removal makes the graph acyclic, that is, G can be seen as a tree with the additional edges in F. This implies that deleting in D a vertex with in- or outdegree at least $k+2$ results in a disconnected DAG: Deleting a vertex of degree $k+2$ in a tree results in $k+2$ connected components. At most $k+1$ of these $k+2$ components can be pairwise connected by the edges in the feedback edge set.

If there are two connected components in a realizing DAG D, then we can restructure D by copying parts from one component into the other component without creating cycles. This restructuring of a realizing DAG is our core tool to develop the data reduction rule and is formally stated in the next lemma.

See Figure 4.6 for a schematic view on the requirements and the statement of the lemma.

Lemma 4.29. *Let $D = (V, A)$ be a realizing DAG for a degree sequence $\mathcal{S}$, and let $v^p, v^t \in V$ be two vertices with $(v^p, v^t) \in A$. Furthermore, let K be the vertices of the connected component in $D[V \setminus \{v^t\}]$ containing the vertex v^p and let $v^s, u, w, w' \in V$ be vertices such that:*

(i) $(v^s, u), (w, w') \in A$,

(ii) $v^s, u \in K$, *but* $w, w' \notin K$,

(iii) the underlying undirected graph of $D[K]$ is acyclic,

(iv) u lies on the uniquely defined undirected path between v^p and v^s or $u = v^p$, and

(v) v^p is the only neighbor of v^t in K.

Then, the digraph

$$D' = (V, (A \setminus \{(v^s, u), (v^p, v^t), (w, w')\}) \cup \{(v^s, v^t), (w, u), (v^p, w')\})$$

is a realizing DAG for $\mathcal{S}$. Furthermore, the underlying undirected graph of the connected component in $D'[V \setminus \{v^t\}]$ which contains v^s is acyclic and contains no further neighbor of v^t.

Proof. Let K' be the connected component in $D'[V \setminus \{v^t\}]$ which contains the vertex v^s and let K'_u be the underlying undirected graph of K'. We first prove that K'_u is acyclic. Observe that by assumption (iii) the underlying undirected graph K_u of K is acyclic. Hence, K_u is a tree. Next, root the tree K_u in the vertex v^s. Since by assumption (iv) the vertex u lies on the (uniquely defined) path between v^s and v^p, the graph K'_u accords in K_u to the subtree with root v^s. Hence, K'_u is acyclic and contains no further neighbor of v^t.

Next, we prove that D' is a realizing DAG for $\mathcal{S}$. Clearly, the vertices v^s, u, v^t, v^p, w, and w' have the same indegree and outdegree in D' as in D. To show that D' does not contain any cycle, assume towards a contradiction that there is a directed cycle C' in D'. Recall that K'_u is acyclic and, hence, also K' is acyclic. From this and from assumption (v) it follows that C' cannot contain any vertex of K'. Since D is acyclic, this implies that at least one of the arcs $(w, u), (v^p, w')$

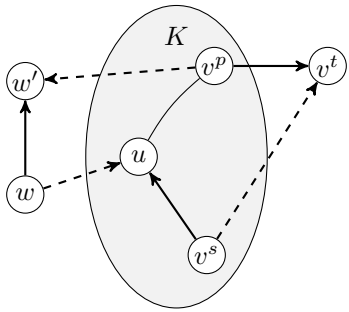


Figure 4.6.: The situation when Lemma 4.29 is applicable: The underlying undirected graph of the induced subgraph K is acyclic and there is only one arc connecting a vertex inside K with a vertex outside K: the arc (v^p, v^t). The vertex v^s is inside of K with the outneighbor u lying on the uniquely defined undirected path (indicated by the thin edge) between v^s and v^p (with the possibility that $u = v^p$). Furthermore, (w, w') is some arc with w and w' lying outside of K. Then deleting the solid arcs in the picture and adding the dashed arcs results again in a DAG where the degrees of the vertices remain unchanged.

is contained in C'. Denote with $\widetilde{K}$ the subgraph containing all vertices that are in K but not in K', that is, the graph induced by the vertices that are cut out of K. By assumption (iii) also $\widetilde{K}$ is acyclic and, thus, by assumption (ii) C' has to contain both arcs $(w, u), (v^p, w')$. This implies that in D' and so in D there is a directed path from w' to w, implying that, since $(w, w') \in A$, there is also a cycle in D, a contradiction. □

Observe that the version of Lemma 4.29, where all arcs appear in the reversed direction, is also true. To see this, first swap in every tuple $\binom{a}{b}$ in $\mathcal{S}$ the values of a and b and, correspondingly, swap in a realizing DAG the direction of each arc. Then, apply Lemma 4.29 and finally swap again the values in the tuples and the arcs in the restructured DAG.

Using the restructuring operation provided by Lemma 4.29 and its reverted-arc version, we can show the following.

Lemma 4.30. *Let $\mathcal{S}$ be a realizable degree sequence with $k = m - n + 1 > 0$ and let $t = \binom{a}{b} \in \mathcal{S}$ be a tuple with $a > 2k$. Furthermore, let $s_{\min} \in \mathcal{S}$ be a source with minimum outdegree. Then, there is a realizing DAG for $\mathcal{S}$ such that the vertex that corresponds to t is an outneighbor of the vertex $v^s_{\min}$ which*

corresponds to $s_{\min}$. Furthermore, all other outneighbors of $v^s_{\min}$ are degree-one sinks.

Proof. Let $\mathcal{S}$ be a realizable degree sequence and let $t = \binom{a}{b} \in \mathcal{S}$ be a tuple with $a > 2k$. Furthermore, let $D = (V, A)$ be a connected realizing DAG for $\mathcal{S}$ and denote by v^t the vertex that corresponds to t. We prove the statement of the lemma in three steps. First, we show in Step 1 that we can assume that v^t has some source v^s as inneighbor such that the connected component of the underlying undirected graph of $D[V \setminus \{v^t\}]$ containing v^s is acyclic and v^s is the only vertex in this connected component that is adjacent to v^t. In Step 2, we prove that we can replace this source by $v^s_{\min}$ such that the connected component of the underlying undirected graph of $D[V \setminus \{v^t\}]$ containing $v^s_{\min}$ is acyclic and $v^s_{\min}$ is the only vertex in this connected component that is adjacent to v^t. Finally, in Step 3 we show that we can replace all outneighbors of $v^s_{\min}$ except v^t by degree-one sinks. The reason for these restrictive requirements for the outcome in Steps 1 and 2 is that we apply Lemma 4.29 in Steps 2 and 3.

Step 1: Assume that v^t has no source as inneighbor satisfying the outcome we want after Step 1; otherwise go to Step 2. As k is the size of a feedback edge set in the underlying undirected graph of D and $a > 2k$, there are at least $k + 1$ connected components in $D[V \setminus \{v^t\}]$ and at most k of them can contain a cycle or more than one neighbor of v^t. Thus, there is at least one connected component K in $D[V \setminus \{v^t\}]$ that contains only one neighbor (an inneighbor), say v^p, of v^t and does not contain any cycle in the underlying undirected graph of $D[V \setminus \{v^t\}]$. Clearly, K contains a source, say v^s, such that there is a directed path from v^s to v^p. Let u be the outneighbor of v^s lying on this path (with u not necessarily being distinct from v^p). Furthermore, let $(w, w') \in A$ be an arc where both endpoints w and w' are in $D[V \setminus \{v^t\}]$ in a connected component different from that of v^s. Observe that such an arc must exist, as otherwise, by the choice of v^s, the underlying undirected graph of D would be acyclic, contradicting the assumption $k > 0$. Then, by Lemma 4.29, $D' = (V, (A \setminus \{(v^s, u), (v^p, v^t), (w, w')\}) \cup \{(v^s, v^t), (w, u), (v^p, w')\})$ is also a realizing DAG for $\mathcal{S}$. Furthermore, denoting the connected component in $D'[V \setminus \{v^t\}]$ containing v^s by K_s, only s in K_s is a neighbor of v^t and the underlying undirected graph of K_s is acyclic.

Step 2: Assume that $\deg^+(v^s) > \deg^+(v^s_{\min})$; otherwise go to Step 3. We show that we can replace v^s as inneighbor of v^t by $v^s_{\min}$. To this end, we distinguish two cases.

Case 1: $v^s_{\min} \notin K_s$: Let $s_1, s_2, \ldots, s_x$ be the outneighbors of v^s with $s_1 = v^t$ and let $o^s_1, o^s_2, \ldots, o^s_y$ be the outneighbors of $v^s_{\min}$. Observe that $v^s_{\min}$ and v^s have pairwise disjoint sets of outneighbors and that $x > y$. Now, we obtain a directed graph D'' from D' by deleting the arcs from v^s to all $s_{x+1}, s_{x+2}, \ldots, s_y$ and by adding arcs from $v^s_{\min}$ to all $s_{x+1}, s_{x+2}, \ldots, s_y$. Since in D'' compared to D' the outdegrees of $v^s_{\min}$ and v^s have been exchanged, v^s is a minimum-degree source in D''. Since $v^s_{\min}$ and v^s are sources and all arcs that have been modified to get D'' from D' have either $v^s_{\min}$ or v^s as an endpoint and a source can never be contained in a directed cycle, D'' is a realizing DAG for $\mathcal{S}$. Furthermore, since K_s does not contain any cycle in $D[V \setminus \{v^t\}]$ and all modifications involving v^s did only delete arcs, it follows that also the connected component of v^s in $D''[V \setminus \{v^t\}]$ does not contain any undirected cycle. For convenience, we also exchange the names of v^s and $v^s_{\min}$ so that in the following $v^s_{\min}$ is the source with minimum outdegree.

Case 2: $v^s_{\min} \in K_s$: Since K_s induces a (directed) tree in D', there is a unique undirected path from v^s to $v^s_{\min}$ in K_s. Denote by $o^s_{\min}$ the outneighbor of $v^s_{\min}$ on this path. Furthermore, let $(w, w') \in A$ be an arc where both endpoints w and w' are in $D'[V \setminus \{v^t\}]$ in a connected component different from that of v^s. Observe that such an arc must exist, see the argumentation in Step 1. By Lemma 4.29 the digraph

$$D'' = (V, A' \setminus \{(w, w'), (v^s, v^t), (v^s_{\min}, o^s_{\min})\} \cup \{(w, o^s_{\min}), (v^s_{\min}, v^t), (v^s, w')\})$$

is a realizing DAG for $\mathcal{S}$. Furthermore, denoting the connected component in $D''[V \setminus \{v^t\}]$ containing $v^s_{\min}$ by K'', only $v^s_{\min}$ in K'' is a neighbor of v^t and the underlying undirected graph of K'' is acyclic.

Step 3: By the argumentation above there is a realizing DAG D'' for $\mathcal{S}$ such that there is an inneighbor of v^t that is a minimum degree source $v^s_{\min}$. By restructuring the arcs in D'', we show that there is also a realizing DAG such that except for v^t all outneighbors of $v^s_{\min}$ are degree-one sinks. Towards this, assume that $v^s_{\min}$ does have an outneighbor u that is not a degree-one sink and that is different from v^t; otherwise we are done. Root the connected component K'' of $v^s_{\min}$ in $D''[V \setminus \{v^t\}]$ in $v^s_{\min}$. Then there is at least one leaf v^ℓ in the subtree with root u. Since $v^s_{\min}$ is the source with the minimal outdegree and $v^s_{\min}$ has at least two outneighbors (v^t and u), it follows that v^ℓ is a degree-one sink. Let $v^{\ell'}$ be the inneighbor of this sink. Furthermore, let $(w, w') \in A''$ be an arc that is not contained in K''. Since the underlying undirected graph of K'' is a tree, by the version of Lemma 4.29 with

all arcs being reverted (see the discussion before Lemma 4.30), the digraph $(V, A'' \setminus \{(w, w'), (v^s_{\min}, u), (v^{\ell'}, v^\ell)\} \cup \{(v^s_{\min}, v^\ell), (w, u), (v^{\ell'}, w')\})$ is also a realizing DAG for $\mathcal{S}$. By repeatedly applying this procedure, we reorder the realizing DAG such that all outneighbors of $v^s_{\min}$ except v^t are degree-one sinks. □

Lemma 4.30 shows how we can restructure a realizing DAG if there is a vertex with indegree greater than $2k$. By applying the same procedure as for Lemma 4.29 procedure, one obtains similar results in case of a high outdegree: First, swap in every tuple $\binom{a}{b}$ in the corresponding degree sequence the values of a and b and, correspondingly, swap in a realizing DAG the direction of each arc. Then, apply the Lemmas 4.29 and 4.30 and finally swap again the values in the tuples and the arcs in the restructured DAG. This proves the correctness of an analogous version of Lemma 4.30 for a vertex with outdegree greater than $2k$, which leads to the following data reduction rule.

Reduction Rule 4.1. *Let $\mathcal{S}$ be degree sequence containing a tuple $\binom{a_t}{b_t}$ with $a_t > 2k$ ($b_t > 2k$). Furthermore, let $s = \binom{a_s}{b_s} \in \mathcal{S}$ be a tuple with $a_s = 0$ ($b_s = 0$) and b_s (a_s) be minimal among all tuples with $a_s = 0$ ($b_s = 0$). Then, replace $\binom{a_t}{b_t}$ by $\binom{a_t-1}{b_t}$ ($\binom{a_t}{b_t-1}$), delete s, and delete $b_s - 1$ ($a_s - 1$) tuples of the form $\binom{1}{0}$ ($\binom{0}{1}$).*

Clearly, Reduction Rule 4.1 can be applied in polynomial time. Furthermore, each application of Reduction Rule 4.1 decreases the number of tuples in the input by one. Thus, exhaustively applying Reduction Rule 4.1 can be done in polynomial time. In addition, the maximum degree in an instance that is reduced with respect to Reduction Rule 4.1 is upper-bounded by $2k$. Together with Theorem 4.27 this implies the following.

Theorem 4.31. DAG Realization *is fixed-parameter tractable with respect to the parameter $m - n + 1$.*

The running time is similar to the one in Theorem 4.27: $k^{k^{k^{\mathcal{O}(k)}}} \cdot n^{\mathcal{O}(1)}$.

4.5. Conclusion

Answering an open question of Berger and Müller-Hannemann [BM11] we proved the NP-completeness of DAG Realization even in sparse and in dense graphs. Following the spirit of deconstructing intractability [KNU11] we

proved the necessity of large degrees in the NP-hardness proof by showing fixed-parameter tractability for DAG Realization with respect to the maximum degree Δ. Furthermore, we showed fixed-parameter tractability with respect to the feedback edge set size of the underlying undirected graph of a realizing DAG. It is open whether DAG Realization is solvable in single-exponential FPT time and whether it admits polynomial-size problem kernels with respect to these two parameters. In our NP-hardness reduction other parameters occur with unbounded values, for instance, the number of different tuples in the degree sequence. Note that this is a "stronger parameterization" [KN12] than the parameter maximum degree Δ as the number of different tuples is at most $(\Delta + 1)^2$. Hence, investigating this parameter is an interesting task for future work.

Chapter 5.

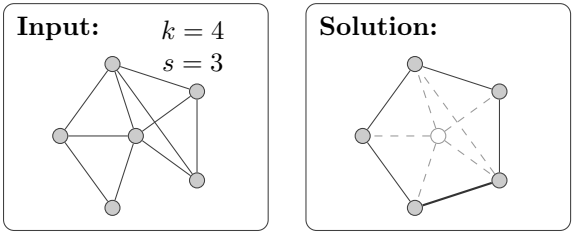
Degree Anonymity by Vertex and Edge Deletion

In this chapter, we examine the computational complexity of making a given undirected graph k-anonymous by either at most s vertex deletions or at most s edge deletions; the corresponding problems are Anonym V-Del and Anonym E-Del. We present numerous hardness results for these problems, showing strong intractability of both problems from the parameterized as well as from approximation point of view.

5.1. Introduction

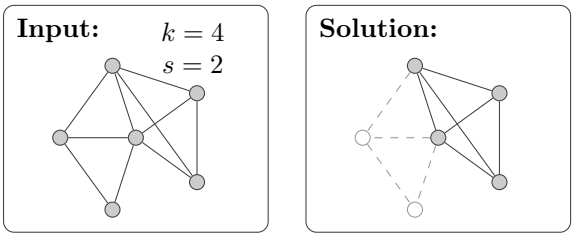
With the enormously growing relevance of social networks, the protection of privacy when releasing underlying data sets has become an important and active field of research [Wu+10]. If a graph contains only few vertices with some distinguished feature, then this might allow the identification (and violation of privacy) of the underlying real-world entities with that particular feature. Hence, in order to ensure pretty good privacy and anonymity, every vertex should share its feature with many other vertices. In a landmark paper, Liu and Terzi [LT08] considered the vertex degrees as feature; see Wu et al. [Wu+10] for other features considered in the literature. Correspondingly, a graph is called k-anonymous if for each vertex there are at least $k-1$ other vertices of same degree. Therein, different values of k reflect different privacy demands and the natural computational task arises, given some fixed k, to perform few changes to a graph in order to make it k-anonymous.

Degree Anonymity (Anonym)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Can G be transformed with at most s modification operations into a k-anonymous graph $G' = (V', E')$, that is, for each vertex in G' there are $k - 1$ other vertices of the same degree?



Liu and Terzi [LT08] proposed a heuristic algorithm for the task of making a graph k-anonymous by adding edges. We refer to Section 1.2 for a thorough discussion about the model and related work. In this chapter, we study the vertex and edge deletion variants of Degree Anonymity. We start our investigations with the vertex deletion variant which is defined as follows.

Degree Anonymity by Vertex Deletion (Anonym V-Del)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Is there a vertex subset $S \subseteq V$ of size at most s such that $G - S$ is k-anonymous?



Considering vertex deletions seems to be a promising approach on practical instances, especially on social networks. In these networks, the degree distribution of the underlying graphs often follows a so-called power law distribution [BA99], implying that there are only few high-degree vertices and most vertices are of moderate degree; this suggests that only few vertices have to be removed in order to obtain a k-anonymous graph. For instance, consider the DBLP co-author

graph[1] (generated in Feb. 2012) with ≈ 715 thousand vertices corresponding to authors and ≈ 2.5 million edges indicating whenever two authors have a common scientific paper: This graph has maximum degree 804 but only 208 vertices are of degree larger than 208, whereas the average degree is 7. Interestingly, a heuristic that simply removes vertices violating the k-anonymous property shows that one has to remove at most 338 vertices to make it 5-anonymous and even to make it 10-anonymous requires at most 635 vertex deletions.

In Section 5.2, we will show that already the simple and highly specialized privacy model of Anonym V-Del is computationally hard from the parameterized as well as from the approximation point of view. A variety of hardness results holds even for very restricted graph classes, as for instance trees, cographs, and split graphs.

One reason of this hardness is that being k-anonymous is not a hereditary property: Simply deleting one vertex in a three-regular graph, that is, an n-anonymous graph, results in an only 3-anonymous graph. Another reason is shown in the following two examples illustrating that the number s of allowed removals and the anonymity level k are independent of each other, and that a small change in one of these parameter values might lead to a large jump of the other parameter value.

Example 2. Let G be an undirected graph on $n \geq 5$ vertices that consists of two connected components: a clique of order $n-2$ and an isolated edge. This 2-anonymous graph cannot be transformed into a 3-anonymous graph by deleting only one vertex; however, deleting the two degree-one vertices makes it $(n-2)$-anonymous. Hence, by slightly increasing s from 1 to 2 the reachable anonymity level jumps from $k = 2$ to $k = n-2$.

Example 3. Let $G = (V, E)$ be an undirected bipartite graph with the disjoint vertex sets $X := \{x_1, x_2, \ldots, x_\ell\}$ and $Y := \{y_1, y_2, \ldots, y_\ell\}$, $V = X \cup Y$, and there is an edge between x_i and y_j if $i + j > \ell$, see Figure 5.1 for a visualization. Clearly, x_i and y_i are of degree i implying that G is 2-anonymous. Since $N(x_i) \subseteq N(x_{i+1})$ for all i, deleting any subset of Y preserves the invariant $\deg(x_1) \leq \deg(x_2) \leq \ldots \leq \deg(x_\ell)$. As the previous argument is symmetric, one can observe that to make G 3-anonymous one has to remove $2/3$ of the "jumps" in the initial sequences $\deg(x_1) < \deg(x_2) < \ldots < \deg(x_\ell)$ and $\deg(y_1) < \deg(y_2) < \ldots < \deg(y_\ell)$. Since removing one vertex in X (Y) removes only one jump in the sequence of X (Y) and only one in Y (X), it follows that at

[1] The current dataset and a corresponding documentation are available online (http://dblp.uni-trier.de/xml/).

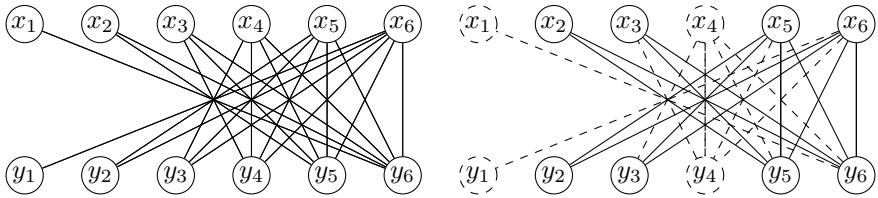
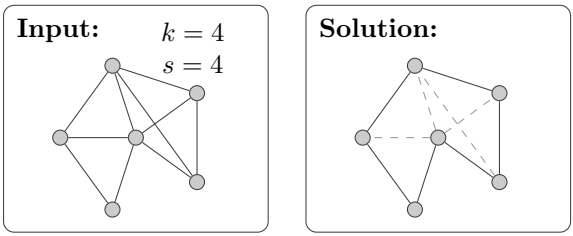


Figure 5.1.: *Left:* A graph where a constant fraction of the vertices has to be removed in order to obtain a 3-anonymous graph. *Right:* A minimum size solution to make the graph on the left side 3-anonymous. See Example 3 for a detailed explanation.

least $2(\ell-1)\cdot 2/3\cdot 1/2 \approx (2\ell)/3 = |V|/3$ vertices have to be deleted in order to get a 3-anonymous graph. Summarizing, by requiring anonymity level $k=3$ instead of anonymity level $k=2$, the number of vertices that need to be removed jumps from zero to a constant fraction of the vertices.

The second part of this chapter deals with the edge deletion variant which is defined as follows:

Degree Anonymity by Edge Deletion (Anonym E-Del)
Input: An undirected graph $G=(V,E)$ and two integers $k,s\in\mathbb{N}$.
Question: Is there an edge subset $S\subseteq E$ of size at most s such that $G-S$ is k-anonymous?



Considering social networks, their power law degree distribution suggests that the solution size in the edge deletion variant is significantly larger than in the vertex deletion variant. However, in the edge deletion variant the resulting graph contains, by definition, all vertices of the input graph, which might be important in some applications. Furthermore, deleting a vertex with high degree reduces the degree of many other vertices whereas deleting an edge reduces the degree of only two vertices. Hence, although requiring more edge deletions than

vertex deletions, deleting edges might result in a graph that is actually "closer" to the original graph.

In Section 5.3, we transfer most hardness results from Anonym V-Del to Anonym E-Del, showing strong intractability results concerning parameterized complexity and approximability. Similarly to the vertex deletion variant, a small change in one of the two parameters k and s might lead to a large jump of the other parameter as demonstrated in the following two examples.

Example 4. Let G be an n-vertex cycle with two chords, that is, two additional edges within the cycle. As G contains four degree-three vertices and $n-4$ degree-two vertices, G is 4-anonymous. Deleting one edge does not increase the anonymity level k; however, deleting the two chords results in an n-anonymous graph—a cycle. Hence, by slightly increasing s from one to two the reachable anonymity level jumps from $k=4$ to $k=n$.

Example 5. Let $G=(V,E)$ be a disjoint union of a clique and an independent set, each containing $n/2$ vertices. Thus, G is $n/2$-anonymous. However, in order to obtain an $(n/2+1)$-anonymous graph, all edges have to be removed. Hence, by slightly increasing k from $n/2$ to $n/2+1$ the number of edges that have to be removed jumps from zero to $|E| = \binom{n/2}{2}$.

Related work. For a discussion about the k-anonymity model, introduced by Liu and Terzi [LT08], we refer to Section 1.2. Concerning the vertex deletion variant, the work which is probably closest to ours is by Moser and Thilikos [MT09]. They studied the parameterized complexity of the Regular-Degree-d Vertex Deletion problem, where given an undirected graph G and an integer $s \in \mathbb{N}$, the task is to decide whether G can be made d-regular by at most s vertex deletions. Moser and Thilikos [MT09] showed that Regular-Degree-d Vertex Deletion can be solved in $\mathcal{O}(n(s+d)+(d+2)^s)$ time and presented a polynomial problem kernel of size $\mathcal{O}(sd(d+s)^2)$. Fellows et al. [Fel+11] provided an "improved" problem kernel of size $\mathcal{O}(s^{1+\varepsilon})$ for any constant degree d and $\varepsilon > 0$. Observe that for $k > n/2$ the problem of Anonym V-Del asks whether at most s vertices can be deleted to obtain a regular graph.

Concerning the edge deletion variant, there is a close connection to the edge insertion variant which we will consider in Chapter 6. Anonym E-Del can also be seen as degree factor problem, see Section 3.2 for a thorough discussion.

Our contributions. While every graph is trivially 1-anonymous, we will show that the combinatorial structure of 2-anonymous graphs is already rich and

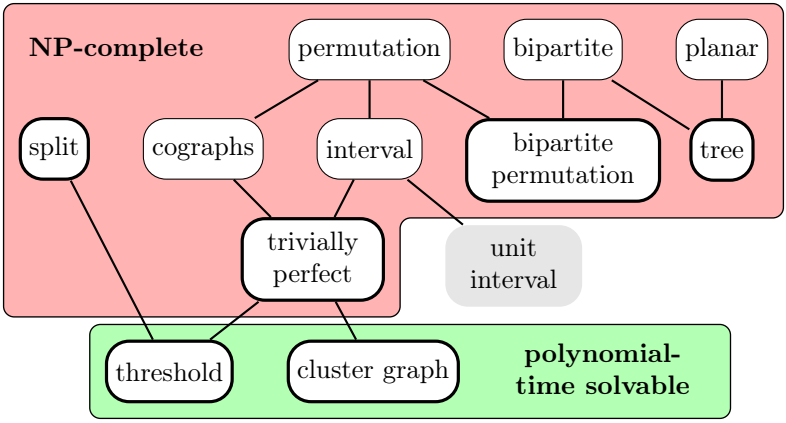


Figure 5.2.: The complexity landscape of Anonym V-Del for various graph classes. The results for classes with thick frames are discussed in this chapter and they imply the results for classes with thin frames. The complexity of Anonym V-Del on unit interval graphs remains open.

complicated: Anonym V-Del for $k = 2$ is NP-complete, even for strongly restricted graph classes like trees, interval graphs, split graphs, trivially perfect graphs, and bipartite permutation graphs. All these hardness results are established by means of a general framework. Furthermore, we show that Anonym V-Del is NP-complete even on graphs with maximum degree three.

On the positive side, we present (polynomial-time) dynamic programming approaches for Anonym V-Del on three graph classes: graphs of maximum degree two, cluster graphs, and threshold graphs. We frankly admit that these three graph classes carry an *extremely constraining* combinatorial structure: Anonym V-Del is such a vicious problem that without these heavily constraining structures there is basically no hope for polynomial-time algorithms. Figure 5.2 summarizes the considered graph classes and their containment relations.

For Anonym E-Del, we show NP-completeness on caterpillars and on graphs with maximum degree seven; this later result is in stark contrast with the fixed-parameter tractability of Anonym E-Ins with respect to the maximum degree Δ (see Theorem 6.16).

Table 5.1.: Overview on the computational complexity classification of Anonym V-Del and Anonym E-Del.

Parameter	Anonym V-Del	Anonym E-Del
k	NP-complete for $k = 2$ (Theorem 5.2 on page 102)	NP-complete for $k = 2$ (Theorem 5.17 on page 117)
(s, k)	W[2]-hard (Corollary 5.6 on page 107)	W[1]-hard (Corollary 5.22 on page 124)
Δ	NP-complete for $\Delta = 3$ (Theorem 5.1 on page 99)	NP-complete for $\Delta = 7$ (Theorem 5.19 on page 119)
(s, Δ)	FPT (Theorem 5.23 on page 125)	
(k, Δ)	FPT (Corollary 5.26 on page 129)	

We analyze the parameterized complexity of Anonym V-Del and Anonym E-Del, see Table 5.1 for an overview. Once again, both problems show a difficult and challenging behavior: They are intractable with respect to each of the three (single) parameters s, k, and Δ. Furthermore, they are intractable with respect to the combined parameter (s, k). The only positive parameterized results come with the combined parameters (Δ, s) and (Δ, k). The latter result is based on bounding the number s of deleted vertices in terms of Δ and k.

Finally, studying the approximability of the optimization problems naturally associated with Anonym E-Del or Anonym V-Del, we obtain hardness results showing that none of the considered problems can be approximated in polynomial time better than within a factor of $n^{1/2}$. Furthermore, for the optimization variants where the solution size s is given and the task is to maximize the anonymity level k, this inapproximability even holds if we allow a running time of $f(s)n^{\mathcal{O}(1)}$ for any computable f. Again, this result holds for the edge deletion and the vertex deletion variant, see Table 5.2 for an overview.

Organization. We first provide our results for Anonym V-Del in Section 5.2, starting with the NP-completeness results. To this end, we present in Section 5.2.1 a reduction showing NP-hardness on trees. This reduction serves in Section 5.2.2 as blueprint for a generic reduction yielding NP-hardness on several restricted graph classes. In Section 5.2.3, we then adjust this reduction in order to prove the inapproximability results for Anonym V-Del. We present the polynomial-time solvable cases of Anonym V-Del in Section 5.2.4. In

Table 5.2.: Overview on the inapproximability of the optimization variants associated with Anonym V-Del and Anonym E-Del.

vertex deletion running time	Anonym Min-V-Del (fixed k, minimize s)	Max-Anonym V-Del (fixed s, maximize k)
polynomial time	no $n^{1-\varepsilon}$-approximation (Theorem 5.10 on page 109)	no $n^{1/2-\varepsilon}$-approximation (Theorem 5.12 on page 112)
$f(s) \cdot n^{O(1)}$	open	no $n^{1/2-\varepsilon}$-approximation (Theorem 5.11 on page 110)
edge deletion running time	Anonym Min-E-Del (fixed k, minimize s)	Max-Anonym E-Del (fixed s, maximize k)
polynomial time	no $n^{1-\varepsilon}$-approximation (Theorem 5.20 on page 121)	no $n^{1-\varepsilon}$-approximation (Theorem 5.19 on page 119)
$f(s) \cdot n^{O(1)}$	open	no $n^{1-\varepsilon}$-approximation (Theorem 5.21 on page 122)

Section 5.3, we transfer the central intractability results for Anonym V-Del to Anonym E-Del. In particular, we show in Section 5.3.1 that Anonym E-Del is NP-complete on caterpillars. In Section 5.3.2, we then give the inapproximability results. Finally, we show in Section 5.4 the fixed-parameter tractability of Anonym V-Del and Anonym E-Del with respect to the combined parameters (s, Δ) and (s, k).

5.2. Vertex Deletion

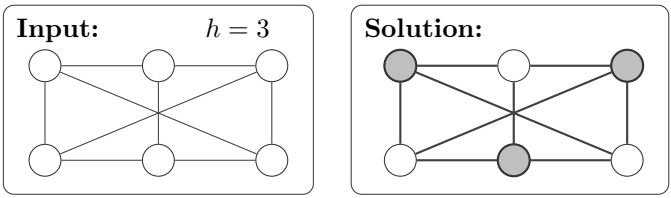
In this section, we provide various hardness results for Anonym V-Del on several restricted graph classes. In Section 5.2.1, we show that Anonym V-Del remains NP-hard even on trees. Extracting the basic ideas of this result, subsequently we provide a generic reduction to show NP-hardness on trivially perfect graphs, bipartite permutation graphs, and split graphs (see Section 5.2.2) and strong inapproximability results for the two natural optimization problems associated with Anonym V-Del (see Section 5.2.3). We also identify several classes of graphs for which Anonym V-Del is polynomial-time solvable (see

Section 5.2.4). As a warm up, we first prove that Anonym V-Del is NP-complete on graphs with maximum degree three.

Theorem 5.1. Anonym V-Del *is NP-complete on graphs with maximum degree three.*

Proof. Since containment in NP is obvious, we focus on showing NP-hardness. To this end, we give a reduction from the Vertex Cover problem which is known to be NP-complete even in three-regular graphs [GJ79, GT1] and is formally defined as follows.

Vertex Cover [GJ79, GT1]
Input: An undirected graph $G = (V, E)$ and $h \in \mathbb{N}$.
Question: Is there a vertex subset $V' \subseteq V$, $|S| \leq h$, such that every edge has an endpoint in V'?



Given a Vertex Cover instance $(G = (V, E), h)$ with G being three-regular, start by copying G into a new graph G'. Finally, add $h + 1$ degree-zero vertices to G', set $s := h$, and $k := |V| + 1$.

If G contains a vertex cover V' of size h, then deleting V' in G' clearly results in an edgeless graph with $|V| + 1 = k$ vertices, implying that (G', s, k) is a yes-instance of Anonym V-Del. In the reverse direction, for any k-deletion set S, since $2k > n + h + 1$ and G' contains $s + 1$ degree-zero vertices, all vertices in $G' - S$ have degree zero. Thus, $S \cap V$ is a vertex cover in G. □

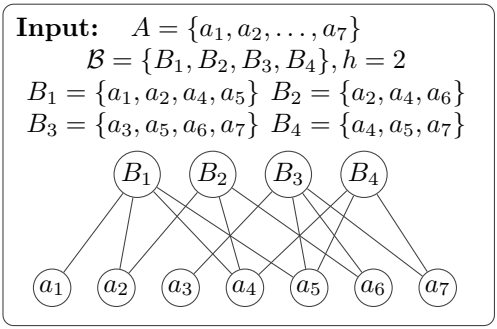
5.2.1. NP-Hardness on Trees

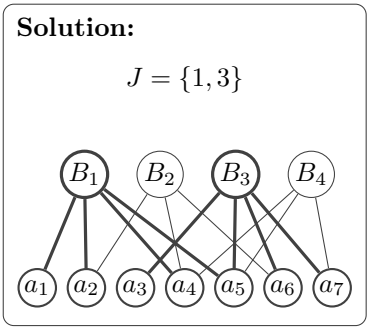
In this section, we show that Anonym V-Del remains NP-hard even on trees. This result and many further hardness results will be obtained using reductions from the NP-complete Set Cover problem, which is defined as follows:

Set Cover [GJ79, SP5]

Input: A universe $A = \{a_1, a_2, \ldots, a_\alpha\}$, a collection $\mathcal{B} = \{B_1, B_2, \ldots, B_\beta\}$ of subsets of A, and $h \in \mathbb{N}$.

Question: Is there an index set $J \subseteq \{1, 2, \ldots, \beta\}$ with $|J| \leq h$, such that $\bigcup_{j \in J} B_j = A$?





If a Set Cover instance $I = (A, \mathcal{B}, h)$ contains such an index set J, then we refer to the set $\{B_j \mid j \in J\}$ as a *set cover* for I.

Reduction 1. The reduction showing NP-hardness of Anonym V-Del on trees is as follows: Let $(A, \mathcal{B}, h)$ be an instance of Set Cover. We assume without loss of generality that for each element $a \in A$ there exists a set $B \in \mathcal{B}$ with $a \in B$. Furthermore, we assume without loss of generality that each set $B \in \mathcal{B}$ occurs at least three times in $\mathcal{B}$. To decrease the amount of indices in the construction given below we introduce the function $f \colon \mathbb{N} \to \mathbb{N}$ with $f(i) = \alpha + (h+1)i$.

The reduction for trees is as follows, see Figure 5.3 for an example. Set $k := 2$ and $s := h$ such that (G, k, s) is an equivalent Anonym V-Del-instance. The graph $G = (V, E)$ is constructed as follows: For each element $a_i \in A$ add an *element gadget* consisting of a star $K_{1,f(i)}$ with the center vertex $v(a_i)$. Denote with $V_A := \{v(a_1), v(a_2), \ldots, v(a_\alpha)\}$ the set of all these center vertices.

For each set $B_j \in \mathcal{B}$ add a *set gadget* which is a tree rooted in a vertex $v(B_j)$. The root has $|B_j|$ child vertices where each element $a_i \in B_j$ corresponds to exactly one of the children of $v(B_j)$, denoted by $v(a_i, B_j)$. Additionally, we add to $v(a_i, B_j)$ exactly $f(i)$ degree-one neighbors. Hence, the set gadget is a tree of depth two rooted in $v(B_j)$. We denote with $V_\mathcal{B} := \{v(B_1), v(B_2), \ldots, v(B_\beta)\}$ the set of all root vertices. Observe that, as each set $B_j \in \mathcal{B}$ occurs at least three times, the set gadgets are 2-anonymous. Finally, to end up with one tree

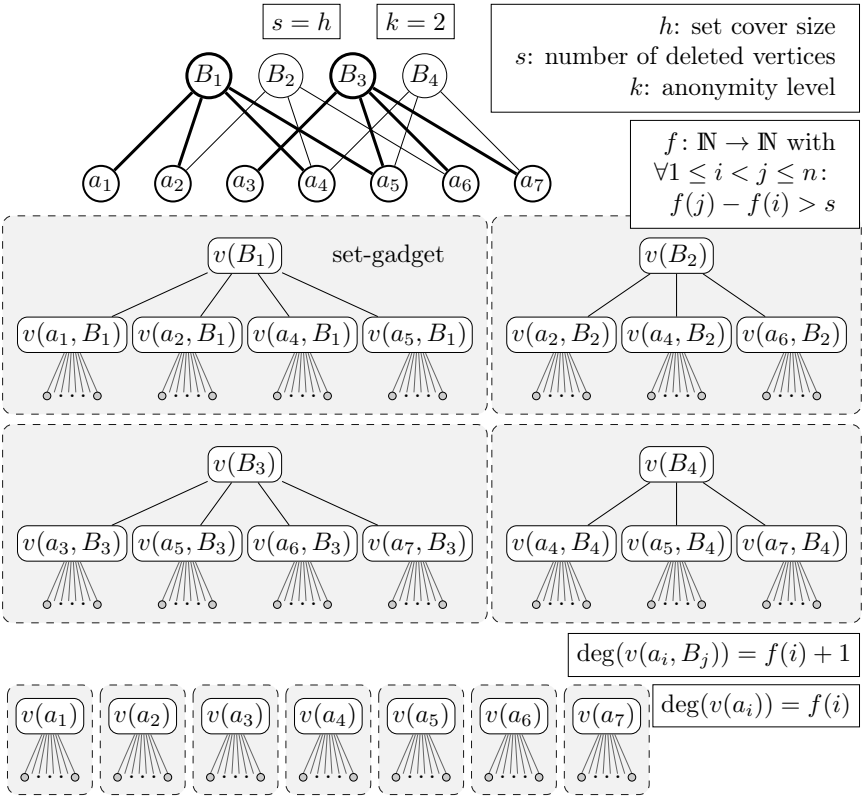


Figure 5.3.: Example of the reduction for trees. Above the Set Cover instance with twelve sets (each set B_i , $i = 1, \ldots, 4$ appears three times) and seven elements is graphically displayed (for example, the set B_1 contains the elements a_1, a_2, a_4, and a_5, and $\{B_1, B_3\}$ forms a set cover). In our reduction, we assume without loss of generality that each set occurs at least times. However, to keep the figure clearly arranged, we omit these copies in the figure. Below are the four different set gadgets and the element gadgets are at the bottom of the picture. Observe that by the choice of f, the degrees of the vertices in the set-gadgets and vertex-gadgets are ensured to not interfere, even if s vertices are removed. The effect of these copies to the construction is that each of the four set-gadgets appears three times. Thus, deleting the vertices $v(B_1)$ and $v(B_3)$ makes the displayed graph 2-anonymous.

instead of a forest, repeatedly add edges between any degree-one-vertices of different connected components.

Correctness of Reduction 1 Observe that for each element $a_i \in A$ the only vertex of degree $f(i)$ is $v(a_i)$ and there are no other vertices violating the 2-anonymous property. The key point in the construction is that, in order to get a 2-anonymous graph, one has to delete vertices of $V_{\mathcal{B}}$: Let $a_i \in A$ be an element and $v(B_j)$ a root vertex such that $a_i \in B_j$. By construction the child vertex $v(a_i, B_j)$ of $v(B_j)$ corresponds to a_i and therefore has $f(i)$ child vertices. Thus, deleting $v(B_j)$ lowers the degree of $v(a_i, B_j)$ to $f(i)$ and, hence, $v(a_i)$ no longer violates the 2-anonymous property. Furthermore, as each set $B_j \in \mathcal{B}$ occurs at least three times, the vertices $V_{\mathcal{B}}$ are 2-anonymous. Hence, given a set cover one can construct a corresponding k-deletion set of the same size and, thus, if $(A, \mathcal{B}, h)$ is a yes-instance, then (G, k, s) is a yes-instance. The basic idea in the converse direction is that due to the choice of f, deleting vertices from $V_{\mathcal{B}}$ is the only possibility to make the graph 2-anonymous. The formal proof which implies the following theorem will be given later (see Lemma 5.4), after introducing the generic reduction.

Theorem 5.2. Anonym V-Del *is NP-complete on trees even if* $k = 2$.

5.2.2. Generic Reduction

In this section, we generalize Reduction 1 given in the previous subsection. More specifically, we will define properties such that a graph G fulfilling them together with $s := h$ and $k := 2$ forms a yes-instance of Anonym V-Del if and only if the given Set Cover instance $(A, \mathcal{B}, h)$ is a yes-instance. Based on that, we then describe the construction of several graphs contained in different graph classes and fulfilling the properties. Formally, we require the constructed graph $G = (V, E)$ to fulfill the following:

1. Element-gadgets:
   (a) For each element $a_i \in A$ there is a corresponding vertex, denoted by $v(a_i)$, in G and the vertex set $V_A := \{v(a_1), v(a_2), \ldots, v(a_\alpha)\}$ is exactly the set of vertices not being 2-anonymous in G.
   (b) For each vertex $v \in V$ it holds that $|N[v] \cap V_A| \leq 1$.
2. Set-gadgets:

(a) For each set $B_j \in \mathcal{B}$ there is a corresponding vertex $v(B_j)$ in G and for each element $a_i \in B_j$ the vertex $v(B_j)$ has a neighbor $v(a_i, B_j)$ with $\deg(v(a_i, B_j)) = \deg(v(a_i)) + 1$.
Set $V_{\mathcal{B}} := \{v(B_1), v(B_2), \ldots, v(B_\beta)\}$ and $A_{B_j} := \{v(a_i, B_j) \mid a_i \in B_j\}$.
Set $A_{\mathcal{B}} := \bigcup_{B_j \in \mathcal{B}} A_{B_j}$.

(b) For all $B_j \in \mathcal{B}$ it holds that $N(A_{B_j}) \cap V_{\mathcal{B}} = \{v(B_j)\}$.

(c) For each vertex $v \in V$ there is a vertex $u \in V_{\mathcal{B}}$ such that $N(v) \cap A_{\mathcal{B}} \subseteq N(u)$.

3. Interaction between these gadgets:

(a) The vertex subsets V_A, $V_{\mathcal{B}}$, and $A_{B_1}, A_{B_2}, \ldots, A_{B_\beta}$ are pairwise disjoint.

(b) It holds that $N(V_A) \cap (V_{\mathcal{B}} \cup A_{\mathcal{B}}) = \emptyset$.

(c) For each $D \subseteq V_{\mathcal{B}}$, $|D| \leq h$, the set of vertices violating the 2-anonymous property in $G - D$ is a subset of V_A.

(d) Any two vertices $u \in V_A$ and $v \notin A_{\mathcal{B}}$ satisfy $|\deg(u) - \deg(v)| > s$.

It is not hard to verify that the graph constructed in the reduction in the previous paragraph has the above properties. Before proving the correctness of the generic reduction we make the following observation.

Observation 5.3. *For each $D \subseteq V_{\mathcal{B}}$, $|D| \leq h$, the set $V_A \setminus \{v(a_i) \mid \exists v(B_j) \in D \colon a_i \in B_j\}$ is exactly the set of vertices not being 2-anonymous in $G - D$.*

Proof. By Property 1a only the vertices in V_A are not 2-anonymous in G. Property 3c ensures that the set of vertices X violating the 2-anonymous property in $G - D$ is a subset of V_A.

Because of Property 3b ($N(V_A) \cap V_{\mathcal{B}} = \emptyset$) it holds that $\deg_G(v) = \deg_{G-D}(v)$ for all $v \in X$. Moreover, because $N(A_{B_j}) \cap V_{\mathcal{B}} = \{v(B_j)\}$ (Property 2b) it holds for all $B_j \in \mathcal{B}$ and all $v(a_i, B_j) \in A_{B_j}$ that $\deg_{G-D}(v(a_i, B_j)) = \deg_G(v(a_i, B_j)) - x$ where x is one if $v(B_j) \in D$ and otherwise zero. This implies with Property 2a that $X \subseteq V_A \setminus \{v(a_i) \mid \exists v(B_j) \in D \colon a_i \in B_j\}$.

By Property 3a it follows that $V_A \subseteq V \setminus D$. To show that $V_A \setminus \{v(a_i) \mid \exists v(B_j) \in D \colon a_i \in B_j\} \subseteq X$, assume by contradiction that there is a vertex $v(a_i) \in V_A \setminus X$ but for all $v(B_j) \in D$ it holds that $a_i \notin B_j$. By Property 3b it holds that $\deg_G(v(a_i)) = \deg_{G-D}(v(a_i))$ and hence by Property 3d it follows that there is some vertex $v \in A_{\mathcal{B}}$ with $\deg_{G-D}(v) = \deg_{G-D}(v(a_i))$. Thus,

since $D \subseteq V_{\mathcal{B}}$ and $N(A_{B_j}) \cap V_{\mathcal{B}} = \{v(B_j)\}$ (Property 2b), by Property 2a it follows that there is some $v(B_j) \in D$ with $a_i \in B_j$, a contradiction. □

Lemma 5.4. *Let G be a graph satisfying Properties 1a to 3d for a given instance $(A, \mathcal{B}, h)$ of* Set Cover. *Then $(G, 2, h)$ is a yes-instance of* Anonym V-Del *if and only if $(A, \mathcal{B}, h)$ is a yes-instance of* Set Cover.

Proof. If there is an index set J, $|J| \leq h$, such that $\bigcup_{j \in J} B_j = A$, then by Observation 5.3 the set $S = \{v(B_j) \mid j \in J\} \subseteq V_{\mathcal{B}}$, $|S| = |J|$, is a k-deletion set for G. It remains to prove the reverse direction.

Let S be a k-deletion set of size at most $s = h$ for $G = (V, E)$. We form a k-deletion set S' for G such that $S' \subseteq V_{\mathcal{B}}$ and $|S'| \leq |S|$. Consider each vertex $v \in S$: If $v \in V_{\mathcal{B}}$, then add v to S' (Case 1). If $v \in N[V_A]$, then by Property 1b there is only one a_i such that $v \in N[v(a_i)]$ and we add a vertex $v(B_j) \in V_{\mathcal{B}}$ with $a_i \in B_j$ to S' (Case 2). Finally, if $v \in N[A_{\mathcal{B}}]$, then by Property 2c there is a vertex $u \in V_{\mathcal{B}}$ with $N(v) \cap A_{\mathcal{B}} \subseteq N(u)$ and we add u to S' (Case 3).

We next prove that S' is a k-deletion set for G and thus by Observation 5.3 the index set corresponding to the vertices in S' is a solution of size $|S'|$ to the Set Cover instance.

Assume towards a contradiction that $G - S'$ is not 2-anonymous. Denoting by $X \subseteq V \setminus S'$ the set of vertices not being 2-anonymous, it follows from Observation 5.3 that $X \subseteq V_A$. Moreover, by the construction of S' (see Case 2) and Observation 5.3 it follows that $N[X] \cap S = \emptyset$ and thus $\deg_{G-S}(u) = \deg_G(u)$ for all $u \in X$. Hence, for each $u \in X$ there is a vertex $w \in V$ such that $\deg_G(u) = \deg_{G-S}(w)$ and thus by Property 3d it follows that $w \in N[A_{\mathcal{B}}]$. This implies a contradiction to the construction of S' because from $w \in N[A_{\mathcal{B}}]$ it follows that S' contains w's neighbor in $V_{\mathcal{B}}$ (see Case 3) and thus $u \notin X$ by Observation 5.3. □

Using this generic reduction we now show NP-hardness on several graph classes which are defined as follows (see Brandstädt et al. [BLS99]): *Trivially perfect graphs* are the $\{P_4, C_4\}$-free graphs, that is, they do not contain an induced path or cycle on four vertices. A graph G is a *bipartite permutation graph* if G is bipartite and does not contain an asteroidal triple (is *AT-free*). Three vertices of a graph form an *asteroidal triple* if every two of them are connected by a path avoiding the neighborhood of the third. A graph is a *split graph* if it can be partitioned into a clique and an independent set.

Theorem 5.5. Anonym V-Del *is NP-complete on trivially perfect graphs, bipartite permutation graphs, and split graphs, even if* $k = 2$.

Proof. Since containment in NP is easy to see, we focus on showing NP-hardness. Let $\mathcal{B} = \{B_1, B_2, \ldots, B_\beta\}$ be a collection of subsets of some universe $A = \{a_1, a_2, \ldots, a_\alpha\}$ which form together with some $h \in \mathbb{N}$ an instance of Set Cover. As in Reduction 1, we assume without loss of generality that for each element $a \in A$ there exists a set $B \in \mathcal{B}$ with $a \in B$. Furthermore, we assume without loss of generality that each set $B \in \mathcal{B}$ occurs at least three times in $\mathcal{B}$.

We first describe the reductions for each the three graph classes and then, due to the similarities in the constructed graphs, we show for all three graphs together that the above properties are satisfied. Let $f\colon \mathbb{N} \to \mathbb{N}$ be $f(i) = i(h+1) + \alpha$.

Bipartite permutation graphs: Analogously to the reduction for trees add for each set $a_i \in A$ an *element-gadget* consisting of star $K_{1,f(i)}$ with a center vertex denoted by $v(a_i)$. Clearly, a star is a bipartite permutation graph.

For each set $B_j \in \mathcal{B}$ we add a *set-gadget* as follows: First, add a vertex $v(B_j)$ to G. For each element $a_i \in B_j$ add a child vertex, denoted by $v(a_i, B_j)$, to $v(B_j)$. Let $i_{\max} := \max_{a_i \in B_j}\{i\}$, $\ell := |B_j|$, and $A_{B_j} := \{v(a_i, B_j) \mid a_i \in B_j\}$. Next, add the vertex set $U(B_j) := \{u_1(B_j), u_2(B_j), \ldots, u_{f(i_{\max})}(B_j)\}$ and for each $a_i \in B_j$ the edge set $\{(u_r(B_j), v(a_i, B_j)) \mid 1 \le r \le f(i)\}$. Note that $\deg(v(a_i, B_j)) = \deg(v(a_i)) + 1$. Denote with $C(B_j)$ the set-gadget, that is, the connected component containing $v(B_j)$ which consists of the vertices $\{v(B_j)\} \cup A_{B_j} \cup U(B_j)$; see Figure 5.4 for an example. Furthermore, observe that $N(u_1(B_j)) \supseteq N(u_2(B_j)) \supseteq \ldots \supseteq N(u_{i_\ell}(B_j))$ and thus, in contrast to the previous reduction for trees, $C(B_j)$ is AT-free.

Overall, the constructed graph is AT-free and clearly bipartite.

Trivially perfect graphs: First, construct the graph as described above for the case of bipartite permutation graphs. Next for each $B_j \in \mathcal{B}$ apply the following changes to $C(B_j)$, see Figure 5.5 for an illustration: Add edges so that the vertices in A_{B_j} form a clique. To ensure that the degree of the vertices in A_{B_j} does not change by the previous "clique operation", remove the first $|B_j| - 1$ vertices from $U(B_j)$ which are all adjacent to each vertex in A_{B_j} due to the definition of f and $|B_j| \le \alpha$.

Clearly, the star components containing the vertices from V_A are trivially perfect. Furthermore, note that each $C(B_j)$ is trivially perfect: since $\{v(B_j)\} \cup A_{B_j}$ is a clique, the remaining vertices in $U(B_j)$ form an independent set, and since $N(u_{|B_j|}(B_j)) \supseteq N(u_{|B_j|+1}(B_j)) \supseteq \ldots \supseteq N(u_{i_\ell}(B_j))$ it is easy to verify

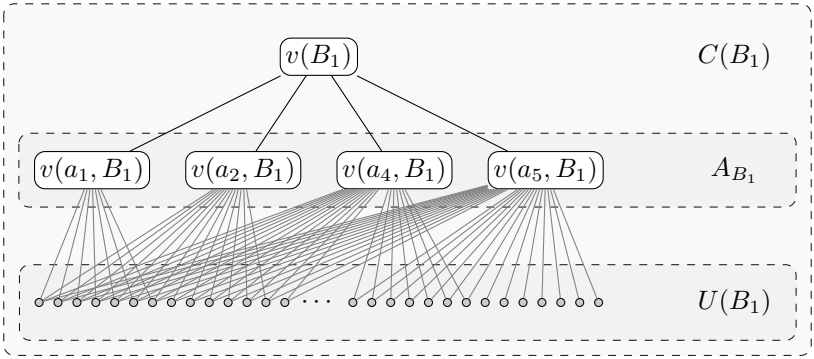


Figure 5.4.: The set-gadget $C(B_1)$ for the constructed bipartite permutation graph. The given Set Cover instance is the same as in Figure 5.3 where $B_1 = \{a, a_2, a_4, a_5\}$.

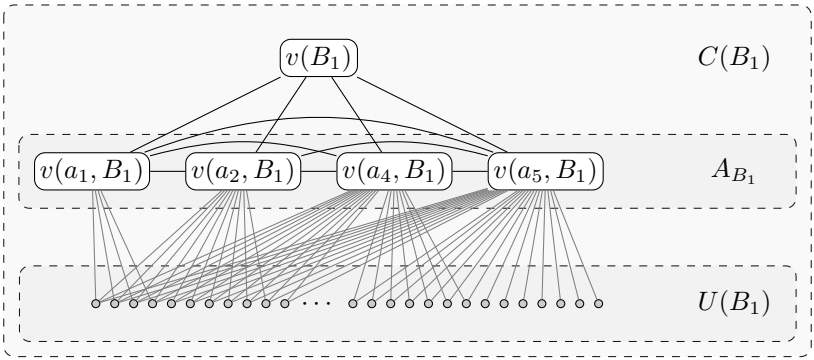


Figure 5.5.: The set-gadget $C(B_1)$ for the constructed trivially perfect graphs. The given Set Cover instance is the same as in Figure 5.3 where $B_1 = \{a, a_2, a_4, a_5\}$.

that $C(B_j)$ is indeed a threshold graph which is a special form of a trivially perfect graph [BLS99].

Note that since the connected components containing the vertices in V_A are also threshold graphs, by the reduction above we have proven that Anonym V-Del is indeed NP-hard on graphs whose connected components are threshold

graphs. However, in Theorem 5.16 we prove that Anonym V-Del is polynomial-time solvable on threshold graphs.

Split graphs: First, construct the graph G as described above for the case of bipartite permutation graphs. For each set $B_j \in \mathcal{B}$ set $W(B_j) := \{v(B_j)\} \cup U(B_j)$. Then, set $W_{\mathcal{B}} := \bigcup_{B \in \mathcal{B}} W(B)$. Finally, add edges to make the vertex subset $N(V_A) \cup W_{\mathcal{B}}$ to a clique. Observe that the remaining vertices form an independent set and, hence, the graph is a split graph.

Correctness: We now show that the constructed graphs satisfy Properties 1a to 3d. To this end, observe that, due to assumption that each set occurs three times in $\mathcal{B}$, each vertex in $C(B_j)$ is 2-anonymous. Hence, the vertices in V_A are exactly the ones that are not 2-anonymous. Thus, Property 1a is satisfied. Properties 2a and 3a are clearly satisfied. Since for each vertex $v(a_i) \in V_A$ the vertex set $N[v(a_i)]$ induces a star (a clique in the split graph case) and for each $j \neq i$ we have $N[v(a_i)] \cap N[v(a_j)] = \emptyset$, Property 1b is fulfilled. Observe that $A_{B_j} \subseteq N(v(B_j))$ for each $B_j \in \mathcal{B}$. Furthermore, the vertices in V_A and $V_{\mathcal{B}}$ are pairwise in different connected components in the case for trivially perfect graphs and bipartite permutation graphs. Thus, Properties 2b and 3b are fulfilled for these cases. For the case of split graphs, observe that we started with the construction for the bipartite permutation graphs and the vertices of V_A and $A_{\mathcal{B}}$ remained unchanged. Hence, Properties 2b and 3b are also fulfilled for the case of split graphs. In the constructed graphs for each $B_j, B_{j'} \in \mathcal{B}$, $j \neq j'$, we have $N(A_{B_j}) \cap N(A_{B_{j'}}) = \emptyset$. From this and $A_{B_j} \subseteq N(v(B_j))$, it follows that Property 2c is satisfied. Since $A_{\mathcal{B}} \subseteq N(V_{\mathcal{B}})$ this implies that Property 3c is fulfilled. Finally, since each vertex in $V \setminus (V_A \cup A_{\mathcal{B}})$ has degree at most α (at least $|N(V_A) \cup W_{\mathcal{B}}|$ in the split graph case), it follows from the definition of f that Property 3d is satisfied. □

Since Set Cover is W[2]-complete with respect to the solution size h [DF13] and the solution size s in the constructed instance was $s := h$, we have the following.

Corollary 5.6. Anonym V-Del *is W[2]-hard with respect to parameter s, even if $k = 2$ and if the input graph is a tree, a bipartite permutation graph, a split graph, or a trivially perfect graph.*

Set Cover is fixed-parameter tractable with respect to the combined parameter (α, h) [FKW04] but does not admit a polynomial kernel with respect to (α, h) [DLS14], unless NP $\subseteq$ coNP/poly. Observe that in all constructions

for Theorem 5.5 except the one for split graphs we can bound s and Δ in a polynomial in α and h.

Corollary 5.7. Anonym V-Del *on trees, bipartite permutation graphs or trivially perfect graphs does not admit a polynomial kernel with respect to the combined parameter* (k, s, Δ)*, unless NP* $\subseteq$ *coNP/poly.*

There are two natural optimization versions associated with Anonym V-Del: in one version (called Max-Anonym V-Del) the goal is to maximize the anonymity k subject to the constraint that the number s of deleted vertices does not exceed a given bound; in the other version (called Anonym Min-V-Del) the goal is to minimize the number s of deleted vertices subject to the constraint that the anonymity does not go below a certain given bound. As Set Cover is NP-hard to approximate within a ratio $o(\log n)$ [RS97, Vaz01], the above reduction yields the following inapproximability result.

Corollary 5.8. Anonym Min-V-Del *cannot be approximated within a factor of* $o(\log n)$ *in polynomial-time, even if* $k = 2$ *and if the input graph is a tree, a bipartite permutation graph, a split graph, or a trivially perfect graph, unless* $P = NP$*.*

Since the above reduction gives NP-hardness for $k = 2$ and the input graph is 1-anonymous, we immediately get inapproximability within a factor of two for Max-Anonym V-Del.

Corollary 5.9. *For every* $0 < \varepsilon < 1$*,* Max-Anonym V-Del *cannot be approximated within a factor of* $2-\varepsilon$ *in polynomial time, unless* $P = NP$*. Furthermore, if* Max-Anonym V-Del *admits for any* $0 < \varepsilon \leq 1$ *a fixed-parameter* $(2-\varepsilon)$*-approximation algorithm with respect to parameter* s*, then* $FPT = W[2]$*.*

In the next section, we show that we can strengthen these inapproximability results.

5.2.3. Inapproximability Results

Corollaries 5.8 and 5.9 give first lower bounds on the polynomial-time approximability of the two optimization problems associated to Anonym V-Del, namely Anonym Min-V-Del and Max-Anonym V-Del. For general graphs, these results, however, can be strengthened considerably in terms of the achievable approximation factor and, in case of Max-Anonym V-Del, also in terms of

the allowed running time. Specifically, we prove that Anonym Min-V-Del is not $n^{1-\varepsilon}$-approximable in polynomial time, while Max-Anonym V-Del is not $n^{1/2-\varepsilon}$-approximable in fpt-time with respect to the parameter s, even on trees.

To this end, for the polynomial-time inapproximability of Anonym Min-V-Del, we slightly adjust the reduction given in the proof of Theorem 5.1.

Theorem 5.10. *For every $0 < \varepsilon \leq 1$, Anonym Min-V-Del is not $n^{1-\varepsilon}$-approximable in polynomial time, even on graphs with maximum degree three, unless $P = NP$.*

Proof. Let $0 < \varepsilon \leq 1$ be a constant. We establish a gap-reduction with gap $n^{1-\varepsilon}$ from the Vertex Cover problem which is known to be NP-complete even in three-regular graphs [GJ79, GT1]; see page 99 for the formal definition.

Given a Vertex Cover instance $(G = (V, E), h)$ we construct an instance $I' = (G' = (V', E'), k)$ of Anonym Min-V-Del. Start by copying G into a new graph G'. Next, set $x := \lceil n^{1/\varepsilon} \rceil - n + h$. Finally, add x degree-zero vertices to G' and set $k := n - h + x$. Denote by n' the number of vertices of G', thus $n' = n + x$.

We now show that if I is a yes-instance, then $\mathrm{opt}(I') \geq h$ and if I is a no-instance, then $\mathrm{opt}(I') = n + x$.

Suppose that G contains a vertex cover S of size h. Then, deleting S in G' clearly results in an edgeless graph with $n - h + x = k$ vertices, implying that $\mathrm{opt}(I') \leq h$.

Suppose that G' contains a k-deletion set S of size at most $|V'| - 1$. Since $2k > n - h + x$ and G' contains $x > h$ degree-zero vertices, all vertices in $G' - S$ have degree zero. Furthermore, at least $k = n - h + x$ degree-zero vertices are contained in $G' - S$ and hence, $|S| \leq h$ and $S \cap V$ is a vertex cover in G. Thus, if G does not contain a vertex cover of size h, then $\mathrm{opt}(I') = |V'| = n + x$.

We obtain a gap-reduction with the gap at least

$$\frac{n+x}{h} = \frac{\lceil n^{\frac{1}{\varepsilon}} \rceil + h}{h} = \frac{(\lceil n^{\frac{1}{\varepsilon}} \rceil + h)^{(\varepsilon+1-\varepsilon)}}{h} \geq \frac{n \cdot (\lceil n^{\frac{1}{\varepsilon}} \rceil + h)^{(1-\varepsilon)}}{h}$$
$$\geq (\lceil n^{\frac{1}{\varepsilon}} \rceil + h)^{(1-\varepsilon)} = (n+x)^{(1-\varepsilon)} = (n')^{1-\varepsilon}.$$

□

Next we show strong parameterized inapproximability results for Max-Anonym V-Del. To this end, we adjust Reduction 1 in order to obtain an fpt gap-reduction.

Theorem 5.11. *If* Max-Anonym V-Del *admits on trees for any* $0 < \varepsilon \leq 1/2$ *a fixed-parameter* $n^{1/2-\varepsilon}$*-approximation algorithm with respect to parameter* s*, then* $FPT = W[2]$*.*

Proof. Let $0 < \varepsilon \leq 1/2$ be a constant. We provide an fpt gap-reduction with gap $n^{1/2-\varepsilon}$ from the W[2]-hard Set Cover problem [DF13] parameterized by the solution size h; see page 100 for the formal definition. Let $I = (A, \mathcal{B}, h)$ be an instance of Set Cover. We assume without loss of generality that for each element $a_i \in A$ there exists a set $B_j \in \mathcal{B}$ with $a_i \in B_j$. Let $f\colon \mathbb{N} \to \mathbb{N}$ be $f(i) = (h+4)i$. Set $t := \lceil (\alpha\beta)^{(1-2\varepsilon)/(2\varepsilon)} \rceil$. We will aim for making the constructed graph t-anonymous.

The instance I' of Max-Anonym V-Del is defined by $s = h$ and a graph $G = (V, E)$ constructed as follows: For each element $a_i \in A$ add a star $K_{1,f(i)}$ with the center vertex $v(a_i)$. Denote with $V_A = \{v(a_1), v(a_2), \ldots, v(a_\alpha)\}$ the set of all these center vertices. Furthermore, for each element $a_i \in A$ add t stars $K_{1,f(i)+1}$.

For each set $B_j \in \mathcal{B}$ add a set-gadget which will consist of a tree rooted in a vertex $v(B_j)$, see Figure 5.6 for an illustration. The root has $|B_j|t$ child vertices where each element $a_i \in B_j$ corresponds to exactly t of these children, denoted by $v_1(a_i, B_j), v_2(a_i, B_j), \ldots, v_t(a_i, B_j)$. Additionally, for each $\ell \in \{1, 2, \ldots, t\}$ we add to $v_\ell(a_i, B_j)$ exactly $f(i)$ degree-one neighbors. Hence, the set gadget is a tree of depth two rooted in $v(B_j)$. To ensure that the root $v(B_j)$ does not violate the t-anonymous property we add t stars $K_{1,\deg(v(B_j))}$. We denote with $V_{\mathcal{B}} = \{v(B_1), v(B_2), \ldots, v(B_\beta)\}$ the set of all root vertices. Finally, to end up with one tree instead of a forest, repeatedly add edges between any degree-one-vertices of different connected components. Denoting by n the number of vertices in G it holds that

$$n \leq \underbrace{t\beta\alpha^2}_{\text{vertices for elements}} + \underbrace{t(\beta\alpha)^2 + t\beta\alpha}_{\text{vertices for sets}} < (t\beta\alpha)^2.$$

We now show that if I is a yes-instance, then $\mathrm{opt}(I') \geq t$ and if I is a no-instance, then $\mathrm{opt}(I') = 1$.

Suppose that I has a set cover of size h. Observe that for each element $a_i \in A$ the only vertex of degree $f(i)$ is $v(a_i)$, and there are no other vertices violating the t-anonymous property. The key point in the construction is that, in order to get a t-anonymous graph, one has to delete vertices of $V_{\mathcal{B}}$. Indeed, let $a_i \in A$ be an element and $v(a_i)$ a root vertex such that $a_i \in B_j$. By construction, for each $1 \leq \ell \leq t$ the child vertex $v_\ell(a_i, B_j)$ of $v(B_j)$ has $f(i)$ child vertices and hence a degree of $f(i) + 1$. Thus, deleting $v(B_j)$ lowers the degree of

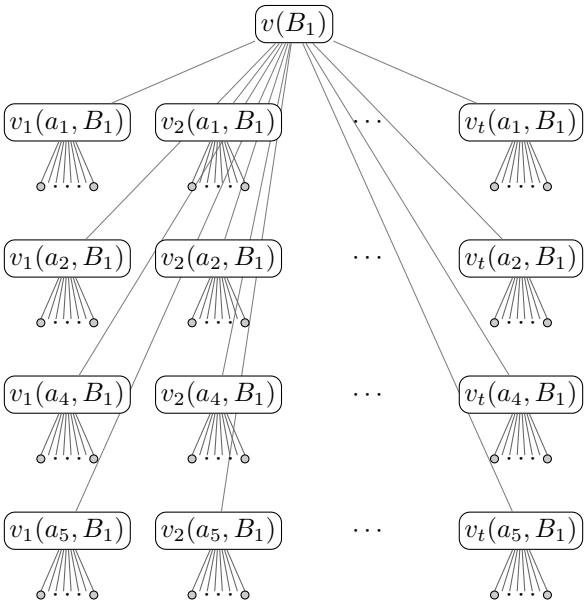


Figure 5.6.: The set-gadget for the set B_1 in the fpt gap-reduction of Theorem 5.11. The given Set Cover instance is the same as in Figure 5.3 where $B_1 = \{a, a_2, a_4, a_5\}$. The fpt gap-reduction is an extension of Reduction 1 (depicted in Figure 5.3). The main difference is that the fpt gap-reduction introduces a lot of copies of certain vertices to increase the anonymity level. This can be seen in the set-gadget above: While in Reduction 1 only one vertex corresponds to the combination (a_1, B_1) with $a_1 \in B_1$, namely $v(a_1, B_1)$, in the fpt gap-reduction t vertices correspond to the combination, namely $v_1(a_1, B_1), v_2(a_1, B_1), \ldots, v_t(a_1, B_1)$.

each $v_\ell(a_i, B_j)$ to $f(i)$ and, hence, $v(a_i)$ no longer violates the t-anonymous property. Hence, given a set cover of size h one can construct a corresponding t-deletion set for G.

Conversely, we show that if there exists a 2-deletion set of size at most h in G, then $(A, \mathcal{B}, h)$ is a yes-instance of Set Cover. Let $S \subseteq V$ be a 2-deletion set of size at most h. First, we show how to construct a 2-deletion set $S' \subseteq V_{\mathcal{B}}$ such that $|S'| \leq |S|$. To this end, initialize S' as $S' = S \cap V_{\mathcal{B}}$. If S' is a 2-deletion set, then the construction of S' is finished. Otherwise, there is a

vertex v in $G - S'$ such that there is no other vertex with the same degree as v. Observe that since $S' \subseteq V_{\mathcal{B}}$, it follows that $v \in V_A$, that is $v = v(a_i)$ for some $1 \leq i \leq \alpha$. Furthermore, observe that is exactly one vertex in G having a degree d between $f(i) - h \leq d \leq f(i)$, namely $v(a_i)$. As S is a 2-deletion set, it follows that S either contains $v(a_i)$ or a vertex u that is adjacent to a vertex w with $\deg_G(w) > \deg(v(a_i))$. In either case, we add to S' a vertex $v(B_j) \in V_{\mathcal{B}}$ such that $a_i \in B_j$. By exhaustively applying this procedure, we end up with S' being a 2-deletion set. Since the vertices in $V_{\mathcal{B}}$ are the only ones in G that are adjacent to more than one vertex of degree at least three and all vertices in V_A have degree more than three, it follows that $|S'| \leq |S|$.

It remains to show that the set $\mathcal{B}'$ of sets corresponding to the vertices in S' forms a set cover. To this end, assume by contradiction that $\mathcal{B}'$ is not a set cover, that is, there is an element $a_i \notin \bigcup_{B_j \in \mathcal{B}'} B_j$. However, this implies that in $G - S'$ there is exactly one vertex of degree $f(i)$, namely a_i, implying that S' is not a 2-deletion set, a contradiction. As $|\mathcal{B}'| = |S'| \leq |S| \leq h$, it follows that if G contains a 2-deletion set of size h, then $(A, \mathcal{B}, h)$ is a yes-instance. Hence, if $(A, \mathcal{B}, h)$ is a no-instance, then there exist no 2-deletion set of size at most h.

We obtain an fpt gap-reduction with the gap

$$\begin{aligned} t &= (t^2)^{1/2+\varepsilon-\varepsilon} = t^{2\varepsilon}(t^2)^{1/2-\varepsilon} = (\alpha\beta)^{1-2\varepsilon}(t^2)^{1/2-\varepsilon} \\ &= (\alpha^2\beta^2)^{1/2-\varepsilon}(t^2)^{1/2-\varepsilon} = (\alpha^2\beta^2 t^2)^{1/2-\varepsilon} > n^{1/2-\varepsilon} \end{aligned}$$

since $n < t^2\alpha^2\beta^2$. Thus, the statement of the theorem follows from Lemma 2.2 (see page 26). □

Since the fpt gap-reduction provided in the proof of Theorem 5.11 can be constructed in polynomial time and since Set Cover is NP-complete, we also obtain polynomial-time inapproximability under the stronger assumption P = NP.

Theorem 5.12. *For every $0 < \varepsilon \leq 1/2$,* Max-Anonym V-Del *is not $n^{1/2-\varepsilon}$-approximable in polynomial time, even on trees, unless P = NP.*

5.2.4. Polynomially-Time Solvable Cases

We complement our intractability results for Anonym V-Del from the previous sections by showing that Anonym V-Del is polynomial-time solvable on graphs with maximum degree two, on graphs that are disjoint unions of cliques, and on threshold graphs.

Graphs with Maximum Degree Two

In contrast to graphs of maximum degree three (see Theorem 5.1), we observe that Anonym V-Del is polynomial-time solvable on graphs of maximum degree two. Note that a graph of maximum degree two is just a collection of paths and cycles. Given five integers d_0, d_1, d_2, x, y, it is easy to decide whether it is possible to remove x vertices from a path of length y (respectively, from a cycle of length y) such that there survive precisely d_0 vertices of degree zero, d_1 vertices of degree one, and d_2 vertices of degree two. A straight-forward dynamic programming approach based on this observation leads to the following.

Theorem 5.13. *On graphs of maximum degree two,* Anonym V-Del *is polynomial-time solvable.*

Disjoint Union of Cliques

Note that Anonym V-Del is trivial on cliques: either the clique size is at least k, or otherwise one has to delete all the vertices. The following theorem shows that polynomial-time solvability also carries over to the case where the graph is the disjoint union of several cliques, that is, a cluster graph. Recall that a graph is a cluster graph if and only if it does not contain the 3-vertex path P_3 as an induced subgraph.

Theorem 5.14. *On a cluster graph G with maximum degree Δ,* Anonym V-Del *can be solved in $\mathcal{O}(n^2\Delta)$ time.*

Proof. Note that removing any number of vertices from a cluster graph yields another cluster graph. For an integer $c \geq 1$, we denote by $\#_{\text{comp}}(G, c)$ the number of components of size c in G. For integers $x, y \geq 1$, we denote by $G(x, y)$ the graph that consists of all components of G of size up to x, together with y new components (cliques) of size exactly x.

We design a dynamic program that solves Anonym V-Del for all such graphs $G(x, y)$. We denote by $f(x, y)$ the smallest possible number of vertices whose removal from $G(x, y)$ yields a k-anonymous graph, and we store all these values in the dynamic programming table. In the initialization phase of the dynamic program we handle the cases with $x = 1$. Note that the graph $G(1, y)$ consists of $t := \#_{\text{comp}}(G, 1) + y$ isolated vertices. Then $f(1, y) = 0$ whenever $t \geq k$, and $f(1, y) = t$ whenever $t < k$.

The cases with $x \geq 2$ are handled as follows. Consider a graph $G(x, y)$ that contains $t := \#_{\text{comp}}(G, x) + y$ components of size x. A k-anonymous subgraph of $G(x, y)$ will contain a certain number z of these components, while from each of the remaining $t - z$ components (at least) one vertex is to be removed; note that this requires $x \cdot z \geq k$ whenever $z \neq 0$. This yields the formula

$$f(x, y) \;=\; \min\{f(x-1, t-z) + t - z \;\mid\; z = 0 \text{ or } k/x \leq z \leq t\}\,.$$

As the largest clique in G contains $\Delta + 1$ vertices, the dynamic programming table has $\mathcal{O}(n\Delta)$ entries. We precompute all the values $\#_{\text{comp}}$, and then determine every value $f(x, y)$ in $\mathcal{O}(n)$ time per entry. All in all, this yields the claimed running time of $\mathcal{O}(n^2\Delta)$. The final answer for the graph G is given by $f(\Delta + 1, 0)$. □

Threshold Graphs

We recall that a graph $G = (V, E)$ is a *threshold graph* if there are positive real vertex weights $w(v)$ for $v \in V$ such that $\{v_1, v_2\} \in E$ if and only if $w(v_1) + w(v_2) \geq 1$; see Brandstädt et al. [BLS99] for more information. Without loss of generality we will assume that the vertex weights satisfy the following conditions:

- The vertex weights are pairwise distinct, and satisfy $0 < w(v) < 1$.
- Any $v_1, v_2 \in V$ satisfy $w(v_1) + w(v_2) \neq 1$; in particular $w(v_1) \neq 1/2$.

Note that the closed neighborhoods in a threshold graph are totally ordered by inclusion: whenever $w(v_1) < w(v_2)$, then $N_G[v_1] \subseteq N_G[v_2]$ and consequently $\deg(v_1) \leq \deg(v_2)$.

Lemma 5.15. *Let $U \subseteq V$ be a subset of vertices with $|U| \geq 2$, let $w_{\min} = \min_{u \in U} w(u)$ and $w_{\max} = \max_{u \in U} w(u)$, and let $u_0, u_1 \in U$ be the vertices with $w(u_0) = w_{\min}$ and $w(u_1) = w_{\max}$. All vertices in U have identical degree if and only if there is no vertex $v \in V \setminus \{u_0, u_1\}$ with $1 - w_{\max} < w(v) < 1 - w_{\min}$.*

Proof. Note that all vertices in U have identical degree if and only if $N_G[u_0] = N_G[u_1]$. The latter condition in turn holds if and only if there is no vertex v in the graph (with $v \neq u_0$ and $v \neq u_1$) that is adjacent to u_1 but not to u_0, and this is equivalent to the stated condition $1 - w_{\max} < w(v) < 1 - w_{\min}$. □

Theorem 5.16. Anonym V-Del *on threshold graphs is solvable in $\mathcal{O}(n^2)$ time.*

Proof. We provide a dynamic program to solve the problem in the claimed running time. To this end, we first need some further notation: Recall that a block $B_G(d)$ of degree d contains all degree-d vertices of G. Now consider some block $B_G(d)$ of constant degree d in an optimal subgraph for ANONYM V-DEL, and let $u_0, u_1 \in B_G(d)$ and $w_{\min}$ and $w_{\max}$ be defined as in the lemma. The *territory* of this block is defined as the union of the two closed intervals $[w_{\min}, w_{\max}]$ and $[1 - w_{\max}, 1 - w_{\min}]$; note that these two intervals will overlap if $w_{\min} < 1/2 < w_{\max}$. The *canonical superset* $U^* \subseteq V$ consists of u_0 and u_1, together with all vertices $v \in V$ that satisfy $w_{\min} \leq w(v) \leq w_{\max}$ but not $1 - w_{\max} < w(v) < 1 - w_{\min}$. One message of Lemma 5.15 is that distinct blocks in an optimal subgraph must have disjoint territories. Another message of the Lemma 5.15 is that we may as well replace every block $B_G(d)$ by its canonical superset U^*: By adding these vertices, the degree in every block either remains the same or is uniformly increased by $|U^*| - |B_G(d)|$. And if the territories of distinct blocks were disjoint before the replacement, then they will also be disjoint after the replacement. In other words, such a replacement does not violate k-anonymity but simplifies the combinatorial structure of the considered subgraph.

This suggests the following dynamic programming approach. For every real number r with $0 \leq r \leq 1/2$, we consider the threshold graph G_r that is induced by the vertices $v \in V$ with $r \leq w(v) \leq 1 - r$; note that the only crucial values for r are the $\mathcal{O}(n)$ values $w(v)$ and $1 - w(v)$ that fall between the bounds 0 and $1/2$. The goal is to compute for every graph G_r a largest k-anonymous subgraph. We start our computations with $r = 1/2$ and work downwards towards $r = 0$.

The initialization step of the dynamic program handles subgraphs that consist of a single block whose territory contains the number $1/2$. Such a block will either be empty, or it is a canonical superset specified by two values $w_{\min}$ and $w_{\max}$. All in all, this only yields a polynomial number of cases to handle. In the main computation phase of the dynamic program, we consider a general graph G_r and check all possibilities for the outermost block, which is the block whose territory is farthest away from the center point $1/2$. Since this territory is the union of two intervals $[r, q]$ and $[1 - q, 1 - r]$, we may simply check all possibilities for the interval boundary q, and then combine the corresponding block with the (previously computed) largest k-anonymous subgraph for graph G_q. Since there is only a linear number $\mathcal{O}(n)$ of candidate values for q, the largest k-anonymous subgraph of G_r can be found in linear time. □

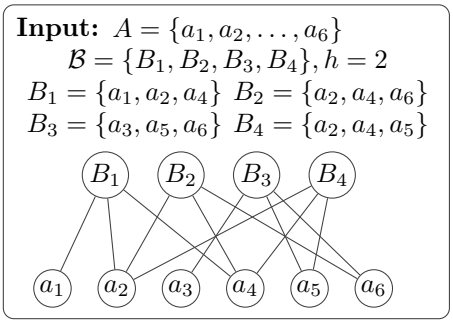
5.3. Edge Deletion

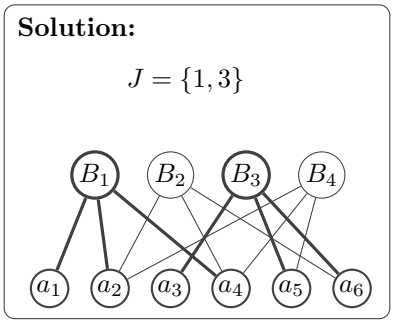
In this section, we transfer the central intractability results from Section 5.2 to the setting where instead of vertices edges are removed; see Section 5.1 for a discussion about vertex deletions versus edge deletions. To this end, we first show in Section 5.3.1 that Anonym E-Del is NP-complete on caterpillars, a subclass of trees. Compared to the NP-completeness of Anonym V-Del on trees (see Section 5.2.1) this gives a slightly stronger intractability result for Anonym E-Del. The employed reduction is, however, more complicated than the one given in Section 5.2.1 and we could not come up with a general reduction scheme as provided in the vertex deletion case in Section 5.2.2. We then provide in Section 5.3.2 polynomial-time inapproximability results for Anonym Min-E-Del and Max-Anonym E-Del for bounded-degree graphs and parameterized inapproximability results for Max-Anonym E-Del on general graphs.

5.3.1. NP-Hardness on Caterpillars

In this section, we establish a polynomial-time reduction from the NP-complete Exact Cover by 3-Sets problem, which is defined as follows:

Exact Cover by 3-Sets [GJ79, SP2]
Input: A universe $A = \{a_1, a_2, \ldots, a_{3h}\}$, a collection $\mathcal{B} = \{B_1, B_2, \ldots, B_\beta\}$ of 3-element sets over A, and $h \in \mathbb{N}$.
Question: Is there an index set $J \subseteq \{1, 2, \ldots, \beta\}$ with $|J| = h$, such that $\bigcup_{j \in J} B_j = A$?





If an Exact Cover by 3-Sets instance $I = (A, \mathcal{B}, h)$ contains such an index set J, then we refer to the set $\{B_j \mid j \in J\}$ as an *exact cover* for I.

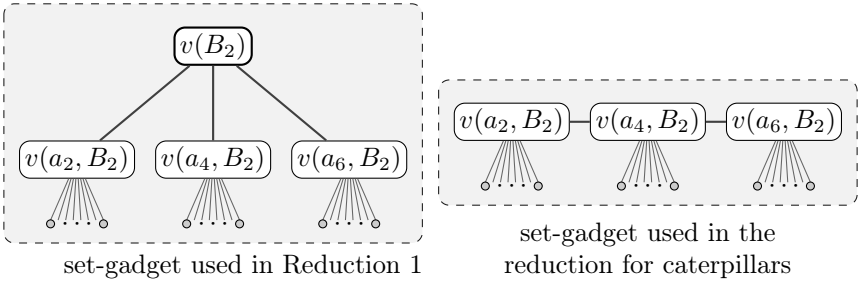


Figure 5.7.: The difference between the set-gadgets used in Reduction 1 and the reduction showing NP-hardness of Anonym E-Del on caterpillars. Deleting the vertex $v(B_2)$ corresponds to deleting the two edges $\{v(a_2, B_2), v(a_4, B_2)\}$ and $\{v(a_4, B_2), v(a_6, B_2)\}$.

The reduction in the following proof, showing that Anonym E-Del is NP-complete on caterpillars, is an adaption of the reduction provided in Section 5.2.1. A caterpillar is a tree that has a dominating path [BLS99], that is, a caterpillar is a tree such that deleting all leaves results in a path.

Theorem 5.17. Anonym E-Del *is NP-complete on caterpillars, even if* $k = 2$.

Proof. Since containment in NP is easy to see, we focus on showing NP-hardness. To this end, we provide a polynomial-time reduction from Exact Cover by 3-Sets. Let $I = (A, \mathcal{B}, h)$ be an instance of Exact Cover by 3-Sets. We assume without loss of generality that for each element $a_i \in A$ there exists a set $B_j \in \mathcal{B}$ with $a_i \in B_j$. Let $f\colon \mathbb{N} \to \mathbb{N}$ be $f(i) = (2h + 3)i$.

The instance I' of Anonym E-Del is defined on a graph $G = (V, E)$ constructed as follows. For each element $a_i \in A$ add a star $K_{1,f(i)}$ with the center vertex $v(a_i)$. Denote with $V_A = \{v(a_1), v(a_2), \ldots, v(a_{3h})\}$ the set of all these center vertices. Furthermore, for each element $a_i \in A$ add two stars $K_{1,f(i)+1}$ and two stars $K_{1,f(i)+2}$.

For each set $B_j \in \mathcal{B}$ with $B_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$ add a *set-gadget* containing the stars $K_{1,f(j_1)}$, $K_{1,f(j_2)}$, and $K_{1,f(j_3)}$. See Figure 5.7 for the difference of the set-gadget in this reduction and the reduction in Section 5.2.1. Denote with $v(a_{j_1}, B_j)$, $v(a_{j_2}, B_j)$, and $v(a_{j_3}, B_j)$ the center vertices of these stars and denote with $V_{\mathcal{B}}$ the set of all these center vertices, formally $V_{\mathcal{B}} = \{v(a_i, B_j) \mid 1 \leq i \leq 3h \wedge 1 \leq j \leq \beta \wedge a_i \in B_j\}$. Next, add the edges $\{v(a_{j_1}, B_j), v(a_{j_2}, B_j)\}$ and $\{v(a_{j_2}, B_j), v(a_{j_3}, B_j)\}$ to E. Observe that $\deg(v(a_{j_1}, B_j)) = f(j_1) + 1$,

$\deg(v(a_{j_2}, B_j)) = f(j_2) + 2$, and $\deg(v(a_{j_3}, B_j)) = f(j_3) + 1$. To end up with one caterpillar instead of a forest of caterpillars, do the following:

1. Take two different connected components (caterpillars) C_1 and C_2, let v_1 be an endpoint of a dominating path in C_1, and let v_2 be an endpoint of a dominating path in C_2, such that $\deg_G(v_1) = \deg_G(v_2) = 1$.

2. Then, add the edge $\{v_1, v_2\}$ to reduce the number of connected components by one.

3. If there exists more than one connected component, goto Step 1.

The resulting graph is clearly a caterpillar. We complete the construction of I' by setting $s = 2h$ and $k = 2$.

We now prove that I is a yes-instance of Exact Cover by 3-Sets if and only if $I' = (G, k, s)$ is a yes-instance of Anonym E-Del.

"$\Rightarrow$:" Let $\mathcal{B}' \subseteq \mathcal{B}$ be an exact cover of size h. Then we construct a 2-deletion set $S \subseteq E$ of size $2h$ as follows: For each set $B_j \in \mathcal{B}'$ with $B_j = \{a_{j_1}, a_{j_2}, a_{j_3}\}$ insert the edges $\{v(a_{j_1}, B_j), v(a_{j_2}, B_j)\}$ and $\{v(a_{j_2}, B_j), v(a_{j_3}, B_j)\}$ into S. First, observe that $|S| = 2h$. Next, we show that S is indeed a 2-deletion set. Suppose towards a contradiction that there exists a vertex $v \in V$ such that there is no further vertex of the same degree in $G - S$. Then, by construction of G, it follows that $v = v(a_i) \in V_A$ for some $i \in \{1, 2, \ldots, 3h\}$ and, by construction of S, it follows that $a_i \notin \bigcup_{B_j \in \mathcal{B}'} B_j$, a contradiction.

"$\Leftarrow$:" Let S be a 2-deletion set of edges of size at most $2h$. Observe that the only vertices in G that violate the 2-anonymous property are the vertices in V_A. Furthermore, for each $a_i \in A$ there is exactly one vertex in G with a degree d between $f(i) - 2h \leq d \leq f(i)$, namely $v(a_i)$. Since S is a 2-deletion set, it follows that for each $v(a_i) \in V_A$ there is a vertex $v \in V(S)$ having the same degree as $v(a_i)$ in $G - S$. Since $|V_A| = 3h$ and $|\deg(v(a_i)) - \deg(v(a_{i'}))| > 2h$ for all $i, i' \in \{1, 2, \ldots, 3h\}$, it follows that $|V(S)| \geq 3h$. For the further argumentation we need some notation. A vertex $v \in V$ is a *type-ℓ* vertex, $\ell \in \mathbb{N}$, if there exists a vertex $v(a_i) \in V_A$ such that $\deg_G(v) = \deg_G(v(a_i)) + \ell$. Now, observe that in G the type-1 vertices are all pairwise non-adjacent and have pairwise disjoint neighborhood sets. Thus, $V(S)$ contains at most $2h$ type-1 vertices. Furthermore, since $|V(S)| \geq 3h$, this implies that $V(S)$ contains exactly $2h$ type-1 vertices and exactly h type-2 vertices and that $|V(S)| = 3h$. Thus, for each edge in S it follows that one endpoint is a type-1 vertex and the other endpoint is a type-2 vertex. Note that the only edges fulfilling

this requirement are the ones making two vertices in $V_{\mathcal{B}}$ adjacent and, thus, $V(S) \subseteq V_{\mathcal{B}}$. Thus, each type-2 vertex of $V(S)$ is contained in some set-gadget. Denote with $\mathcal{B}'$ the set of h sets corresponding to the set-gadgets that contain the h type-2 vertices in $V(S)$. We now prove that $\mathcal{B}'$ is an exact cover. Suppose towards a contradiction that there is an element $a_i \notin \bigcup_{B_j \in \mathcal{B}'} B_j$. This implies, that no vertex $v(a_i, B_j)$ such that $j \in \{1, 2, \ldots, n\}$ and $a_i \in B_j$ is contained in $V(S)$. However, as $V(S) \subseteq V_{\mathcal{B}}$, this means that $v(a_i)$ has a unique degree in $G - S$, a contradiction to the fact that S is a 2-deletion set. Finally, since $|\mathcal{B}'| = h$, $\bigcup_{B_j \in \mathcal{B}'} B_j = A$, each set contains exactly three elements, and $|A| = 3h$, it follows that no element is covered twice. Hence, $\mathcal{B}'$ is an exact cover and, thus, I is a yes-instance. □

Note that Exact Cover by 3-Sets is fixed-parameter tractable with respect to the solution size h: There is a simple polynomial kernel which can be obtained by removing for each set all copies from the collection $\mathcal{B}$. After this deletion of the copies, the number of sets in the collection is bounded by $|\mathcal{B}| \leq |A|^3 = (3h)^3$. Hence, we cannot state an equivalent of Corollary 5.8. However, since we established NP-completeness for $k = 2$, we obtain the following equivalent of the polynomial-time inapproximability result in Corollary 5.9.

Corollary 5.18. *For every $0 < \varepsilon < 1$,* Max-Anonym V-Del *on caterpillars cannot be approximated within a factor of $2 - \varepsilon$ in polynomial time, unless $P = NP$.*

5.3.2. Inapproximability Results

As in Section 5.2.3, we can state strong inapproximability results for Anonym Min-E-Del and Max-Anonym E-Del. We remark that these inapproximability results transfer modulo the bounded-degree restriction to Anonym Min-E-Ins and Max-Anonym E-Ins, since the edge insertion variant is equivalent to the edge deletion variant in the complement graph.

Two very similar gap-reductions from Exact Cover by 3-Sets yield that Max-Anonym E-Del as well as Anonym Min-E-Del are not $n^{1-\varepsilon}$-approximable in polynomial-time on bounded degree graphs.

Theorem 5.19. *For every $0 < \varepsilon \leq 1$,* Max-Anonym E-Del *is not $n^{1-\varepsilon}$-approximable in polynomial time, even on bounded degree graphs, unless $P = NP$.*

Proof. Let $0 < \varepsilon \leq 1$ be a constant. We provide a gap-reduction with gap $n^{1-\varepsilon}$ from Exact Cover by 3-Sets which remains NP-complete even when no element occurs in more than three subsets [GJ79, SP2]. For these instances we have $h \leq \beta \leq 3h$.

Let $I = (A, \mathcal{B}, h)$ be an instance of Exact Cover by 3-Sets where no element occurs in more than three subsets. Construct an instance $I' = (G, s)$ of Max-Anonym E-Del as follows. The graph $G = (V, E)$ contains an *element-vertex* $v(a_i)$ for each element a_i from A and a *set-vertex* $v(B_j)$ for each subset B_j from $\mathcal{B}$. There is an edge in G between $v(a_i)$ and $v(B_j)$ if B_j contains a_i. For each vertex $v(B_j)$ add four degree-one vertices that are adjacent to $v(B_j)$, thus the degree of each vertex $v(B_j)$ is seven. For each vertex $v(a_i)$ add up to three degree-one vertices that are adjacent to $v(a_i)$ such that the degree of $v(a_i)$ is three (observe that each element occurs in at most three sets). Set $x := \lceil (6h17^{1-\varepsilon})^{1/\varepsilon} \rceil$. Next, add x stars $K_{1,7}$ and x stars $K_{1,4}$ to G. If the number of degree-one vertices is odd, then add one further star $K_{1,7}$ to G to ensure that the number of degree-one vertices is even. Now, add a perfect matching on the degree-one vertices to increase their degrees to two. Finally set $s := 3h$. Thus the graph G has $\beta + x$ or $\beta + x + 1$ degree-seven vertices, x degree-four vertices, $3h$ degree-three vertices, and between $4\beta + 11x$ and $4\beta + 9h + 11x + 7$ degree-two vertices. Hence, G is $3h$-anonymous. Overall, G is a graph with maximum degree seven and at most $12x + 12h + 5\beta + 7$ vertices. Observe, that $x \geq 6h \geq 2\beta$ and thus $|V| \leq 17x$.

We now show that if I is a yes-instance, then $\mathrm{opt}(I') \geq x$ and if I is a no-instance, then $\mathrm{opt}(I') \leq 6h$.

Suppose that I contains an exact cover $\mathcal{B}' \subseteq \mathcal{B}$ of size h. Then removing from G the $3h$ edges between $v(B_j) \in \mathcal{B}'$ and $v(a_i) \in A$, we obtain an x-anonymous graph G', since all vertices from the block of degree three from G are in G' in the block of degree two.

Suppose that $S \subseteq E$ is a $(6h+1)$-deletion set of size $|S| \leq s = 3h$, that is, $G - S$ is $(6h+1)$-anonymous. First, observe that $V(S)$ does not contain a vertex having degree two in G: Since $|S| \leq 3h$, at most $6h$ degree-two vertices can be contained in $V(S)$. Since $G - S$ is $(6h+1)$-anonymous and G does not contain any degree-zero or degree-one vertices, this implies that $V(S)$ does not contain any degree-two vertex. Next, observe that the only edges in G that have no degree-two vertex as endpoint are edges with one set-vertex and one element-vertex as endpoints. Since each set-vertex is, by construction, adjacent to at most three element-vertices, this implies that all set-vertices in $G - S$ have degree at least four. Furthermore, since the $3h$ element-vertices are the

only vertices in G having degree three and S is a $(6h+1)$-deletion set, this implies that $V(S)$ contains all element-vertices. Hence, $|S| = 3h$ and each element-vertex is incident to exactly one edge in S. Observe that G contains no vertex of degree five or six. Since S is a $(6h+1)$-deletion set, this implies that each set-vertex in $V(S)$ has degree four in $G - S$ and is incident to exactly three edges in S. Hence, $V(S)$ contains exactly h set-vertices and the corresponding sets form an exact cover of size h for I. Thus, if I does not contain any exact cover of size h, then there exists no $(6h+1)$-deletion set of size h for G and, hence, $\mathrm{opt}(I') \leq 6h$.

Thus we obtain a gap-reduction with the gap

$$\frac{x}{6h} = \frac{x^{\varepsilon} x^{1-\varepsilon}}{6h} = \frac{6h \cdot 17^{1-\varepsilon} \cdot x^{1-\varepsilon}}{3h} \geq (17x)^{1-\varepsilon} \geq |V|^{1-\varepsilon}.$$

□

Adjusting the gap-reduction above a little bit yields the following result.

Theorem 5.20. *For every $0 < \varepsilon \leq 1$,* Anonym Min-E-Del *is not $n^{1-\varepsilon}$-approximable in polynomial time, even on bounded degree graphs, unless $P = NP$.*

Proof. Let $0 < \varepsilon \leq 1$ be a constant. We provide a gap-reduction with gap $n^{1-\varepsilon}$ from Exact Cover by 3-Sets to Anonym Min-E-Del. This reduction is very similar to the gap-reduction provided in the proof of Theorem 5.19. Let $I = (A, \mathcal{B}, h)$ be an instance of Exact Cover by 3-Sets where no element occurs in more than three subsets. We provide an instance $I' = (G, k)$ of Anonym Min-E-Del where the graph is constructed as in the proof of Theorem 5.19 and $k := x$.

We now show that if I is a yes-instance then $\mathrm{opt}(I') = 3h$ and if I is a no-instance then $\mathrm{opt}(I') \geq x/2$.

Suppose that I contains an exact cover $\mathcal{B}' \subseteq \mathcal{B}$ of size h. Then removing from G the $3h$ edges between $v(B_j) \in \mathcal{B}'$ and $v(a_i) \in A$, we obtain a k-anonymous graph G', since all vertices from the block of degree three from G are in G' in the block of degree two.

Suppose that G has a k-deletion set S of size at most $x/2 - 1$. First, observe that $V(S)$ does not contain a vertex having degree two in G: Since $|S| \leq x/2 - 1$, at most $x - 2$ degree-two vertices can be contained in $V(S)$. Since $G - S$ is k-anonymous, $k = x$, and G does not contain any degree-zero or degree-one vertex, this implies that $V(S)$ does not contain any degree-two vertex. Next, observe that the only edges in G that have no degree-two vertex as endpoint

are edges with one set-vertex and one element-vertex as endpoints. Since each set-vertex is, by construction, adjacent to at most three element-vertices, this implies that all set-vertices in $G - S$ have degree at least four. Furthermore, since the $3h$ element-vertices are the only vertices in G having degree three and S is a k-deletion set with $k = x > 3h$, this implies that $V(S)$ contains all element-vertices. Furthermore, as G does not contain any degree-zero or degree-two vertex, it follows that each element-vertex is incident to exactly one edge in S. Observe that G contains no vertex of degree five or six. Since S is a k-deletion set of size at most $x/2 - 1$, this implies that each set-vertex in $V(S)$ has degree four in $G - S$ and is incident to exactly three edges in S. Hence, $V(S)$ contains exactly h set-vertices and the corresponding sets form an exact cover of size h for I. Thus, if I does not contain any exact cover of size h, then there exists no k-deletion set of size $x/2 - 1$ for G and, hence, $\mathrm{opt}(I') \geq x/2$.

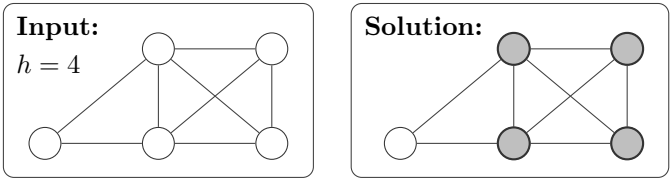
Thus we obtain a gap-reduction with the gap at least $x/(2 \cdot 3h) \geq |V|^{1-\varepsilon}$ (see the proof of Theorem 5.19 for intermediate steps in the inequality). □

Similarly to Max-Anonym V-Del, we now show strong inapproximability of Max-Anonym E-Del, even when allowing fpt-time instead of polynomial time. Note that, in contrast to the vertex deletion case in Section 5.2.3, we obtain the same inapproximability result as in the minimization variant in terms of the approximation factor. Unlike the previous reductions and the reductions in Section 5.2.3, we reduce from the W[1]-complete Clique problem, thus building on a slightly stronger assumption.

Theorem 5.21. *If* Max-Anonym E-Del *admits for any* $0 < \varepsilon \leq 1$ *a fixed-parameter* $n^{1-\varepsilon}$*-approximation algorithm with respect to parameter* s*, then* $FPT = W[1]$*.*

Proof. Let $0 < \varepsilon \leq 1$ be a constant. We provide an fpt gap-reduction with gap $n^{1-\varepsilon}$ from the W[1]-complete Clique problem [DF13] parameterized by the solution size h.

Clique [GJ79, GT19]
Input: An undirected graph $G = (V, E)$ and an integer $h \in \mathbb{N}$.
Question: Is there a subset $V' \subseteq V$ of at least h pairwise adjacent vertices?



Let $I = (G, h)$ be an instance of Clique. Assume without loss of generality that $\Delta_G + 2h + 1 \leq n$, where $n = |V|$. If this is not the case, then one can add isolated vertices to G until the bound holds.

We construct an instance $I' = (G' = (V', E'), s)$ of Max-Anonym E-Del as follows: First, copy G into G'. Then, add a vertex u and connect it to the n vertices in G'. Next, for each vertex $v \in V$ add to G' degree-one vertices that are adjacent only to v such that $\deg_{G'}(v) = n - h$. This is always possible since we assumed $\Delta_G + 2h + 1 \leq n$. Observe that in this way at most $n(n-h)$ degree-one vertices are added. Now, set $x := \lceil (4n)^{3/\varepsilon} \rceil$ and add cliques with two, $n-2h+1$, and $n-h+1$ vertices such that after adding these cliques the number of degree-d vertices in G', for each $d \in \{1, n-2h, n-h\}$, is between $x+h$ and $x+h+n$, that is, $x+h \leq |B_{G'}(d)| \leq x+h+n$; recall that $B_{G'}(d)$ is the set of vertices having degree d in G'. After inserting these cliques, the graph consists of four blocks: of degree one, $n-h$, $n-2h$, and n, where the first three blocks are roughly of the same size (between $x+h$ and $x+h+n$ vertices) and the last block of degree n contains exactly one vertex. To finish the construction, set $s := \binom{h}{2} + h$.

Now we show that if I is a yes-instance, then $\mathrm{opt}(I') \geq x$, and if I is a no-instance, then $\mathrm{opt}(I') < 2s$.

Suppose that I contains a clique $C \subseteq V$ of size h. Then, deleting the $\binom{h}{2}$ edges within C and the h edges between the vertices in C and u does not exceed the budget s and results in an x-anonymous graph G'': Since h edges incident to u are deleted, it follows that $\deg_{G''}(u) = n - h$. Furthermore, for each clique-vertex $v \in C$ also h incident edges are deleted ($h-1$ edges to other clique-vertices and the edge to u), thus it follows that $\deg_{G''}(v) = n - 2h$. Since the degrees of the remaining vertices remain unchanged, and $|B_{G'}(n-h)| \geq x+h$, it follows that each of the three blocks in G'' has size at least x. Hence, G'' is x-anonymous.

For the reverse direction, suppose that there is a $2s$-deletion set S of size at most s in G'. Since u is the only vertex in G' with degree n, and all other vertices in G' have degree at most $n-h$, it follows that S contains at least h

edges that are incident to u. Since $N_{G'}(u) = V$, it follows that the degree of at least h vertices of the block $B_{G'}(n-h)$ is decreased by one. Denote these vertices by C. Since $|S| \leq s$ and h edges incident to u are contained in S, it follows that at most $2s-h+1$ vertices are incident to an edge in S. Furthermore, since S is a $2s$-deletion set, it follows that the vertices in C have in $G'-S$ either degree one or degree $n-2h$. Thus, by deleting the at most $\binom{h}{2}$ remaining edges in S, the degree of each of the h vertices in C is decreased by at least $h-1$. Hence, these $\binom{h}{2}$ edges in S form a clique on the vertices in C and thus I is a yes-instance. Therefore, it follows that if I is a no-instance, then there is no $2s$-deletion set of size s in G'' and hence $\mathrm{opt}(I') < 2s$.

Altogether, we obtain a gap-reduction with the gap at least $x/(2s)$. Set $n' := |V'|$. By construction we have $3x \leq n' \leq n^2 + 3x + 3h + 3n + 1$. By the choice of x it follows that $x > n'/4$, since

$$\frac{n'}{4} \leq \frac{1}{4}(n^2 + 3x + 3h + 3n + 1) = x + \underbrace{\frac{1}{4}(n^2 + 3h + 3n + 1 - x)}_{<0} < x.$$

Hence the gap is

$$\frac{x}{2s} > \frac{(n')^{1-\varepsilon+\varepsilon}}{4(h^2+h)} \geq n'^{1-\varepsilon}\frac{(n')^{\varepsilon}}{8h^2} > (n')^{1-\varepsilon}\frac{x^{\varepsilon}}{8n^2} = (n')^{1-\varepsilon}\frac{(4n)^{3\varepsilon/\varepsilon}}{8n^2} > (n')^{1-\varepsilon}.$$

Thus, the statement of the theorem follows from Lemma 2.2 (see page 26). □

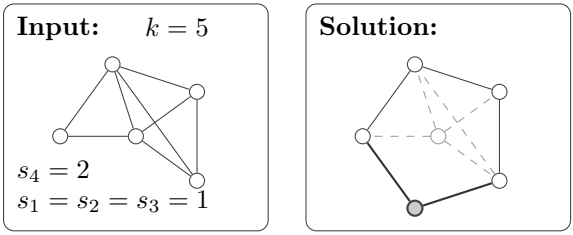
Note that the reduction above also shows that Anonym E-Del is W[1]-hard with respect to the combined parameter (s, k): It is shown that if the input graph G contains a clique of size h, then there exists an x-deletion set S of size $s = \binom{h}{2} + h$ in G'. Since $x > 2s$ it follows that S is also a $2s$-deletion set of size s. We also proved that if G' contains a $2s$-deletion set of size s, then there exists a size-h clique in G. Hence, we obtain the following: (G, h) is a yes-instance of Clique if and only if $(G', 2s, s)$ is a yes-instance of Anonym E-Del. Thus, we arrive at the following corollary.

Corollary 5.22. Anonym E-Del *is W[1]-hard with respect to the combined parameter* (s, k).

5.4. Fixed-Parameter Tractable Cases

Theorem 5.1 and Corollaries 5.6 and 5.22 show that Anonym E-Del and Anonym E-Del are fixed-parameter intractable for the each of single parameters s, k, and Δ as well as for the combined parameter (s, k). Here we show fixed-parameter tractability with respect to the combined parameter (s, Δ) for the following general problem variant where one might insert and delete specified numbers of vertices and edges.

Degree Anonymity Editing (Anonym-Edt)
Input: An undirected graph $G = (V, E)$ and five positive integers s_1, s_2, s_3, s_4, and k.
Question: Is it possible to obtain a graph $G' = (V', E')$ from G using at most s_1 vertex deletions, s_2 vertex insertions, s_3 edge deletions, and s_4 edge insertions such that G' is k-anonymous?



Observe that here we require that the inserted vertices have degree zero and we have to "pay" for making the inserted vertices adjacent to the existing ones. In particular, if $s_4 = 0$, then all inserted vertices are isolated in the target graph. Note that there are other models where the added vertices can be made adjacent to an arbitrary number of vertices [Bre+14b, Che+13a]. Our ideas, however, do not directly transfer to this variant.

For convenience, we set $s := s_1 + s_2 + s_3 + s_4$ to be the number of allowed graph modification operations.

Theorem 5.23. Anonym-Edt *is fixed-parameter tractable with respect to the combined parameter* (s, Δ).

Proof. Let $I = (G = (V, E), k, s_1, s_2, s_3, s_4)$ be an instance of Anonym-Edt. In the following we describe an algorithm finding a solution if it exists. Intuitively, the algorithm first guesses a "solution structure" and then checks whether

the graph modification operations associated to this solution structure can be performed in G. A solution structure is a graph S with at most $s(\Delta+1)$ vertices where

1. each vertex is equipped with an color from $\{0, 1, \ldots, \Delta\}$ indicating the degree of the vertex in G and
2. each edge and each vertex is marked either as "to be deleted", "to be inserted", or "not to be changed" such that
   a) all edges incident to a vertex marked as "to be inserted" are also marked as "to be inserted",
   b) at most s_1 vertices and at most s_3 edges are marked as "to be deleted", and
   c) at most s_2 vertices and at most s_4 edges are marked as "to be inserted".

The intuition behind this definition is that a solution structure S contains all graph modification operations in a solution *and* the vertices that are affected by the modification operations, that is, the vertices whose degree is changed when performing these modification operations. Observe that any solution for I defines such a solution structure with at most $s(\Delta + 1)$ vertices as each graph modification affects at most $\Delta+1$ vertices. This bound is tight in the sense that deleting a vertex v affects v and its up to Δ neighbors. Furthermore, observe that once given such a solution structure, we can check in polynomial time whether performing the marked edge/vertex insertions/deletions results in a k-anonymous graph G' since the coloring of the vertex indicates the degrees of the vertices that are affected by the graph modification operations.

Our algorithm works as follows: First it branches into all possibilities for the solution structure S. In each branch it checks whether performing the graph modification operations indicated by the marks in S indeed result in a k-anonymous graph. If yes, then the algorithm checks whether the graph modification operations associated to S can be performed in G. To this end, all edges and vertices marked as "to be inserted" are removed from S and the marks at the remaining vertices and edges are also removed and the resulting "cleaned" graph is called S'. Finally the algorithm tries to find S' as an induced subgraph of G such that the vertex degrees coincide with the vertex-coloring in S'. If the algorithm succeeds and finds S' as an induced subgraph, then the graph modification operations encoded in S can be performed which proves

that I is a yes-instance. If the algorithm fails in every branch, then, due to the exhaustive search over all possibilities for S, it follows that I is a no-instance. Thus, the algorithm is correct.

As to the running time: There are $s(\Delta + 1)$ possibilities for the number of vertices in the solution structure. Hence, there are at most $s(\Delta + 1) \cdot 2^{\binom{s(\Delta+1)}{2}} < 2^{(s(\Delta+1))^2}$ graphs with $s(\Delta + 1)$ vertices. Furthermore, there are at most $(\Delta + 1)^{s(\Delta+1)}$ possibilities to equip the vertices with colors $\{0, 1, \ldots, \Delta\}$ and $3^{s(\Delta+1)+\binom{s(\Delta+1)}{2}}$ possibilities to mark the vertices and edges. Overall, the algorithm branches into $2^{\mathcal{O}((s\Delta)^2)}$ possibilities for the solution structure S. As mentioned above, checking whether performing the graph modification operations indicated by S indeed results in a k-anonymous graph can be done in polynomial time.

Next, the algorithm checks for each S that may lead to a k-anonymous graph whether the cleaned graph S' occurs as an induced subgraph in G such that degree constraints given by the vertex coloring are fulfilled. Observe that since our input graph G has maximum degree Δ it also has a local tree-width of at most Δ [FG01]. Thus, for finding S' as induced subgraph, we can use a general result of Frick and Grohe [FG01, Theorem 1.2] showing that deciding whether a graph H of local tree-width at most ℓ satisfies a property ϕ definable in first-order logic is fixed-parameter tractable with respect to the combined parameter $(|\phi|, \ell)$. The subgraph isomorphism problem can be solved with this result on graphs with bounded local tree-width [FG01]. Thus it remains to specify the part of the formula ϕ that ensures the degree constraints. To this end, Frick and Grohe [FG01] gave as example the formula

$$x \in V \wedge \neg\exists y \exists z(\neg(y = z) \wedge (x, y) \in E \wedge (x, z) \in E)$$

to express that a vertex $x \in V$ has degree at most one. This formula can be extended to express that $x \in V$ has degree at most ℓ for some $1 \leq \ell \leq \Delta$ and the size of the formula is upper-bounded in a function of Δ. Similarly, removing the first negation symbol yields the statement that $x \in V$ has a degree of at least two (degree at least $\ell + 1$ in the extended version). Hence, we can express the degree constraints and the formula size is still bounded by a function of s and Δ (as there are up to $s(\Delta + 1)$ vertices in S'). Hence, applying the results of Frick and Grohe [FG01] shows that the overall algorithm runs in fpt-time with respect to (s, Δ). □

We remark that Theorem 5.23 is a mere classification result. We claim without proof by slightly adapting the color-coding approach used by Cai et al. [CCC06]

and Golovach [Gol14b] one can obtain a running time of $2^{(s\Delta)^{\mathcal{O}(1)}} n^{\mathcal{O}(1)}$: The idea is to randomly color the vertices in the graph with green and red. Then the subgraph $G' = (V', E')$ we are looking for is with probability $2^{(\Delta+1)|V'|}$ completely contained within the green vertices and $N_G(V \setminus V')$ are colored red. By brute-force, one can determine in $\mathcal{O}(|V'|!)$ whether a green component fits with a connected component of the sought subgraph such that the degree constraints are fulfilled. Thus, using a knapsack dynamic program over the green components, one can compute the whole subgraph G' in the claimed running time. As the running time would be still impractical, we refrain from giving a formal proof.

Next, we show that considering Anonym V-Del we can assume that $s < f(\Delta, k)$ for some function f. This implies that the above fixed-parameter tractability results transfers to the parameter (k, Δ).

Lemma 5.24. *For every yes-instance $(G = (V, E), k, s)$ of* Anonym V-Del *with Δ denoting the maximum degree of G, there is a subset $S \subseteq V$ with $|S| < 2^{\Delta+1}\Delta^3 k$ such that $G - S$ is k-anonymous.*

Proof. Let $(G = (V, E), k, s)$ be a yes-instance of Anonym V-Del and let $S \subseteq V$ be a k-deletion set. We show that if $|S| \geq 2^{\Delta}\Delta^3 2k$, then we get a smaller k-deletion set by removing a subset of k vertices from S.

Let $D = \{0, 1, \ldots, \Delta\}$ be the set of possible vertex degrees in $G - S$. We say a vertex $v \in S$ is of type (D', d') with $D' \subseteq D$ and $0 \leq d' \leq \Delta$ if $D' = \{\deg_{G-S}(v') \mid v' \in N_{G-S}(v)\}$ and $d' = \deg_{G[(V \setminus S) \cup \{v\}]}(v)$. If $|S| \geq 2^{\Delta}\Delta^3 2k$, then S contains a set S' of $\Delta^2 \cdot 2k$ vertices which are of the type (D', d') for some $D' \subseteq D$ and $0 \leq d' \leq \Delta$. Note that each vertex has at most Δ vertices in its first and at most $\Delta(\Delta - 1)$ vertices in its second neighborhood. Hence, there must be a set $S'' \subset S'$ of $2k$ independent vertices with pairwise disjoint neighborhoods. Let $S_+, S_- \subseteq S''$ be any two sets of size k each such that $S_+ \cup S_- = S''$. Consider the graphs $G_1 = G - S$ and $G_2 = G - (S \setminus S_+)$, that is, S_+ is the subset of vertices from S'' that remains in G_2 and S_- is the subset of vertices from S'' that is not in G_2.

We show that if G_1 is k-anonymous then G_2 is also k-anonymous. Every vertex from S_+ has degree d' in G_2 because S_+ is an independent set. Since $|S_+| = k$, there are at least k vertices of degree d', that is, the vertices from S_+ are k-anonymous. Every vertex v that is in G_1 and in G_2 satisfies that either $\deg_{G_2}(v) = \deg_{G_1}(v)$ or $\deg_{G_2}(v) = \deg_{G_1}(v) + 1$, because the vertices from S_+ have pairwise disjoint neighborhoods. Now, there are two

cases for $d'' = \deg_{G_1}(v)$: If $d'' \notin D'$, then $\deg_{G_2}(v) = d''$. Furthermore, there are at least as many vertices of degree d'' in G_1 as in G_2, because no vertex from S_+ is adjacent to any vertex of degree d'' in G_1. If $d'' \in D'$, then a vertex with degree d'' in G_1 may have degree $d''+1$ in G_2 because it is adjacent to some vertex in S_+. However, since the vertices from S_+ have pairwise disjoint neighborhoods, for each of the k vertices from S_+ there is at least one vertex that has degree d'' in G_1 and degree $d''+1$ in G_2 Furthermore, for each of the k vertices from S_- there is at least one vertex that has degree d'' in G_1 and G_2. In each case, there are at least k vertices with degree $\deg_{G_2}(v)$ in G_2. Thus, G_2 is k-anonymous. □

By combining Theorem 5.23 and Lemma 5.24 we obtain fixed-parameter tractability with respect to the parameter (k, Δ). For an instance (G, k, s) of Anonym V-Del simply run the algorithm from Theorem 5.23 on the instance $(G, k, \min\{s, 2^{\Delta}\Delta^3 2k\})$.

The ideas behind Lemma 5.24 can be easily transferred to the edge deletion variant.

Lemma 5.25. *For every yes-instance $(G = (V, E), k, s)$ of* Anonym E-Del *with Δ denoting the maximum degree of G there is a subset $S \subseteq E$ with $|S| < 2\Delta^3 2k$ such that $G - S$ is k-anonymous.*

Proof. Let $(G = (V, E), k, s)$ be a yes-instance of Anonym E-Del and let $S \subseteq E$ be a k-deletion set. We show that if $|S| \geq 2\Delta^3 2k$, then we get a smaller k-deletion set by removing a subset of k edges from S.

We say an edge $e = \{u, v\} \in S$ is of type (d_1, d_2) with $1 \leq d_1, d_2 \leq \Delta$ if $d_1 = \deg_{G-S}(u)$ and $d_2 = \deg_{G-S}(v)$. If $|S| \geq 2\Delta^3 2k$, then S contains a set S' of $2\Delta \cdot 2k$ edges which are of the type (d_1, d_2) for some $0 \leq d_1, d_2 \leq \Delta$. Since each vertex has, by definition of Δ, at most Δ neighbors, there must be a set $S'' \subset S'$ of $2k$ pairwise disjoint edges. Let $S_+ \subseteq S''$ be a set of size k. Now, similarly to proof of Lemma 5.24, it follows that $G - (S \setminus S_+)$ is also k-anonymous as it contains at least k vertices of degree d_1, $d_1 + 1$, d_2, and $d_2 + 1$, respectively and the other vertices remain untouched. □

By combining Theorem 5.23 and Lemma 5.25 we also obtain fixed-parameter tractability for Anonym E-Del with respect to the parameter (k, Δ). Thus, we arrive at the following classification result.

Corollary 5.26. Anonym V-Del *and* Anonym E-Del *are fixed-parameter tractable with respect to the combined parameter (k, Δ).*

5.5. Conclusion

In this chapter, we provided a thorough overview on the computational complexity of the Degree Anonymity problem when considering vertex or edge deletions. We obtained various hardness results from the viewpoints of approximation and parameterized complexity, even in restricted graph classes. Besides this large amount of hardness results we obtained a few positive results (polynomial-time solvable cases) on highly structured graph classes.

Despite this in terms of algorithmic tractability discouraging picture of the computational complexity, some open questions remain that still raise hope for broader positive results. In particular, these questions are:

1. Are Anonym Min-E-Del or Anonym Min-V-Del constant-factor approximable in polynomial time when k is a constant?

2. Are the two optimization variants of Anonym E-Edt constant-factor approximable in polynomial time?

3. What is the complexity of Anonym V-Del on unit interval graphs?

4. Do all NP-completeness results of Anonym V-Del on special graph classes (see Section 5.2.2) transfer to Anonym E-Del?

Despite serious efforts, we failed to extend the polynomial-time inapproximability results for Anonym Min-E-Del and Anonym Min-V-Del to exclude approximation algorithms running in fpt-time with respect to the parameter k. The reason is that all our gap-reductions relied on k being in the order of n. This restriction made it easy to control the possibilities for the solutions in the constructed graph, but leaves Question 1 as challenge for future research. Question 2 seems to be closely related to Question 1 as we failed to answer both questions for the same reason: The variant of editing edges allows to "repair" a suboptimal decisions by reverting the degree of a vertex with one further operation (edge deletion or insertion). In the case of edge deletions with constant values of k it might be possible to "repair" suboptimal decisions by decreasing the degrees of just a few other vertices. We found no way of dealing even with one of these two possibilities to repair suboptimal decisions. As to Question 4, our findings so far support the conjecture that the hardness results mostly transfer, but the reductions to prove this will become messy.

Chapter 6.

Degree Anonymity by Edge Insertion

In this chapter, we study the Anonym E-Ins problem. It asks whether a given undirected graph can be made k-anonymous by at most s edge insertions. Contrasting the numerous intractability results that we obtained in Chapter 5 for the edge deletion and vertex deletion variants of Anonym E-Ins, we present a polynomial-size problem kernel with respect to the parameter maximum degree. The kernelization result relies on a heuristic two-phase approach due to Liu and Terzi [LT08]. Generalizing the ideas of this approach, we show that a class of graph completion problems, where the goal is to obtain a graph whose degree sequence satisfies some prescribed property, is fixed-parameter tractable with respect to the maximum degree. Finally, we also implemented the two-phase approach and optimized it for large-scale social networks providing upper and lower bounds for an optimal solution. Experiments on several real-world datasets show that our implementation significantly improves on known heuristics and provides (provably) optimal solutions on about 21 % of the real-world data.

6.1. Introduction

In a landmark paper, Liu and Terzi [LT08] (also see Clarkson et al. [CLT10] for an extended version) introduced the following simple graph-theoretic model for identity anonymization on (social) networks. Herein, they transferred the k-anonymity concept known for tabular data in databases [Fun+10, Sam01, SS98, Swe02] to graphs; see Figure 6.1 for examples and Chapter 5 for the vertex and edge deletion variants.

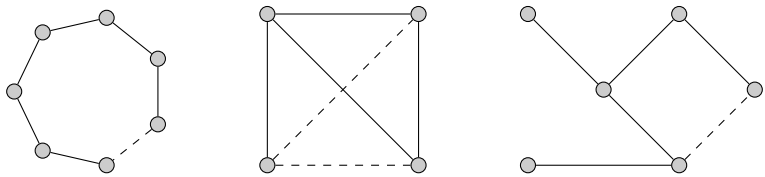
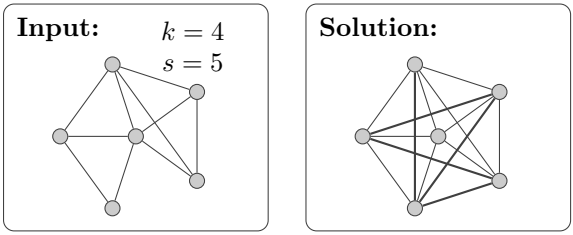


Figure 6.1.: Three illustrating examples. The solid edges indicate the original graphs. Adding the dashed edges changes the graphs (from left to right) from being 2-anonymous to 7-anonymous, from 1-anonymous to 4-anonymous, and from 1-anonymous to 2-anonymous.

Degree Anonymity by Edge Insertion (Anonym E-Ins)
Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Question: Is there an edge set $S \subseteq \binom{V}{2} \setminus E$ of size at most s such that $G + S$ is k-anonymous?



We refer to Section 1.2 for a thorough discussion about the model and related work. Notably, there is a close connection between Anonym E-Ins and Anonym E-Del—the edge deletion version of Anonym E-Ins: To see this, let us recall that for a given graph $G = (V, E)$ an edge set $S \subseteq \binom{V}{2} \setminus E$ is called a *k-insertion set* if $G + S$ is k-anonymous. Furthermore, an edge subset $S \subseteq E$ is called *k-deletion set* if $G - S$ is k-anonymous. Now, observe that a graph G is k-anonymous if and only if the complement graph $\overline{G}$ is k-anonymous. Thus, in Anonym E-Ins the task is to find a small k-insertion set while in Anonym E-Del the task is to find a small k-deletion set. The connection between these two problems is now as follows: any k-insertion set for G is a k-deletion set for the complement graph $\overline{G}$. Thus, most of the intractability results that we obtained in Chapter 5 for Anonym E-Del transfer to Anonym E-Ins. The NP-hardness on bounded-degree graphs (see Theorem 5.19), however, does *not* transfer to Anonym E-Ins.

Table 6.1.: Overview on the computational complexity classification of Anonym E-Ins.

Parameter	Anonym E-Ins
k	NP-complete for $k = 2$ (Theorem 6.2 on page 136)
s	W[1]-hard even for $k = 2$ (Theorem 6.2 on page 136)
Δ	FPT (Corollary 6.18 on page 156) and admits a polynomial problem kernel (Theorem 6.15 on page 155)

Our contributions. Our central theoretical result is to show that Anonym E-Ins has a polynomial-size problem kernel when parameterized by the maximum vertex degree Δ of the input graph. In other words, we prove that there is a polynomial-time algorithm that transforms any input instance of Anonym E-Ins into an equivalent instance with at most $\mathcal{O}(\Delta^7)$ vertices. Indeed, we encounter a "win-win" situation when proving this result: We show that Liu and Terzi's heuristic strategy [LT08] finds an optimal solution when the size s of a minimum solution is larger than $2\Delta^4$. As a consequence, we can bound s in $\mathcal{O}(\Delta^4)$ and, hence, a polynomial kernel we provide for the combined parameter (Δ, s) actually is also a polynomial kernel only for Δ. We then generalize the ideas behind the kernelization to further degree sequence completion problems where the task is to insert edges so that the degree sequence of the resulting graph fulfills some prescribed property Π.

While our kernelization directly implies fixed-parameter tractability for Anonym E-Ins parameterized by Δ, we also develop a further fixed-parameter algorithm with an improved worst-case running time. In addition, we prove that Anonym E-Ins becomes NP-complete on graphs with H-index three. The same proof also yields NP-completeness in three-colorable graphs. Further, adopting the viewpoint of "standard parameterization", we show that Anonym E-Ins is W[1]-hard when parameterized by the solution size s (the number of inserted edges), even when $k = 2$. In other words, there is no hope for fixed-parameter tractability even when the anonymity level k is low and the graph needs only few edge insertions (meaning little perturbation) to achieve k-anonymity, see Table 6.1 for an overview.

Finally, we extend Liu and Terzi's heuristic strategy [LT08]. To this end, let us briefly describe their two-phase approach:

(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.

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 size. Notably, on about 21 % of the real-world data we provide (provably) optimal solutions, whereas in the other cases our upper bounds significantly improve on a recent heuristic due to Lu et al. [LSB12].

Organization. We start by giving computational hardness results in Section 6.2. Then we present in Section 6.3 our theoretical main result of this chapter: the polynomial problem kernel with respect to the parameter maximum degree. In Section 6.4, we complement the kernel with a direct fixed-parameter algorithm. Then we generalize the ideas from Section 6.5 to a wide class of graph completion problems. In Section 6.6 we describe how we implemented upper- and lower-bound heuristics that are based on the ideas behind the kernelization result. Finally, we experimentally evaluate our implementation in Section 6.6.5.

6.2. Computational Hardness

In this section, we study the computational complexity of Anonym E-Ins. Due to the close connection between Anonym E-Del and Anonym E-Ins, it follows from our work in Section 5.3, that Anonym E-Ins

- is NP-complete, even if $k = 2$ (see Theorem 5.17),
- cannot be approximated within a factor of $n^{1-\varepsilon}$ neither in polynomial time (see Theorems 5.19 and 5.20), unless NP $=$ P, nor in $f(s) \cdot n^{\mathcal{O}(1)}$ time for any computable f (see Theorem 5.21), unless W[1] $=$ FPT, and
- is W[1]-hard with respect to the combined parameter (s, k).

In this section, we provide two polynomial-time many-to-one reductions yielding three additional (parameterized) hardness results: Anonym E-Ins is

- NP-complete on three-colorable graphs,

- NP-complete on graphs with H-index three (that is on graphs with at most three vertices of degree more than three), and
- W[1]-hard with respect to s even if $k = 2$.

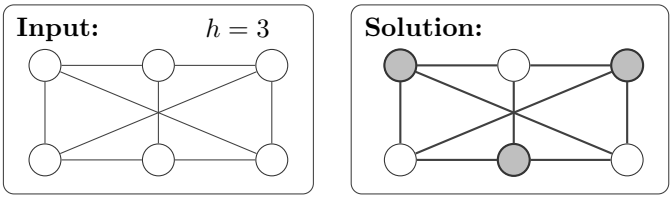
Notably, only the last of these three results transfers back to Anonym E-Del as the complement graph operation "destroys" three-colorability as well as the H-index of a graph.

As we will show in Section 6.3, Anonym E-Ins is fixed-parameter tractable with respect to the maximum degree, showing that a small maximum degree makes the problem easy. Interestingly, the reduction given in the next proof contains exactly one vertex with degree more than three, showing that one high-degree vertex is sufficient to make the problem hard.

Theorem 6.1. Anonym E-Ins *is NP-complete on three-colorable graphs and on graphs with H-index three.*

Proof. As the containment in NP is obvious, we focus on showing the NP-hardness. To this end, we give a reduction from the NP-complete Independent Set problem.

Independent Set [GJ79, GT20]
Input: An undirected graph $G = (V, E)$ and a positive integer h.
Question: Is there an independent set $V' \subseteq V$ of size $|V'| \geq h$, that is, a vertex subset of pairwise nonadjacent vertices?



We assume without loss of generality that in the given Independent Set instance (G, h) it holds that $|V| \geq 2h + 1$. We construct an equivalent instance $(G' = (V', E'), k, s)$ for Anonym E-Ins as follows. We start with a copy G' of G, denoting with $v' \in V'$ the copy of the vertex $v \in V$. Then, for each vertex $v \in V$ we insert in G' degree-one vertices adjacent to v' such that v' has degree Δ_G in G'. Finally, we insert a star with $\Delta_G + h - 1$ leaves and denote its central vertex c. We conclude the construction by setting $k := h + 1$ and $s := \binom{h}{2}$.

We prove the correctness of the reduction by showing that (G, h) is a yes-instance of Independent Set if and only if (G', k, s) is a yes-instance of Anonym E-Ins.

"$\Rightarrow$:" Let $I \subseteq V$ be an independent set in G with $|I| = h$. We show that the edge set $\binom{I}{2}$ is a solution for (G', k, s): Since I is an independent set, none of the edges in $\binom{I}{2}$ is contained in G'. Furthermore, observe that $G' + \binom{I}{2}$ is k-anonymous: There are three different degrees in the degree distribution of G': 1, Δ_G, and $\Delta_G + h - 1$. By construction, there are at least k degree-one and $|V| - |I| \geq k$ degree-Δ_G vertices in $G' + \binom{I}{2}$. Furthermore, the vertices with degree $\Delta_G + h - 1$ in $G' + \binom{I}{2}$ are all in $I \cup \{c\}$. Thus, there are $|I| + 1 = k$ vertices with degree $\Delta_G + h - 1$. Finally, observe that $|\binom{I}{2}| = \binom{h}{2}$.

"$\Leftarrow$:" Let $E_s \subseteq \binom{V'}{2}$ be a solution to (G', k, s) with $|E_s| = s$. The following degrees occur in G': 1, Δ_G, and $\Delta_G + h - 1$. Furthermore, observe that there is exactly one vertex with degree $\Delta_G + h - 1$ in G'. In $G' + E_s$ there must be at least $k - 1 = h$ further vertices of degree at least $\Delta_G + h - 1$ and, hence, each of them has to have at least $h - 1$ incident edges in E_s. Since $s = \binom{h}{2}$, there are exactly h such vertices, each incident to exactly $h - 1$ edges in E_s. These h vertices form an independent set of size h in G' and, by construction, the h corresponding vertices form an independent set of size h in G. This completes the proof of the correctness of the reduction.

Independent Set is NP-complete on three-colorable graphs [PW96, Lemma 6] and on graphs with maximum degree three [GJ79, GT20]. Clearly, if G is three-colorable, then G' is three-colorable as well. Furthermore, if G has maximum degree three, then only the central vertex c has degree larger than three, implying that the H-index of G' is three. □

The NP-completeness for constant H-index directly implies that Anonym E-Ins remains NP-complete even if the prominent parameters average degree and degeneracy[1] are constant. We next prove W[1]-hardness for the "standard parameterization", that is, the number of edges s that may be inserted.

Theorem 6.2. Anonym E-Ins *is W[1]-hard parameterized by the number s of inserted edges, even if $k = 2$.*

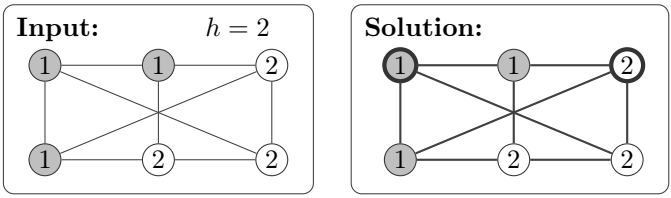
Proof. We give a parameterized reduction from the Multicolored Independent Set problem.

[1] An undirected graph G has degeneracy d if every subgraph of G (including G) contains a vertex of degree at most d.

Multicolored Independent Set

Input: An undirected graph $G = (V, E)$, an integer h, and a vertex coloring $\mathrm{col}\colon V \to \{1, 2, \ldots, h\}$.

Question: Is there a multicolored independent set $V' \subseteq V$ of size $|V'| = h$, that is, for every pair of vertices $u, v \in V'$ it holds that $\mathrm{col}(u) \neq \mathrm{col}(v)$ and $\{u, v\} \notin E$?



The W[1]-hardness of Multicolored Independent Set directly follows from the W[1]-hardness of the Multicolored Clique problem [Fel+09]. We assume without loss of generality that each color class contains at least three vertices.

Given a Multicolored Independent Set instance (G, h, col), we construct an equivalent instance $(G' = (V', E'), k, s)$ for Anonym E-Ins as follows. We start with copying the graph G to G'. Then, for each vertex $v \in V$ we insert in G' degree-one vertices adjacent to v until v has degree $h^3 \cdot \mathrm{col}(v) + \Delta_G$ in G'. Next we insert h disjoint stars to G'—one for each color in $\{1, 2, \ldots, h\}$. The star for color i has $h^3 i + \Delta_G + h - 1$ leaves and its central vertex is denoted by w_i. We conclude the construction by setting $k := 2$ and $s := \binom{h}{2}$.

We prove the correctness of the reduction by showing that (G, h, col) is a yes-instance of Multicolored Independent Set if and only if $(G', 2, s)$ is a yes-instance of Anonym E-Ins.

"$\Rightarrow$:" Let $I \subseteq V$ be a multicolored independent set in G with $|I| = h$. It is easy to verify that $E_s = \binom{I}{2}$ is a k-insertion set for G' of size $|E_s| = s = \binom{h}{2}$.

"$\Leftarrow$:" Let S be a k-insertion set for G' with $|S| \leq s$. Recall that a block $B_G(d)$ of degree d is the set of all vertices with degree d in G, formally, $B_G(d) := \{v \in V \mid \deg_G(v) = d\}$. Observe that G' contains h blocks $B_{G'}(h^3 i + \Delta_G + h - 1)$ for $i \in \{1, 2, \ldots, h\}$ of size exactly one. Since $k = 2$, this implies that $|V(S)| \geq h$. Since for $i \in \{1, 2, \ldots, h\}$ there is no vertex in G' with degree $h^3 i + \Delta_G + h - 1 + j$ for any $j \in \{1, 2, \ldots, h^2\}$ and $s = \binom{h}{2} < h^2$, it follows that in order to get a vertex of the same degree as w_i, the set S must increase the degree of at least one vertex by at least $h - 1$. It follows that such a vertex must have degree $h^3 i + \Delta_G$ in G', there is one such vertex for each $i \in \{1, 2, \ldots, h\}$ in $V(S)$, and each

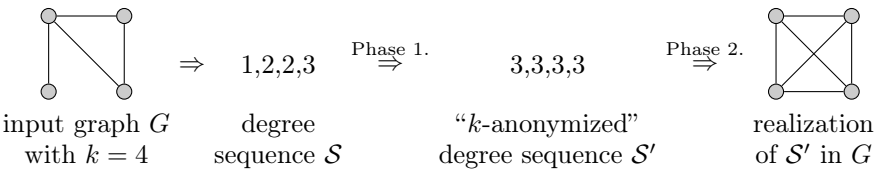


Figure 6.2.: Example for the basic steps in the heuristic of Liu and Terzi [LT08]. Phase 1. Anonymize the degree sequence (ignoring the graph), that is, increase its numbers such that each resulting number occurs at least k times. Phase 2. Realize the anonymized degree sequence as super-graph of G.

of them is incident to exactly $h - 1$ edges of S. Since $S \leq \binom{h}{2}$, this implies that $|V(S)| = h$ and that $V(S)$ is an independent set in G'. By construction, it follows that $V(S)$ is also a multicolored independent in G. □

6.3. Polynomial Kernel for the Parameter Maximum Degree

In this section, we provide our main theoretical result of this chapter: a polynomial kernel with respect to the parameter maximum degree Δ (Theorem 6.16). To this end, we first analyze the heuristic Liu and Terzi [LT08] proposed to solve Anonym E-Ins. Basically, this heuristic runs in two phases as follows (see Figure 6.2 for an example and Section 6.3.2 for the technical details):

1. k-anonymize the degree sequence.

2. Realize the k-anonymized degree sequence as a super-graph of G.

The heuristic may fail to find a solution if the anonymized degree sequence computed in Phase 1 cannot be realized in Phase 2. However, in Section 6.3.2 we show that if there is a "large" difference between the degree sequence and the anonymized degree sequence, then there is always a realization of the anonymized degree sequence. This leads, as the heuristic runs in polynomial time, to the following win-win situation: For a given instance of Anonym E-Ins, one can either find a k-insertion set in polynomial time using the above approach, or the solution—if it exists—is "small" (containing less than $(\Delta^2 + 4\Delta + 3)^2$ edges). This win-win situation enables us to show that a polynomial kernel with respect

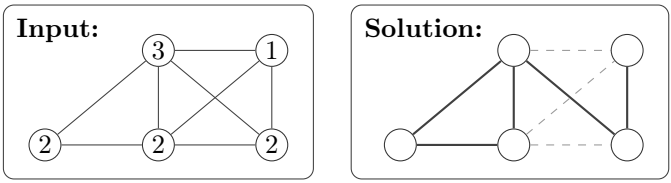
to the combined parameter (Δ, s) provided in Section 6.3.3 is indeed polynomial also in Δ.

We begin, however, with presenting the main technical tool used in this section, the so-called f-Factor problem.

6.3.1. The f-Factor Problem

Anonym E-Ins has a close connection to the polynomial-time solvable f-Factor problem:

f-Factor [LP86, Chapter 10]
Input: An undirected graph $G = (V, E)$ and a function $f\colon V \to \mathbb{N}_0$.
Question: Is there an *f-factor*, that is, a subgraph $G' = (V, E')$ of G such that $\deg_{G'}(v) = f(v)$ for all $v \in V$?



The f-Factor problem can be solved in $\mathcal{O}(\sqrt{\sum_{v \in V} f(v)}|E|)$ time [Gab83], see Section 3.2 for an overview on f-factors. Using f-Factor, one can reformulate Anonym E-Ins as follows: Given an instance (G, k, s), the question is whether there is a function $f\colon V \to \mathbb{N}_0$ such that the complement graph $\overline{G}$ contains an f-factor, $\sum_{v \in V} f(v) \leq 2s$ (every edge is counted twice in the sum of degrees), and for all $v \in V$ it holds that $|\{u \in V \mid \deg_G(u) + f(u) = \deg_G(v) + f(v)\}| \geq k$ (the k-anonymity requirement). As a warm-up, we use this formulation to make the following observation.

Observation 6.3. *If $k > n/2$, then* Anonym E-Ins *can be solved in $\mathcal{O}(n^4)$ time.*

Proof. Observe that if $k > n/2$, then all vertices in the k-anonymous graph have the same degree. Our polynomial-time algorithm is as follows: Branch in the at most n possibilities for the degree $d \geq \Delta$ in the k-anonymous graph. Then compute for each $v \in V$ the value $f(v) = d - \deg_G(v)$. If $1/2 \cdot \sum_{v \in V} f(v) > s$, then return no. Otherwise determine whether there is an f-factor in $\overline{G}$. If there is an f-factor, then return yes (and return the set of edges in the f-factor as solution set), otherwise return no.

As to the running time, observe that we solve at most n f-FACTOR instances. Each instance can be solved in $\mathcal{O}(\sqrt{\sum_{v \in V} f(v)}|E|) = \mathcal{O}(n^3)$ time. Summing up, the running time is bounded by $\mathcal{O}(n^4)$. □

In the above reformulation of ANONYM E-INS one looks for an f-factor in the complement graph. Phase 2 in Liu and Terzi's heuristic [LT08] (see Figure 6.2) can also be formulated as an f-FACTOR problem in the complement graph: Realizing the k-anonymized degree sequence as super-graph of G is equivalent to finding an f-factor in $\overline{G}$, where $f(v)$ captures the difference between the degree of v in G and the corresponding number in the k-anonymized degree sequence.

As mentioned in the introduction of Section 6.3, we prove that under certain conditions there exists a realization of the anonymized degree sequence (Phase 2). These conditions come from the following lemma guaranteeing the existence of an f-factor (see also Section 3.2.1).

Lemma 3.8 (Katerinis and Tsikopoulos [KT00])**.** *Let $G = (V, E)$ be an undirected graph with minimum vertex degree δ and let $a \leq b$ be two positive integers. Suppose further that*

$$\delta \geq \frac{b}{a+b}|V| \text{ and } |V| > \frac{a+b}{a}(a+b-3).$$

Then, for any function $f\colon V \to \{a, a+1, ..., b\}$ where $\sum_{v \in V} f(v)$ is even, G has an f-factor.

As we are interested in an f-factor in the complement graph of our input graph G, we use Lemma 3.8 with minimum degree $\delta \geq n - \Delta - 1$, $a = 1$, and $b = \Delta + 2$. Using the next corollary, we will later show that for a minimal k-insertion set S with $|V(S)| > \Delta^2 + 4\Delta + 3$, the maximum degree in $G + S$ is at most $\Delta + 2$ (Lemma 6.5). This is the reason for setting b to $\Delta + 2$.

Corollary 6.4. *Let $G = (V, E)$ be an undirected graph with n vertices, minimum degree $n - \Delta - 1$, $\Delta \geq 1$, and let $f\colon V \to \{1, 2, \ldots, \Delta + 2\}$ be a function such that $\sum_{v \in V} f(v)$ is even. If $n \geq \Delta^2 + 4\Delta + 3$, then G has an f-factor.*

Proof. Set $a := 1$ and $b := \Delta + 2$. Since $n \geq \Delta^2 + 4\Delta + 3$ it follows that:

$$\frac{b}{a+b}n = \frac{\Delta+2}{\Delta+3}n \leq n - \Delta - 1.$$

Furthermore,

$$\frac{a+b}{a}(b+a-3) = (\Delta+3)\Delta = \Delta^2 + 3\Delta < \Delta^2 + 4\Delta + 3 = n$$

and, thus, all conditions of Lemma 3.8 are fulfilled. □

6.3.2. A Polynomial-Time Algorithm for "Large"-Solution Instances

In this section, we give an algorithm based on the approach of Liu and Terzi [LT08] (see Figure 6.2) that, if a minimum-size k-insertion set S is "large" compared to Δ, solves the given instance in polynomial time (Lemma 6.9). The key point is to prove that in Phase 2 there exists a realization of the anonymized degree sequence, that is, the corresponding f-factor in the complement graph exists (see previous section). To this end, we use Corollary 6.4 and therefore have to ensure that its conditions are fulfilled, namely:

1. The maximum function value is $\Delta + 2$.
2. There are at least $\Delta^2 + 4\Delta + 3$ "affected" vertices, that is, vertices $v \in V$ such that $f(v) > 0$.

In the next lemma we show that a "large" minimum-size k-insertion set increases the maximum degree by at most two implying the first condition. This further implies that if a minimum-size k-insertion set contains more than $(\Delta^2+4\Delta+3)^2$ edges, also the second condition is satisfied.

Lemma 6.5. *Let $G = (V, E)$ be an undirected graph and let S be a minimum-size k-insertion set. If $|V(S)| \geq \Delta_G^2 + 4\Delta_G + 3$, then the maximum degree in $G + S$ is at most $\Delta_G + 2$.*

Proof. Let G be an undirected graph with maximum degree Δ_G and k be an integer. Let S be a minimum-size edge set such that $G + S$ is k-anonymous and suppose that $|V(S)| \geq \Delta^2 + 4\Delta + 3$. Now assume towards a contradiction that the maximum degree in $G + S$ is at least $\Delta_G + 3$. We show that there exists an edge set S' such that $G + S'$ is k-anonymous, $|S'| < |S|$, and $G + S'$ has maximum degree at most $\Delta_G + 2$, contradicting the minimality of S.

First we introduce some notation. Let f be a function $f\colon V \to \mathbb{N}_0$ defined as $f(v) := \deg_{G+S}(v) - \deg_G(v)$ for all $v \in V$. Furthermore, denote with X the set of all vertices having degree more than $\Delta_G + 2$ in $G + S$, that is,

$$X := \{v \in V \mid f(v) + \deg_G(v) \geq \Delta_G + 3\}.$$

Observe that (V, S) is an f-factor of the complement graph $\overline{G}$ and $2|S| = \sum_{v \in V} f(v)$. We now define a new function $f' \colon V \to \mathbb{N}_0$ such that $\overline{G}$ contains an f'-factor denoted by $G' = (V, S')$ such that $G + S'$ is k-anonymous, $|S'| < |S|$, and $G + S'$ has maximum degree at most $\Delta_G + 2$.

We define f' for all $v \in V$ as follows:

$$f'(v) := \begin{cases} f(v) & \text{if } v \notin X, \\ \Delta_G - \deg_G(v) + 1 & \text{if } v \in X \text{ and } f(v) + \deg_G(v) - \Delta_G - 1 \text{ is even,} \\ \Delta_G - \deg_G(v) + 2 & \text{otherwise.} \end{cases}$$

First observe that $\deg_G(v) + f'(v) \leq \Delta_G + 2$ for all $v \in V$. Furthermore, observe that $f'(v) = f(v)$ for all $v \in V \setminus X$ and for all $v \in X$ it holds that $f'(v) < f(v)$ and $f(v) - f'(v)$ is even. Thus, $\sum_{v \in V} f(v) > \sum_{v \in V} f'(v)$ and $\sum_{v \in V} f'(v)$ is even. It remains to show that

(i) $\overline{G}$ contains an f'-factor $G' = (V, S')$ and

(ii) $G + S'$ is k-anonymous.

To prove (i) let $\widetilde{V} := \{v \in V \mid f'(v) > 0\}$. Next, observe that from the definition of X and f' it follows $f(v) > 0$ if and only if $f'(v) > 0$ and hence $\widetilde{V} = V(S)$. Furthermore, let $\widetilde{G} := \overline{G[\widetilde{V}]}$. Observe that $\widetilde{G}$ has minimum degree $|\widetilde{V}| - \Delta_G - 1$ and $|\widetilde{V}| = |V(S)| \geq \Delta^2 + 4\Delta + 3$. Thus, the conditions of Corollary 6.4 are satisfied and hence $\widetilde{G}$ contains an $f'|_{\widetilde{V}}$-factor $\widetilde{G}' = (\widetilde{V}, S')$. Here, $f'|_{\widetilde{V}}$ denotes f restricted to the domain $\widetilde{V}$. By definition of $\widetilde{V}$ it follows that $G' = (V, S')$ is an f'-factor of $\overline{G}$.

To show (ii), assume towards a contradiction that $G + S'$ is not k-anonymous, that is, there exists some vertex $v \in V$ such that $1 \leq |B_{G+S'}(\deg_{G+S'}(v))| < k$. Recall that a block $B_G(d)$ of degree d is the set of all vertices with degree d in G. Let $d := \deg_{G+S}(v)$ and $d' := \deg_{G+S'}(v)$. Observe that $d' = \deg_G(v) + f'(v)$. Thus, if $v \notin X$, then by definition of f' it holds that $d' = \deg_G(v) + f(v) = d \leq \Delta_G + 2$. Hence, for all vertices $u \in B_{G+S}(d')$ it follows that $u \notin X$. Thus, $B_{G+S}(d') \subseteq B_{G+S'}(d')$ and since $G+S$ is k-anonymous we have $|B_{G+S'}(d')| \geq k$, a contradiction. If $v \in X$, that is, $d > \Delta_G + 2$, then $|B_{G+S}(d)| \geq k$ since $G+S$ is k-anonymous. Furthermore, by the definitions of $B_{G+S}(d)$, f, and X we have for all $u \in B_{G+S}(d)$ that $\deg_G(u) + f(u) = d$, $u \in X$, and, thus, $f'(u) + \deg_G(u) = d'$. Therefore, $B_{G+S}(d) \subseteq B_{G+S'}(d')$ and $|B_{G+S'}(d')| \geq k$, a contradiction. $\square$

Note that the bound provided in Lemma 6.5 is tight: Consider a cycle with an odd number of vertices plus two additional vertices that are adjacent to each other. The graph has maximum degree two. By setting $k := |V|$ we ensure that the k-anonymized graph is regular. Observe that inserting any k-insertion set ends up with a graph of maximum degree at least four: Since the two additional vertices need each at least one further neighbor, the maximum degree in the k-anonymized graph is least three. However, increasing the degree of each vertex to three is impossible as the sum of the degrees would be odd (odd number of vertices and odd degree) and in a graph the sum of the degree is always even. Hence, the k-anonymized graph has maximum degree at least four.

As a corollary of Lemma 6.5 we get the following unconditional bound on the maximum degree.

Corollary 6.6. *Let $G = (V, E)$ be an undirected graph and let S be a minimum-size k-insertion set. Then the maximum degree in $G + S$ is at most $\Delta^2 + 5\Delta + 2$.*

Proof. Let S be a k-insertion set for G such that the maximum degree in $G+S$ is more than $\Delta^2+5\Delta+2$. Thus, at least one vertex in G is incident to $\Delta^2+4\Delta+3$ edges in S and hence $|V(S)| \geq \Delta^2 + 4\Delta + 3$. It follows from Lemma 6.5 that S is not a minimum-size k-insertion set. Thus any minimum-size k-insertion set for G yields a graph with maximum degree at most $\Delta^2 + 5\Delta + 2$. □

Remark. We conjecture that the bound in Corollary 6.6 is *not* tight. The example with the highest resulting maximum degree after inserting a k-insertion set is a graph consisting of two disjoint cliques of order Δ and $\Delta+1$, respectively, and setting $k = n$. The only k-insertion set for this instance makes the whole graph a clique and, thus, doubles the degree. We thus conjecture that the bound in Corollary 6.6 can be improved to 2Δ.

Next, we formalize the anonymization of degree sequences. Recall that a multiset of positive integers $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ that corresponds to the degrees of all vertices in an undirected graph is called *degree sequence*—see Section 3.1 for more details about degree sequences. A degree sequence $\mathcal{S}$ is k-anonymous if each number in $\mathcal{S}$ occurs at least k times in $\mathcal{S}$. The degree sequence of a k-anonymous graph G is clearly k-anonymous. Moreover, if an undirected graph G can be transformed by at most s edge insertions into a k-anonymous graph, then the degree sequence of G can be transformed into a k-anonymous degree sequence by increasing the integers by no more than $2s$ in total (clearly, in the other direction this fails in general because of the graph structure). As we are in this section only interested in a degree sequence corresponding to a

graph of a Anonym E-Ins instance where s is large, by Lemma 6.5 we can require the integers in a k-anonymous degree sequence to be upper-bounded by $\Delta + 2$.

Degree Sequence Anonymity (k-DSA)

Input: Two positive integers k and s, and a degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ with $d_1 \leq d_2 \leq \ldots \leq d_n$ and $\Delta = d_n$.

Question: Is there a k-anonymous degree sequence $\mathcal{S}' = \{d'_1, d'_2, \ldots, d'_n\}$ with $d_i \leq d'_i$ and $\max_{1 \leq i \leq n} d'_i \leq \Delta + 2$ such that $\sum_{i=1}^{n} d'_i - d_i = 2s$?

Input:
$\mathcal{S} = \{1, 2, 3, 4\}, k = 2, s = 2$

Solution:
$\mathcal{S} = \{3, 3, 4, 4\}$

Observe that we require that the "cost" of anonymizing the degree sequence $\mathcal{S}$ is exactly $2s$ and not at most $2s$. This is due to the fact that we only can transfer "large" solutions of Degree Sequence Anonymity to Anonym E-Ins, as we will show later. In particular, if we allowed the cost of the solution to be at most $2s$, then we could always get "small" solutions to Degree Sequence Anonymity, which actually might not be realizable in the graph. Note that, due to the degree upper bound of $\Delta + 2$ and the required cost of exactly $2s$, Degree Sequence Anonymity is a modified variant compared to the original degree anonymization problem used in Liu and Terzi [LT08]. Hence, we need to slightly modify their dynamic programming-based approach to prove that Degree Sequence Anonymity is polynomial-time solvable.

Lemma 6.7. Degree Sequence Anonymity *can be solved in* $\mathcal{O}(nsk\Delta)$ *time.*

Proof. We slightly adapt a dynamic programming algorithm provided by Liu and Terzi [LT08, Section 4] and also used by Chester et al. [Che+13b, Section 6.2.2].

The dynamic programming uses a single table T with a boolean entry $T[i, j]$ for every $i \in \{1, 2, \ldots, n\}$ and $j \in \{0, 1, \ldots, 2s\}$. The entry $T[i, j]$ is `true` if and only if there is a k-anonymous sequence $\{d'_1, d'_2, \ldots, d'_i\}$ with $d'_t \geq d_t$ for all $t \in \{1, 2, \ldots, i\}$ and the cost $\sum_{t=1}^{i} d'_t - d_t$ of the anonymization is exactly j. Thus, $T[n, 2s]$ stores the answer to the Degree Sequence Anonymity problem.

Obviously, for $i < k$ we have $T[i, j] :=$ `false` for all j as there is no k-anonymous sequence with less than k numbers. To fill the rest of the table with

increasing i, we use for $1 \leq a \leq b \leq n$ and a positive integer d the function $\text{cost}(a, b, d) := \sum_{t=a}^{b} d - d_t$ (the cost of increasing $d_a, d_{a+1}, \ldots, d_b$ up to d).

For $k \leq i < 2k$ we set $T[i, j]$ to `true` if and only if there is a $d \in \{d_i, d_i + 1, \ldots, \Delta + 2\}$ such that $j = \text{cost}(1, i, d)$. We next prove the correctness of this assignment: Clearly, the corresponding sequence $d'_1 = \cdots = d'_i = d$ is k-anonymous. In the reverse direction, from $i < 2k$ it follows that $d'_1 = \cdots = d'_i$ for each k-anonymous sequence $d'_1, d'_2, \ldots, d'_i$. Hence, the entry $T[i, j]$ is computed correctly in this case.

For $i \geq 2k$ we set $T[i, j]$ to `true` if and only if there are $\ell \in \{k, k+1, \ldots, 2k-1\}$ and $d \in \{d_i, d_i+1, \ldots, \Delta+2\}$ such that $T[i-\ell, j-\text{cost}(i-\ell+1, i, d)] = \texttt{true}$. We next prove that this assignment is correct. In the first direction, corresponding to $T[i-\ell, j-\text{cost}(i-\ell+1, i, d)]$ let $d'_1, d'_2, \ldots, d'_{i-\ell}$ be a k-anonymous sequence for $d_1, d_2, \ldots, d_{i-\ell}$ with anonymization cost $j - \text{cost}(i-\ell+1, i, d)$. Then, since $d \geq d_i$, the sequence

$$d'_1, d'_2, \ldots, d'_{i-\ell}, \underbrace{d, d, \ldots, d}_{\ell}$$

is a k-anonymous sequence of cost j for $d_1, d_2, \ldots, d_i$. In the other direction, let $d'_1, d'_2, \ldots, d'_i$ be a k-anonymous sequence for $d_1, d_2, \ldots, d_i$ with anonymization cost j. Denote by ℓ the largest integer such that $d'_{i-\ell} = \cdots = d'_i$. Since the sequence is k-anonymous ℓ is at least k and if $\ell \geq 2k$, then set $\ell := k$. It follows that the sequence $d'_1, d'_2, \ldots, d'_{i-\ell}$ is k-anonymous and hence $T[i-\ell, j-\text{cost}(i-\ell+1, i, d'_i)] = \texttt{true}$. From this and since $\Delta + 2 \geq d'_i \geq d_i$ it follows that the entry $T[i, j]$ is computed correctly.

As each of the recurrences only depends on at most $k \cdot (\Delta + 2)$ other entries of the table and the table has $n(2s + 1)$ entries, the algorithm runs in $\mathcal{O}(nsk\Delta)$ time. It is easy to modify the algorithm to output the appropriate k-anonymous sequence in the same running time. □

We now have all ingredients to solve Anonym E-Ins in polynomial time in case it has a "large" minimum-size k-insertion set. The basic process is as follows (see Algorithm 6.1 for the pseudocode): Given an instance (G, k, s) of Anonym E-Ins first compute the degree sequence $\mathcal{S}$ of G. Then, search a "large" solution for $(\mathcal{S}, k, s)$, that is a solution of size i, $(\Delta^2 + 4\Delta + 3)^2 \leq i \leq s$. If there is such a large solution for the Degree Sequence Anonymity instance, then the next lemma states that this solution can be transferred to the Anonym E-Ins instance.

Algorithm 6.1: Find a k-insertion set of size at most s for G or decide that the size of a minimum k-insertion set for G is not between $(\Delta^2 + 4\Delta + 3)^2$ and s.

Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Output: A k-insertion set of size s', $(\Delta^2 + 4\Delta + 3)^2 \leq s' \leq s$, if it exists.

```
1  S ← degree sequence of G
2  j ← −1
3  i ← (Δ² + 4Δ + 3)²
4  while j = −1 and i ≤ s do
       // find minimum j s.t. (S, k, j) is a yes-instance of k-DSA
5      if (S, k, i) is a yes-instance of k-DSA then          // see Lemma 6.7
6          j ← i
7          S' ← solution for (S, k, i)
8      else
9          i ← i + 1
10 if j = −1 then        // no k-insertion set of size between (Δ² + 4Δ + 3)² and s
11     return 'NO'
12 else        // (G, k, s) is a yes-instance; the algorithm now computes a solution
13     foreach v_i ∈ V do
14         f(v_i) ← d'_i − deg_G(v_i)        // f(v_i) ≙ number of new incident edges
15     G' = (V, S) ← f-factor of Ḡ
16     return S
```

Lemma 6.8. *Let (G, k, s) be an instance of* Anonym E-Ins. *If the size of a minimum-size k-insertion set is at least $(\Delta^2 + 4\Delta + 3)^2$, then Algorithm 6.1 decides (G, k, s) in polynomial time. Furthermore, if Algorithm 6.1 returns an edge set S, then S is a k-insertion set of size $|S| \leq s$.*

Proof. We first show that if Algorithm 6.1 returns an edge set S, then S is a k-insertion set. Let S be an edge insertion set returned by the algorithm. First, observe that $|S| \leq s$ due to the while loop in Line 4. Since in Line 15 the algorithm determines an f-factor in $\overline{G}$ it follows that $S \cap E = \emptyset$. Furthermore, $(\mathcal{S}, k, |S|)$ is a yes-instance. Thus, by construction of f, it follows that $G + S$ is k-anonymous. Putting all this together implies that S is a k-insertion set of size $|S| \leq s$ and, hence, (G, k, s) is a yes-instance.

Now, let S be the minimum k-insertion set of size $|S| \geq (\Delta^2 + 4\Delta + 3)^2$ and $|S| \leq s$. We show that Algorithm 6.1 returns a k-insertion set. Observe that for any edge set S of size at least $(\Delta^2 + 4\Delta + 3)^2$ it holds that $|V(S)| > \sqrt{(\Delta^2 + 4\Delta + 3)^2} = \Delta^2 + 4\Delta + 3$. Thus, by Lemma 6.5, since S is minimum, $G + S$ has maximum degree $\Delta + 2$. Let $\mathcal{S}$ be the degree sequence of G. As already discussed before, the degree sequence $\mathcal{D}'$ of $G + S$ is a solution for Degree Sequence Anonymity. Thus, $(\mathcal{S}, k, |S|)$ is a yes-instance of Degree Sequence Anonymity. Hence, after leaving the while-loop in Line 4 it holds that $j \leq |S|$ and $\mathcal{S}'$ is the corresponding k-anonymous degree sequence. By definition, $\mathcal{D}'$ has a maximum degree of at most $\Delta + 2$. Hence, there are at least $(\Delta^2 + 4\Delta + 3)^2/(\Delta + 2) > \Delta^2 + 4\Delta + 3$ integers in $\mathcal{S}$ that have been increased to get $\mathcal{D}'$. Thus, for the function f computed in Line 14 it holds that $|\{v \in V \mid f(v) > 1\}| > \Delta^2+4\Delta+3$. Since $\overline{G}$ has minimum degree $|V|-\Delta-1$, if follows from Corollary 6.4 that $\overline{G}$ contains an f-factor. Thus, in Line 15 an f-factor $G' = (V, S)$ is found and the algorithm returns a k-insertion set.

Recall that f-Factor can be solved in $\mathcal{O}(\sqrt{\sum_{v \in V} f(v)}|E|)$ time [Gab83]. Together with Lemma 6.7, this implies that Algorithm 6.1 runs in polynomial time. □

Lemma 6.8 essentially shows that Anonym E-Ins can be decided in polynomial time when a minimum-size k-insertion sets is large. If a minimum-size k-insertion set is not large, then, since any k-insertion set for G of size $j \leq s$ directly implies that $(\mathcal{S}, k, j)$ is a yes-instance for Degree Sequence Anonymity, it follows that we can bound the parameter s by a function in Δ, as stated in the next lemma stating the mentioned win-win situation.

Lemma 6.9. *There is an algorithm running in $\mathcal{O}(ns^2k\Delta)$ time that given an instance (G, k, s) of* Anonym E-Ins *returns 'YES' or 'NO'. If it answers 'YES', then (G, k, s) is a yes-instance. If it returns 'NO', then (G, k, s) is a yes-instance if and only if $(G, k, \min\{(\Delta^2 + 4\Delta + 3)^2, s\})$ is a yes-instance.*

Proof. The algorithm is obtained by replacing the Lines 13 to 16 of Algorithm 6.1 with "return 'YES' ". If the algorithm returns 'YES', then, by Lemma 6.8, the input instance (G, k, s) is a yes-instance. If the algorithm returns 'NO', then consider the following two cases. Let S be a minimum-size k-insertion set.

Case 1 $|S| \geq (\Delta^2 + 4\Delta + 3)^2$**:** As the algorithm returns 'NO', it follows from Lemma 6.8 that the given instance (G, k, s) is a no-instance (thus $s < |S|$). Hence, also $(G, k, \min\{(\Delta^2 + 4\Delta + 3)^2, s\})$ is a no-instance.

Case 2 $|S| < (\Delta^2 + 4\Delta + 3)^2$**:** If $s < |S|$, then $s < (\Delta^2 + 4\Delta + 3)^2$ and, thus, (G, k, s) as well as $(G, k, \min\{(\Delta^2 + 4\Delta + 3)^2, s\})$ are no-instances. Conversely, if $s > |S|$, then $\min\{(\Delta^2 + 4\Delta + 3)^2, s\} > |S|$ and, hence, (G, k, s) as well as $(G, k, \min\{(\Delta^2 + 4\Delta + 3)^2, s\})$ are yes-instances. Hence, it holds that (G, k, s) is a yes-instance if and only if $(G, k, \min\{(\Delta^2 + 4\Delta + 3)^2, s\})$ is a yes-instance.

As to the running time, observe that the algorithm runs in $\mathcal{O}(ns^2k\Delta)$ time: The algorithm basically solves at most s instances of Degree Sequence Anonymity which requires $\mathcal{O}(ns^2k\Delta)$ time, see Lemma 6.7, and then returns 'YES' or 'NO'. □

We remark that Algorithm 6.1, constructing a solution if found, runs in $\mathcal{O}(n^3 + ns^2k\Delta)$ time: The first part of deciding whether there exists a large solution runs in $\mathcal{O}(ns^2k\Delta)$, see Lemma 6.9. Then, computing an f-factor in $\overline{G}$ is doable in $\mathcal{O}(\sqrt{\sum_{v \in V} f(v)}|E|)$ time [Gab83], that is, $\mathcal{O}(n^2\sqrt{n^2}) = \mathcal{O}(n^3)$ time.

6.3.3. Polynomial Kernel

In this section, we first show a kernel with respect to the combined parameter (Δ, s) and then use Lemma 6.9 to show that this kernel is polynomial only in Δ. Our kernelization algorithm is based on the following observation. For a given graph G, consider for some $1 \leq i \leq \Delta$ the block $B_G(i)$, that is, the set of all vertices of degree i. If $B_G(i)$ contains many vertices, then the vertices are "interchangeable":

Observation 6.10. *Let (G, k, s) with graph $G = (V, E)$ be a* Anonym E-Ins *instance, let S be an k-insertion set for G with $|S| \leq s$, and let $v \in V(S) \cap B_G(i)$ be a vertex such that $|B_G(i)| > (\Delta + 2)s$. Then there exists a vertex $u \in B_G(i) \setminus V(S)$ such that replacing in S every edge $\{v, w\}$ by $\{u, w\}$ results in a k-insertion set for G.*

Proof. Since $|S| \leq s$, the vertex v can be incident to at most s edges in S. Denoting the set of these edges by S^v, one obviously can replace v by $u \in B_G(i)$ if u is non-adjacent to all vertices in $V(S^v) \setminus \{v\}$ (this allows to insert all edges) and $u \notin V(S)$ (no block in $G + S$ does change its size). However, as $V(S)$ contains at most $2s$ vertices from $B_G(i)$ and each of the at most s vertices in $V(S^v) \setminus \{v\}$ has at most Δ neighbors in G, it follows that such a vertex $u \in B_G(i)$ exists if $|B_G(i)| > (\Delta + 2)s$. □

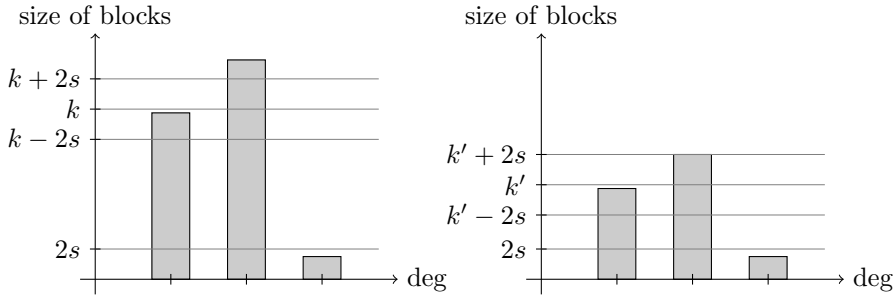


Figure 6.3.: Our general approach for the kernelization when k is "large": *Left:* Three blocks of the input. *Right:* The corresponding three blocks after the kernelization. As one can see, the value of k as well as the size of the "large" blocks is decreased. Note that in the input and in the output the first block needs the same number of vertices added to have size k, respectively k'. The idea is to keep for each block the "distance to being k-anonymous", that is, the difference between k and its size. If, however, one block (like the second block in the picture) has size larger than $2s + k$, then we need to keep only $2s + k'$ since with s edges we can "remove" at most $2s$ vertices from the block. (To simplify the proof we actually keep $4s + k'$ vertices.) Finally, observe that in the reduced instance we have $k' - 2s > 2s$ since we want to be able to distinguish between blocks that can be "filled" up to size k in the input (blocks of size $\geq k - 2s$) and blocks that can be emptied (blocks of size $\leq 2s$). See Algorithm 6.2 for the pseudocode of the kernelization.

By Observation 6.10, in our kernel we only need to keep at most $(\Delta + 2)s$ vertices in each block: If in an optimal k-insertion set S there is a vertex $v \in V(S)$ that we did not keep, then by Observation 6.10 we can replace v by some vertex we kept. There are two major problems that need to be fixed to obtain a kernel: First, when removing vertices from the graph, the degrees of the remaining vertices change. Second, k might be "large" and, thus, removing vertices (during kernelization) in one block may breach the k-anonymity constraint. To overcome the first problem we insert some "dummy-vertices" which are guaranteed not to be contained in any k-insertion set. To solve the second problem, however, we need to adjust the parameter k as well as the number of vertices that we keep from each block, see Figure 6.3 for our general approach.

Algorithm 6.2: The pseudocode or the algorithm computing a polynomial kernel with respect to (Δ, s).

Input: An undirected graph $G = (V, E)$ and two integers $k, s \in \mathbb{N}$.
Output: An undirected graph G', k', and s.

1 **if** $|V| \leq \Delta(\beta + 4s)$ **then** // β is defined as $\beta := (\Delta + 4)s + 1$
2 ⌊ **return** (G, k, s)
3 $k' \leftarrow \min\{k, \beta\}$; $A \leftarrow \emptyset$
4 **for** $i \leftarrow 1$ **to** Δ **do**
5 ⎸ **if** $2s < |B_G(i)| < k - 2s$ **then**
6 ⎸ ⌊ **return** *trivial no-instance* // insufficient budget for $B_G(i)$
7 ⎸ **if** $k \leq \beta$ **then** // determine retained vertices
8 ⎸ ⎸ $x \leftarrow \min\{|B_G(i)|, \beta + 4s\}$ // keep at most $\beta + 4s$ vertices
9 ⎸ **else if** $|B_G(i)| \leq 2s$ **then** // "small" block
10 ⎸ ⎸ $x \leftarrow |B_G(i)|$ // keep all vertices ("distance to size zero")
11 ⎸ **else** // "large" block and $k' = \beta$
12 ⎸ ⌊ $x \leftarrow k' + \min\{4s, (|B_G(i)| - k)\}$ // keep "distance to size k".
13 ⌊ add x vertices from $B_G(i)$ to A
14 $G' := G[A]$
15 **foreach** $v \in A$ **do** // insert vertices to preserve degree of retained vertices
16 ⌊ insert into G' $\deg_G(v) - \deg_{G'}(v)$ many degree-one vertices adjacent to v
17 denote with P the set of vertices inserted in Line 16
18 by inserting matched pairs of vertices, ensure that $|P| \geq \max\{4\Delta + 4s + 4, k'\}$
19 **if** $\Delta + s + 1$ *is even* **then**
20 ⎸ $G^F = (P, E^F) \leftarrow (\Delta + s + 1)$-factor in $\overline{G'[P]}$
21 **else**
22 ⌊ $G^F = (P, E^F) \leftarrow (\Delta + s + 2)$-factor in $\overline{G'[P]}$
23 $G' \leftarrow G' + E^F$
24 **return** (G', k', s)

Details of the kernelization algorithm. We now explain the kernelization algorithm in detail (see Algorithm 6.2 for the pseudocode). Let (G, s, k) be an instance of Anonym E-Ins. For brevity we set $\beta := (\Delta + 4)s + 1$. We compute in polynomial time an equivalent instance (G', k', s) with at most $\mathcal{O}(\Delta^3 s)$ vertices: First set $k' := \min\{k, \beta\}$ (Line 3). We arbitrarily select from each block $B_G(i)$ a certain number x of vertices and collect all these vertices into the set A (Line 13). To cope with the above mentioned second problem, the

"certain number" is defined in a case distinction on the value of k (see Lines 4 to 13). Intuitively, if k is large then we distinguish between "small" blocks of size at most $2s$ and "large" blocks of size at least $k - 2s$. Obviously, if there is a block which is neither small nor large, then the instance is a no-instance (see Line 6). Thus, in the kernel we keep for small blocks the "distance to size zero" and for large blocks the "distance to size k". Furthermore, in order to distinguish between small and large blocks it is sufficient that $k' > 4s$. However, to guarantee that Observation 6.10 is applicable, the case distinction is a little bit more complicated, see Lines 4 to 13. The idea is to take enough vertices from each block into A such that we can guarantee that any solution on G can be transformed to G' and vice versa. Intuitively, for this it is enough to select $2s$ vertices from each block, as no solution can "affect" more vertices.

In Line 14 we start building G' by first copying $G[A]$ into it. Next, adding a pendant vertex to v means that we insert a new vertex in G' and make it adjacent to v. For each $v \in A$ we add pendant vertices to v to ensure that $\deg_{G'}(v) = \deg_G(v)$ (Line 16). The vertices of A stay untouched in the following. Denote the set of all pendant vertices by P. Next, we insert enough pairwise adjacent vertices to P to ensure that $|P| \geq \max\{k', 4\Delta + 4s + 4\}$ (Line 18). Hence, $|P| \leq \max\{|A| \cdot \Delta, k', 4\Delta + 4s + 4\} + 1$. To avoid that vertices in P help to anonymize the vertices in A we "shift" the degree of the vertices in P (see Lines 19 to 23): We insert edges between the vertices in P to ensure that the degree of each vertex in P is $\Delta + s + 2$ (when $\Delta + s + 1$ is even) or $\Delta + s + 3$ (when $\Delta + s + 2$ is even). For the ease of notation let χ denote the new degree of the vertices in P. Observe that before inserting edges all vertices in P have degree one in G'. Thus, the minimum degree in $\overline{G'[P]}$ is $|P| - 2$. Furthermore, for each $v \in P$ we denote by $f(v)$ the number of incident edges v requires to have the described degree. It follows that $f(v)$ is even and hence $\sum_{v \in P} f(v)$ is even. Hence setting $a = b := \chi$ fulfills all conditions of Lemma 3.8. Thus, the required f-factor exists and can be found in $\mathcal{O}(|P|^2 \sqrt{|P|(\Delta + s)})$ time [Gab83]. This completes the description of the kernelization algorithm.

The key point of the correctness of the kernelization is to show that without loss of generality, no k-insertion set S for G' of size $|S| \leq s$ affects any vertex in P. This is ensured by "shifting" the degree of all vertices in P by $s + 1$ (or $s + 2$), implying that none of the vertices in A can "reach" the degree of any vertex in P by inserting at most s edges. Hence each block either is a subset of A or of P. We now prove that we may assume that an edge insertion set does not affect any vertex in P. To prove this, all what we need is the fact that

A contains at least $\beta + 4s$ vertices from at least one block in G. Observe that this is ensured by the condition in Line 1.

Lemma 6.11. *Let (G, k, s) be an instance of* Anonym E-Ins *and let (G', k', s) be the instance computed by Algorithm 6.2. If there is a k-insertion set S for G' with $|S| \leq s$, then there is also a k-insertion set S' for G' with $|S'| = |S|$ such that $V(S') \cap P = \emptyset$.*

Before proving Lemma 6.11, we introduce the term "co-matching" and make an observation concerning its existence. An undirected graph $G = (V, E)$ contains a *co-matching* of size ℓ if its complement graph $\overline{G}$ contains a matching of size ℓ, that is, a subset of ℓ non-overlapping edges of $\overline{G}$. A *perfect* co-matching of G is a co-matching of size $|V|/2$. The following observation shows sufficient conditions for the existence of co-matchings.

Observation 6.12. *Let $G = (V, E)$ be an undirected graph and let $V' \subseteq V$ be a vertex subset such that $|V'| \geq 2\Delta + 1$ and $|V'|$ is even. Then, $G[V']$ contains a perfect co-matching.*

Proof. Since $|V'| \geq 2\Delta + 1$, it follows that in $\overline{G[V']}$ every vertex has degree at least $|V'| - \Delta \geq |V'|/2$. By Dirac's Theorem [Die10] stating that every n-vertex graph with minimum degree larger than $n/2$ contains a Hamiltonian cycle, it follows that $\overline{G[V']}$ contains a Hamiltonian cycle C. If $|V'|$ is even, then taking every second edge of C results in a perfect matching. □

We now can prove Lemma 6.11.

Proof of Lemma 6.11. Let S be a k-insertion set S for G' such that $|S| \leq s$ and $V(S) \cap P \neq \emptyset$. As each block in $G' + S$ is either a subset of A or of P, it follows from $V(S) \cap P \neq \emptyset$ that $|V(S) \cap P| \geq k$. Additionally, as S can affect at most $2s$ vertices and A contains at least $\beta + 4s$ vertices from at least one block, say $B_G(i)$, it follows that $B_{G'+S}(i)$ contains at least $\beta + 2s$ unaffected vertices.

We next restructure S in order to get a k-insertion set fulfilling the claimed properties. For this, one has to exchange all edges in S containing at least one endpoint from P. We start with the edges in S having only one endpoint in P.

Let $A^P \subseteq V(S) \cap A$ be all vertices in A that are incident to some edge in S with the second endpoint in P. For each $v \in A^P$ we select $|(N_{G'+S}(v) \setminus N_{G'}(v)) \cap P|$ vertices among the unaffected vertices from $B_{G'+S}(i)$ and replace each edge in S from v to some vertex in P (there are exactly $|(N_{G'+S}(v) \setminus N_{G'}(v)) \cap P|$ many) by an edge from v to one of the selected vertices (each unaffected vertex

in $B_{G'+S}(i)$ is only used once). Note that this is always possible since each vertex v has at most Δ neighbors among the unaffected vertices in $B_{G'+S}(i)$, since there are at least $s + \Delta + 1$ unaffected vertices in $B_{G'+S}(i)$, and since there can be at most s edges in S that are replaced in this way.

Note that, after having exchanged all edges in S with one endpoint in P, $B_{G'+S}(i)$ still contains at least $\beta + 2s > 2\Delta + 2s$ unaffected vertices. Thus, by Observation 6.12, there exists a co-matching of size exactly $|S| \leq s$ among the unaffected vertices in $B_{G'+S}(i)$. Exchanging each edge in S with two endpoints in P by an edge in this matching yields the following: All vertices in P are unaffected. Hence, the block containing all vertices from P is of size at least k. Additionally, we increased for at least k vertices the degree from i to $i + 1$, thus $|B_{G'+S}(i + 1)| \geq k$. As the block $B_{G'+S}(i)$ still contains at least k vertices after restructuring, it follows that $G' + S$ is k-anonymous. □

Based on Lemma 6.11 we now prove the correctness of our kernelization algorithm.

Lemma 6.13. *If the instance (G', k', s) constructed by Algorithm 6.2 is a yes-instance, then (G, k, s) is a yes-instance.*

Proof. First, observe that if $k \leq \beta$, then $k' = k$ and each edge insertion set that makes G' k-anonymous also makes G k-anonymous as all blocks with less than $\beta + 4s$ vertices remain unchanged. Hence, assume that $k > \beta$ and, thus, $k' = \beta < k$.

Let S' be an edge insertion set with $|S'| \leq s$ such that $G' + S'$ is k-anonymous and $S' \cap P = \emptyset$ (see Lemma 6.11). To prove that $G + S'$ is also k-anonymous, assume towards a contradiction that there is a block $B_{G+S'}(j)$ with $0 < |B_{G+S'}(j)| < k$. We associate two numbers $d_i^G(j), d_o^G(j)$ to S' with respect to G where $d_i^G(j)$ is the number of vertices in $B_{G+S'}(j)$ but not in $B_G(j)$ and $d_o^G(j)$ is the number of vertices in $B_G(j)$ but not in $B_{G+S'}(j)$. Defining the numbers analogously for G', it holds that $d_i^G(j) = d_i^{G'}(j)$ and $d_o^G(j) = d_o^{G'}(j)$.

If $|B_{G'+S'}(j)| = 0$, then $d_o^{G'}(j) = |B_{G'}(j)| \leq 2s$ and $d_i^{G'}(j) = 0$. By Line 10 this implies $B_{G+S'}(j) = \emptyset$. Consider the remaining case, that is, $|B_{G'+S'}(j)| \geq k'$. If $|B_G(j)| \geq k + 2s$, then $|B_{G+S'}|(j) \geq k$. Otherwise $|B_{G'}(j)| = k' + |B_G(j)| - k$ by Line 12. But then we have

$$\begin{aligned} 0 \leq |B_{G'+S'}(j)| - k' &= |B_{G'}(j)| + d_i^{G'}(j) - d_o^{G'}(j) - k' = \\ &= |B_G(j)| + d_i^G(j) - d_o^G(j) - k = |B_{G+S'}(j)| - k. \end{aligned}$$

and, hence, $|B_{G+S'}(j)| \geq k$. □

Lemma 6.14. *If (G, k, s) is a yes-instance, then the instance (G', k', s) constructed by Algorithm 6.2 is a yes-instance.*

Proof. Recall that $k' = \min\{k, \beta\} = \min\{k, (\Delta+4)s+1\}$. Let S be a k-insertion set for G of size at most s. We now show how to construct a k'-insertion set S' for G' of size at most s. If $V(S) \setminus A \neq \emptyset$, then we do the following to ensure $V(S) \subseteq A$. We initialize $S^1 := S$. Observe that for each vertex $v \in V(S) \setminus A$ it holds that $|B_G(\deg_G(v)) \cap A| \geq \beta - 2s > (\Delta + 2)s$. Hence, by Observation 6.10, there exists a vertex $u \in B_G(\deg_G(v)) \cap A$ such that the set S^2 resulting from S^1 by replacing v with u, formally, $S^2 := S^1 \cup \{\{u, w\} \mid \{v, w\} \in S^1\} \setminus \{\{v, w\} \mid \{v, w\} \in S^1\}$, is also a k-insertion set for G. Note that $V(S^2)$ has larger overlap with A as $V(S^1)$, more precisely, $|V(S^2) \cap A| = |V(S^1) \cap A| + 1$. By iteratively applying this procedure we end up with a k-insertion set S' for G with $V(S') \subseteq A$.

We next show that $G' + S'$ is k'-anonymous. Observe that if $k \leq \beta$, then $k = k'$ and all blocks in G' with less than $\beta + 4s > k + 2s$ vertices remained unchanged during the kernelization (see Line 8). Hence, all these blocks fulfill the k-anonymity requirement in $G' + S'$. Furthermore, all blocks with more than $k + 2s$ vertices in G also contain more than $k + 2s$ vertices in G' and more than k vertices in $G' + S'$. Thus, $G' + S'$ is k'-anonymous.

Now assume that $k > \beta$ and, thus, $k' = \beta$. Assume towards a contradiction that there is a block with $0 < |B_{G'+S'}(i)| < k'$. Observe that if $|B_{G'}(i)| \leq 2s$, then also $|B_G(i)| \leq 2s$, thus $B_{G'}(i) = B_G(i)$ (see Line 10) and $B_{G'+S'}(i) = B_{G+S'}(i)$, a contradiction to the assumption that $G+S'$ is k-anonymous. Hence, consider the case $|B_{G'}(i)| \geq 2s$ and, thus, $|B_G(i)| \geq k - 2s$ and $|B_{G'}(i)| = \beta + \min\{4s, (|B_G(i)| - k)\}$ (see Line 12). Observe that $|B_{G'+S'}(i)| - |B_{G'}(i)| = |B_{G+S'}(i)| - |B_G(i)|$ and, thus,

$$|B_{G'+S'}(i)| = (|B_{G+S'}(i)| - |B_G(i)|) + |B_{G'}(i)|. \tag{6.1}$$

Furthermore, observe that $|B_{G+S'}(i)| - |B_G(i)| \geq -2s$ and $|B_{G+S'}(i)| \geq k$. We now distinguish the two cases $|B_G(i)| - k \geq 4s$ and $|B_G(i)| - k < 4s$. In the first case it follows that $|B_{G'}(i)| = \beta + 4s$ and from Equation (6.1) it follows

$$|B_{G'+S'}(i)| \geq -2s + \beta + 4s > k',$$

a contradiction. In the second case it follows that $|B_{G'}(i)| = \beta + |B_G(i)| - k$ (see Line 12), and from Equation (6.1) we conclude that

$$|B_{G'+S'}(i)| \geq k - |B_G(i)| + \beta + |B_G(i)| - k = \beta = k',$$

a contradiction. □

From Lemmas 6.13 and 6.14 it follows that the kernelization algorithm is correct. It is not hard to see that the size of the computed instances is bounded by a polynomial in Δ and s, leading to the following.

Theorem 6.15. Anonym E-Ins *admits a kernel with $\mathcal{O}(\Delta^3 s)$ vertices. The kernelization runs in $\mathcal{O}(\Delta^8 s^3 + \Delta^2 sn)$ time.*

Proof. The kernel is computed by Algorithm 6.2. The correctness of the kernelization algorithm follows from Lemmas 6.13 and 6.14. Observe that each block in A has size at most $\beta + 4s$ (see Lines 8, 10 and 12). Thus, $|A| = \mathcal{O}(\Delta\beta) = \mathcal{O}(\Delta^2 s)$. Furthermore, the set P contains at most $\max\{\Delta|A|, k', 4s + 4\Delta + 1\}$ vertices (see Lines 16 to 18). Thus, $|P| = \mathcal{O}(\Delta^3 s)$ and, hence, the reduced instance contains $\mathcal{O}(\Delta^3 s)$ vertices.

It remains to show the running time. To this end, using bucket sort, one can sort the n vertices by degree in $\mathcal{O}(n)$ time. Furthermore, in the same time one can create Δ lists—each list containing the vertices of some degree i, $1 \leq i \leq \Delta$. Then, the selection of the $\mathcal{O}(\Delta^2 s)$ vertices of A can be done in $\mathcal{O}(\Delta^2 sn)$ time. Clearly, inserting the vertices in P can be done in $\mathcal{O}(\Delta^3 s)$ time. Finally, as P contains $\mathcal{O}(\Delta^3 s)$ vertices and an $(\Delta+s+1)$-factor in $\overline{G[P]}$ can be found in $\mathcal{O}(|P|^2\sqrt{|P|(\Delta+s)})$ time [Gab83], Algorithm 6.2 runs in $\mathcal{O}(\Delta^6 s^2\sqrt{\Delta^3 s(\Delta+s)} + \Delta^2 sn) = \mathcal{O}(\Delta^8 s^3 + \Delta^2 sn)$ time. □

By Lemma 6.9 it follows that in $\mathcal{O}(ns^2 k\Delta)$ time we can either decide the instance or we have $s \leq (\Delta^2 + 4\Delta + 3)^2$. By Theorem 6.15 this implies our main result—a polynomial kernel with respect to the maximum degree.

Theorem 6.16. Anonym E-Ins *admits an $\mathcal{O}(\Delta^7)$-vertex kernel. The kernelization runs in $\mathcal{O}(\Delta^8 s^3 + (sk + \Delta)\Delta sn)$ time.*

6.4. Fixed-Parameter Algorithm for the Parameter Maximum Degree

Theorem 6.16 already implies that Anonym E-Ins is fixed-parameter-tractable with respect to the parameter maximum degree. In this section, however, we provide a faster, direct combinatorial algorithm for the combined parameter (Δ, s) and, by Lemma 6.9, also for the parameter Δ.

Roughly speaking, for fixed k-insertion set S the algorithm branches into all suitable structures of $G[S]$, that is, graphs of at most $2s$ vertices with vertex

labels from $\{1, 2, \ldots, \Delta\}$. Then the algorithm checks whether the respective structure occurs as a subgraph in $\overline{G}$ such that the labels on the vertices match the degree of the corresponding vertex in G.

Theorem 6.17. Anonym E-Ins *can be solved in* $s^2(6s^2\Delta^2)^{2s} \cdot (n+m)$ *time.*

Proof. Let (G, k, s) be an instance of Anonym E-Ins. Let S be a k-insertion set S of size at most s and consider the graph $G[S]$ that is induced by the edges in S. Clearly, $G[S]$ contains at most $2s$ vertices and we label each vertex with its initial degree (some vertices might have the same label). Roughly speaking, we branch into all possibilities for the structure (label of vertices and which "labels" are connected by an edge) of the graph $G[S]$ and then try to find the structure as a subgraph in $\overline{G}$.

More specifically, we first branch into all possibilities to first choose the right number of edges and vertices in $G[S]$. We then branch into all possibilities to choose for each vertex its label, that is, its degree in G. Note that there are at most Δ^{2s} possibilities. Finally, we branch into the at most $\binom{2s}{2}^s \leq 4^s s^{2s}$ possibilities to choose pairs of vertices that are connected by an edge from S. Denote the guessed graph by G^S. Clearly, if G^S corresponds to $G[S]$, then $\overline{G}$ contains G^S. We now give an algorithm that finds the subgraph G^S in $\overline{G}$ if it exists. First, note that there are at most $2s$ vertices in G^S and each of them has degree at most Δ in G. Hence, if a block $B_G(d)$ has size at least $(2s-1)\Delta + 2s$, then it is always possible to choose a vertex from $B_G(d)$ that is non-adjacent to all vertices in a size-at-most-$(2s-1)$ vertex subset where at most s edges have been inserted. Thus we first can ignore vertices in G^s labeled with d where $|B_G(d)| \geq 3s\Delta$. For all other vertices we branch again into the at most $\binom{3s\Delta}{2s} \leq (3s\Delta)^{2s}$ possibilities to choose them from the "small" blocks. Afterwards we greedily insert the required vertices from the blocks of size at least $3s\Delta$ such that they are non-adjacent to the vertices chosen before. As this can be done in $\mathcal{O}(s(n+m))$ time (iterating over the graph for each vertex that needs further neighbors), the algorithm runs overall in $s \cdot \Delta^{2s} \cdot 4^s s^{2s} \cdot (3s\Delta)^{2s} \cdot s(n+m) = s^2(6s^2\Delta^2)^{2s} \cdot (n+m)$ time. The correctness of the algorithm follows from the exhaustive search. □

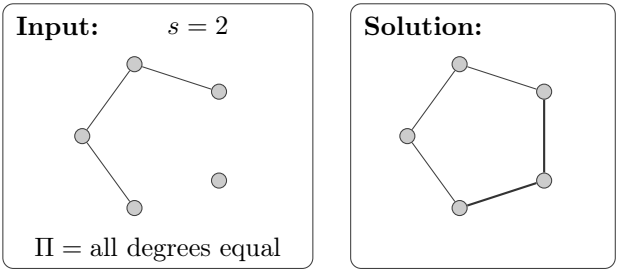
Note that due to the upper bound $s < (\Delta^2 + 4\Delta + 3)^2$ (see Lemma 6.9) and the polynomial kernel for the parameter Δ (see Theorem 6.16), Theorem 6.17 also provides the following.

Corollary 6.18. Anonym E-Ins *can be solved in* $\mathcal{O}(\Delta^{\mathcal{O}(\Delta^4)} + \Delta^8 s^3 + (sk + \Delta)\Delta sn)$ *time.*

6.5. A General Approach for Degree Sequence Completion Problems

In this section, we generalize the ideas behind the polynomial-size problem kernel for Anonym E-Ins, which we presented in Section 6.3. To this end, for some tuple property Π, we consider the following problem.

Π-Degree Sequence Completion (Π-DSC)
Input: An undirected graph $G = (V, E)$, an integer $s \in \mathbb{N}$.
Question: Is there a set of edges $E' \subseteq \binom{V}{2} \setminus E$ with $|E'| \leq s$ such that the degree sequence of $G' := G + E'$ fulfills Π?



In Section 3.2, we reviewed some work done for the f-Factor problem and more general degree factors. As already discussed in Section 3.2.2, all problems mentioned in Section 3.2 have in common that one only has to *locally* satisfy the degree of each vertex. Contrasting this *local* view, the degree constraints in Anonym E-Ins can only be *globally* satisfied. Π-DSC generalizes all these *globally* defined problems. There are many NP-complete problems of this kind, as will be discussed in Section 6.5.2.

6.5.1. Fixed-Parameter Tractability of Π-DSC

In this section, we first generalize the ideas behind Observation 6.10 to show fixed-parameter tractability of Π-DSC with respect to the combined parameter (s, Δ_G). Then, we present an adjusted version of Lemma 6.9 and apply it to show fixed-parameter tractability for Π-DSC with respect to the parameter maximum degree $\Delta_{G'}$ of the resulting graph.

Before we show our results, we make some basic remarks. Note that in the general setting of Π-DSC, there is no equivalent of Lemma 6.5 bounding

the maximum degree in the target graph by a function only depending of the maximum degree of the input graph since the property Π could for example require to turn the input graph into a clique. Furthermore, the generalization of Observation 6.10 does not lead to a polynomial problem kernel as deleting vertices and edges without knowing the property Π seems impossible.

A prerequisite for the mentioned fixed-parameter tractability results for Π-DSC is that the following problem is fixed-parameter tractable with respect to the parameter $\Delta_{\mathcal{S}} := \max\{d_1, d_2, \ldots, d_n\}$.

Π-Decision
Input: A degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$.
Question: Does $\mathcal{S}$ fulfill Π?

Clearly, if deciding whether a degree sequence satisfies the property Π is not fixed-parameter tractable, then the more general problem Π-DSC cannot be fixed-parameter tractable either.

Fixed-parameter tractability with respect to (s, Δ_G). The basic idea behind the polynomial-size problem kernel provided in Section 6.3.3 is presented in Observation 6.10, stating that the vertices in one block are interchangeable given that there are enough, that is at least $(\Delta_G + 2)s$, of them. Thus, it suffices to consider $(\Delta_G + 2)s$ vertices of each block and if there is a solution, then there is also a solution that only affects these at most $(\Delta_G + 2)s \cdot \Delta_G$ vertices. In the remaining six pages of Section 6.3.3, we are "just" dealing with the two problems that deleting the remaining vertices changes the degrees of the vertices we keep and the anonymity level k needs to bounded in some function of s and Δ_G. Due to the more general setting, we cannot deal with these problems when considering Π-DSC. We can, however, generalize Observation 6.10. To this end, we need one further definition. A subset $V' \subseteq V$ is an *α-block set* if V' contains for every $d \in \{0, 1, \ldots, \Delta_G\}$ exactly $\min\{\alpha, |B_G(d)|\}$ vertices, where $\alpha := (\Delta_G + 2)s$.

Lemma 6.19. *Let $I := (G = (V, E), s)$ be a yes-instance of Π-DSC and let $C \subseteq V$ be an $(\Delta_G+2)s$-block set. Then, there exists a set of edges $E' \subseteq \binom{C}{2} \setminus E$ with $|E'| \leq s$ such that the degree sequence of $G + E'$ fulfills Π.*

Proof. Let $I := (G = (V, E), s)$ be a yes-instance of Π-DSC and let $C \subseteq V$ be an $(\Delta_G + 2)s$-block set. Thus, there exists a set of edges $E' \subseteq \binom{V}{2} \setminus E$ with $|E'| \leq s$ such that the degree sequence $\mathcal{S} = \{d'_1, d'_2, \ldots, d'_n\}$ of $G' := G + E'$

fulfills Π. If $V(E') \subseteq C$, then there is nothing to prove. Hence, assume that there exists a vertex $v \in V(E') \setminus C$. We show how to construct from E' an edge set E'' for I such that $(V(E') \setminus C) \setminus V(E'') = \{v\}$ and the degree sequence of $G'' := G + E''$ equals $\mathcal{S}$. Since v is not in the $(\Delta_G + 2)s$-block set C, it follows that $|C \cap B_G(\deg_G(v))| = s(\Delta_G + 2)$. Next, we prove that there is a vertex $u \in B_G(\deg_G(v))$ such that $u \notin V(E')$ and $u \notin N_G(N_{G[E']}(v))$, that is, u is not incident to any edge in E' and also not adjacent to any vertex that is connected to v by an edge in E'. Note that "replacing" v by such a vertex u in the edge set E' yields E'': Formally, for

$$E'' := \{\{u, w\} \mid \{v, w\} \in E'\} \cup \{\{w_1, w_2\} \mid \{w_1, w_2\} \in E' \wedge w_1 \neq v \wedge w_2 \neq v\},$$

it holds that $E'' \cap E = \emptyset$ and since $u \in B_G(\deg_G(v))$, the degree sequence of $G + E''$ is $\mathcal{S}$. Hence, it remains to show that such a vertex u exists, that is, $(C \cap B_G(\deg_G(v))) \setminus (V(E') \cup N_G(N_{G[E']}(v)))$ is indeed non-empty. This is true since $|C \cap B_G(\deg_G(v))| = s(\Delta_G + 2)$, whereas $|V(E') \cup N_G(N_{G[E']}(v))| < 2s + s\Delta_G$. By repeatedly applying this procedure, we obtain a solution $E''' \subseteq \binom{C}{2} \setminus E$ with $|E'''| \leq s$ such that the degree sequence of $G + E'''$ fulfills Π. $\square$

Observe that Lemma 6.19 does not lead to a problem kernel but only to a reduced search space for a solution, namely any $(\Delta_G + 2)s$-block set. Clearly, an $(\Delta_G + 2)s$-block set C can be computed in polynomial time. Then, one can simply try out all possibilities to insert edges with endpoints in C and check whether in one of the cases the degree sequence of the resulting graph satisfies Π. As $|C| \leq (\Delta_G + 2)s(\Delta_G + 1)$, there are at most $\mathcal{O}(2^{((\Delta_G+2)s(\Delta_G+1))^2})$ possible subsets of edges to insert. Altogether, this leads to the following theorem.

Theorem 6.20. *Let Π be some degree sequence property. If Π-Decision is fixed-parameter tractable with respect to the maximum degree $\Delta_{\mathcal{S}}$, then Π-DSC is fixed-parameter tractable with respect to (s, Δ_G).*

Bounding the solution size s in $\Delta_{G'}$. We now show how to extend the ideas of Section 6.3.2 to the context of Π-DSC in order to bound the solution size s by a polynomial in the maximum degree $\Delta_{G'}$ of the resulting graph. The general procedure still is the one inspired by Liu and Terzi [LT08]: Solve the number problem corresponding to Π-DSC on the degree sequence of the input graph and then try to "realize" the solution. To this end, we define the corresponding number problem as follows:

Π-Number Sequence Completion (Π-NSC)
Input: Positive integers $d_1, d_2, \ldots, d_n, s, \Delta$.
Question: Are there n nonnegative integers $x_1, x_2, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = s$ such that $\{d_1+x_1, d_2+x_2, \ldots, d_n+x_n\}$ fulfills Π and $d_i+x_i \leq \Delta$?

With this problem definition, we can now generalize Lemma 6.8.

Lemma 6.21. *Let $I := (G, s)$ be an instance of* Π-DSC *with $V = \{v_1, v_2, \ldots, v_n\}$ and $s \geq \Delta_{G'}(\Delta_{G'}+1)^2$. If there exists an $s' \in \{\Delta_{G'}(\Delta_{G'}+1)^2, \Delta_{G'}(\Delta_{G'}+1)^2+1, \ldots, s\}$ such that the corresponding* Π-NSC *instance $I' := (\deg(v_1), \deg(v_2), \ldots, \deg(v_n), 2s', \Delta_{G'})$ is a yes-instance, then I is a yes-instance.*

Proof. Let $I' := (\deg(v_1), \deg(v_2), \ldots, \deg(v_n), 2s', \Delta_{G'})$ with $s' \in \{\Delta_{G'}(\Delta_{G'}+1)^2, \Delta_{G'}(\Delta_{G'}+1)^2+1, \ldots, s\}$ be a yes-instance of Π-NSC and let $x_1, x_2, \ldots, x_n$ denote a solution for I'. Defining the function $f\colon V \to \mathbb{N}$ as $f(v_i) := x_i$, we now prove that $\overline{G}$ contains an f-factor which forms a solution E' for I. Denote by A the set of affected vertices, formally, $A := \{v_i \in V \mid x_i > 0\}$. Observe that $|A| \geq 2s'/\Delta_{G'} \geq 2(\Delta_{G'}+1)^2$ as $s' \geq \Delta_{G'}(\Delta_{G'}+1)^2$. Furthermore, as the maximum degree Δ_G in G is upper-bounded by$\Delta_{G'}$, it follows that $\overline{G[A]}$ has minimum degree at least $|A| - \Delta_{G'} - 1$. Finally, observe that $f(v_i) \in \{1, 2, \ldots, \Delta_{G'}\}$ for each $v_i \in A$ and that $\sum_{v_i \in A} f(v_i) = 2s'$ is even. Hence, by Corollary 6.4, $\overline{G[A]}$ contains an f-factor. Thus, $\overline{G}$ also contains an f-factor $G' = (V, E')$ and since $(\deg(v_1)+x_1, \deg(v_2)+x_2, \ldots, \deg(v_n)+x_n)$ fulfills Π, it follows that E' is a solution for I, implying that I is a yes-instance. □

Let function $g(|I|)$ denote the running time for solving the Π-NSC instance I. Clearly, if there is a solution for an instance of Π-DSC, then there also exists a solution for the corresponding Π-NSC instance. It follows from Lemma 6.21 that we can decide whether there is a large solution for Π-DSC (inserting at least $\Delta_{G'}(\Delta_{G'}+1)^2$ edges) in $s \cdot g(n \log(\Delta))$ time. Hence, we arrive at the following win-win situation corresponding to Lemma 6.9:

Corollary 6.22. *Let g denote the running time for solving* Π-NSC*. There is an algorithm running in $g(n \log(\Delta)) \cdot n^{\mathcal{O}(1)}$ time that given an instance $I := (G, s)$ of* Π-DSC *returns "yes" or "no" such that if it answers "yes", then I is a yes-instance, and otherwise I is a yes-instance if and only if $(G, \min\{s, \Delta_{G'}(\Delta_{G'}+1)^2\})$ is a yes-instance.*

Using Corollary 6.22, we can transfer the fixed-parameter tractability for Π-NSC with respect to Δ to a fixed-parameter tractability result for Π-DSC

with respect to $\Delta_{G'}$. Note that maximum degree $\Delta_{G'}$ in the output graph is at most $s + \Delta_G$, that is, $\Delta_{G'}$ is a smaller and thus "stronger" parameter [KN12]. Also, showing Π-NSC to be fixed-parameter tractable with respect to Δ might be a significantly easier task than proving fixed-parameter tractability for Π-DSC with respect to $\Delta_{G'}$ directly since the graph structure can be completely ignored.

Theorem 6.23. *If* Π-NSC *is fixed-parameter tractable with respect to* Δ*, then* Π-DSC *is fixed-parameter tractable with respect to* $\Delta_{G'}$*.*

Proof. Let $I := (G, s)$ be a Π-DSC instance. First, note that $\Delta_G \leq \Delta_{G'}$ always holds since we are only inserting edges to G. Thus, if $s \leq \Delta_{G'}(\Delta_{G'} + 1)^2$, then the fixed-parameter tractability with respect to (s, Δ_G) from Theorem 6.20 yields fixed-parameter tractability with respect to $\Delta_{G'}$. Otherwise, we use Corollary 6.22 to check whether there exists a large solution of size at least $\Delta_{G'}(\Delta_{G'} + 1)^2$. Hence, by assumption, in $f(\Delta_{G'}) \cdot n^{\mathcal{O}(1)}$ time for some computable function f, we either find that I is a yes-instance or we can assume that $s \leq \Delta_{G'}(\Delta_{G'} + 1)^2$, which altogether yields fixed-parameter tractability with respect to $\Delta_{G'}$. □

If Π-NSC can be solved in polynomial time, then Corollary 6.22 shows that we can assume that $s \leq \Delta_{G'}(\Delta_{G'} + 1)^2$. Furthermore, it clearly holds that $\Delta_G \leq \Delta_{G'}$. These two inequalities imply, similar to the Anonym E-Ins setting (Theorem 6.16), that polynomial kernels with respect to (s, Δ_G) transfer to the parameter $\Delta_{G'}$. We thus arrive at the following.

Theorem 6.24. *If* Π-NSC *is polynomial-time solvable and* Π-DSC *admits a polynomial kernel with respect to* (s, Δ_G)*, then* Π-DSC *also admits a polynomial kernel with respect to* $\Delta_{G'}$*.*

6.5.2. Applications of Degree Sequence Completion Problems

Besides the graph anonymization setting, one could think of further, more generalized constraints on the degree sequence. For example, if $p_i(d)$ denotes how often degree i appears in a degree sequence $\mathcal{S}$, then being k-anonymous translates into $p_i(\mathcal{S}_{G'}) \geq k$ for all degrees i occurring in the degree sequence $\mathcal{S}_{G'}$ of the modified graph G'. Now, it is natural to consider not only a lower bound $k \leq p_i(\mathcal{S})$ but also an upper bound $p_i(\mathcal{S}) \leq u$ or maybe even a set of allowed frequencies $p_i(\mathcal{S}) \in F_i \subseteq \mathbb{N}$. Constraints like this allow to express some

properties, not of individual degrees itself but on the whole distribution of the degrees in the resulting sequence. For example, to have some "balancedness" one can require that each occurring degree occurs exactly ℓ times for some $\ell \in \mathbb{N}$ [Cha+89]. To obtain some sort of "robustness" it might be useful to ask for an H-index of ℓ, that is, in the solution graph there are at least ℓ vertices with degree at least ℓ [ES12].

Another range of problems which fit naturally into our framework involves completion problems to a graph class that is completely characterized by degree sequences. Many results concerning the relation between a degree sequence and the corresponding realizing graph are known and can be found in the literature, see also Section 3.1. Based on this characterization researchers characterized for example pseudo-split, split, and threshold graphs completely by their degree sequences [HIS75, HS81, MP94]. The NP-complete Split Graph Completion [NSS01] problem, for example, is known to be fixed-parameter tractable with respect to the allowed number of edge insertions [Cai96]. Note, however, that for the mentioned graph classes polynomial kernels with respect to the parameter $\Delta_{G'}$ trivially exist because here we always have $\sqrt{n} \leq \Delta_G \leq \Delta_{G'}$.

We finish with another interesting example of a class of graphs characterized by their degree sequence: A graph is a *unigraph* if it is determined by its degree sequence up to isomorphism [BLS99]. Given a degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$, deciding whether $\mathcal{S}$ defines a unigraph can be done in linear time [BCP11, KL75]. Again, by Theorem 6.23, we conclude fixed-parameter tractability for the unigraph completion problem with respect to the parameter maximum degree $\Delta_{G'}$ in the solution graph G'.

6.6. Upper and Lower Bounds for Degree Anonymity by Edge Insertion

In Section 6.3, we used the heuristic approach of Liu and Terzi [LT08] to develop an algorithm for Anonym E-Ins that provides solutions of size at least $2\Delta^4$ in polynomial time. Based on this theoretical result, we present in this section an enhancement of their 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 solutions. The experimental evaluation of our algorithms, presented in the next section, reveals

that about 21 % of the real-world data can be solved optimally; whereas in the other cases our upper bounds significantly improve on known heuristic solutions.

In Section 6.6.1, we provide a general description how the problem is split into several subproblems (basically corresponding to the two-phase approach of Liu and Terzi [LT08]) and in Sections 6.6.2 and 6.6.3, we describe the corresponding algorithms in detail. In Section 6.6.4, we describe two upper-bound heuristics.

Notation. Before we describe our algorithms, we recall some necessary notation. The block sequence $\mathcal{B}_{\mathcal{S}}$ of a degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ with maximum degree $\Delta := \max\{d_i\}$ is defined as $\mathcal{B}_{\mathcal{S}} := (|\{i \mid d_i = 0\}|, |\{i \mid d_i = 1\}|, \ldots, |\{i \mid d_i = \Delta\}|)$. Let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ and $\mathcal{S}' = \{d'_1, d'_2, \ldots, d'_n\}$ be two degree sequences with corresponding block sequences $\mathcal{B}$ and $\mathcal{B}'$. We define $\|\mathcal{B}\| := |\mathcal{S}| := \sum_{i=1}^{n} d_i$. We write $\mathcal{S}' \geq \mathcal{S}$ and $\mathcal{B}' \circledcirc \mathcal{B}$ 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 "$\mathcal{S}'$ can be obtained from $\mathcal{S}$ by increasing some values". If $\mathcal{S}' \geq \mathcal{S}$, then (for sorted degree sequences) we define the degree sequence $\mathcal{S}' - \mathcal{S} := \{d'_1 - d_1, d'_2 - d_2, \ldots, d'_n - d_n\}$ and set $\mathcal{B}' \ominus \mathcal{B}$ to be the block sequence of $\mathcal{S}' - \mathcal{S}$.

We say that the block i in the block sequence $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ *contains* b_i *degrees*. Furthermore, by *raising* x *degrees from block* i *to block* j for some $1 \leq x \leq b_i$, $0 \leq i < j \leq \Delta$, we mean to change the currently considered block sequence to $\mathcal{B}' := (b_0, b_1, \ldots, b_{i-1}, b_i - x, b_{i+1}, \ldots, b_{j-1}, b_j + x, b_{j+1}, \ldots, b_\Delta)$.

Given a block sequence $\mathcal{B}$, a *minimal* k-anonymous block sequence $\mathcal{B}' \circledcirc \mathcal{B}$ is a k-anonymous block sequence such that there is no k-anonymous block sequence $\mathcal{B}'' \neq \mathcal{B}'$ with $\mathcal{B}' \circledcirc \mathcal{B}'' \circledcirc \mathcal{B}$.

6.6.1. General Framework Description

We first recall the two-phase approach due to Liu and Terzi [LT08] (see Figure 6.4) and then describe how we refine it. To this end, let $(G = (V, E), k)$ be an input instance of Anonym E-Ins.

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

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

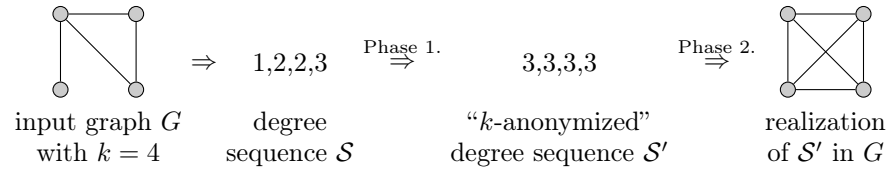


Figure 6.4.: A simple example for the two phases in the heuristic of Liu and Terzi [LT08]. Phase 1: Anonymize the degree sequence $\mathcal{S}$ of the input graph G by increasing the numbers in it such that each resulting number occurs at least k times. Phase 2: Realize the k-anonymized degree sequence $\mathcal{S}'$ as a super-graph of G.

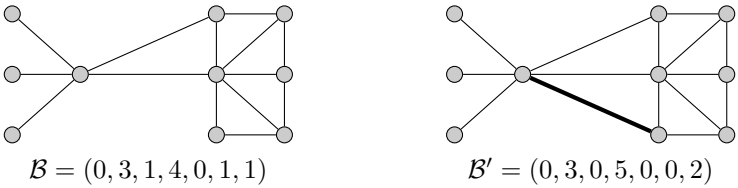


Figure 6.5.: A graph (left side) with block sequence $\mathcal{B}$ that can be 2-anonymized by inserting one edge (right side) resulting in $\mathcal{B}'$. Another 2-anonymous block sequence (also increasing two degrees by one) that will be found by our dynamic programming is $\mathcal{B}'' = (0, 2, 2, 4, 0, 0, 2)$. The realization of $\mathcal{B}''$ in G would require to insert an edge between a degree-five vertex (there is only one) and a degree-one vertex, which is impossible.

The minimum k-anonymization cost of $\mathcal{S}$, formally $|\mathcal{S}' - \mathcal{S}|/2$ (each inserted edge affects two vertices), is a lower bound on the number of edges in a k-insertion set for G as inserting any k-insertion set to G produces by definition a graph with a k-anonymous degree sequence. Hence, if succeeding in Phase 2 to realize $\mathcal{S}'$, then a minimum-size k-insertion set S for G has been found.

Liu and Terzi [LT08] gave a dynamic programming algorithm which optimally solves Phase 1 and they provided the so-called *local exchange heuristic* for Phase 2. If Phase 2 fails, then the heuristic of Liu and Terzi relaxes the model and tries to find a supergraph of G whose degree sequence is "close" to $\mathcal{S}'$.

First attempts and difficulties. We started with a straightforward implementation of the dynamic programming algorithm and the local exchange heuristic and encountered two major difficulties:

Figure 6.6.: A graph (left side) with block sequence $\mathcal{B} = (0, 7, 0, 0, 2, 1)$ that can be 2-anonymized by inserting two edges yielding a graph with block sequence $\mathcal{B}' = (0, 5, 2, 0, 0, 3)$ (right side). A minimum-cost solution for the number problem is, however, $\mathcal{B}'' = (0, 7, 0, 0, 0, 3)$. This solution would also pass our realizablity test, which simply checks whether $\mathcal{B}'' \ominus \mathcal{B}$ is a realizable block sequence: $\mathcal{B}'' \ominus \mathcal{B} = (8, 2)$ is realizable as graph with ten vertices and one edge.

1. Even when iterating through all minimum k-anonymous degree sequences, one often fails to realize them in Phase 2 (see Figure 6.5 for an example where two minimum k-anonymous degree sequences differ in their realizability and Figure 6.6 for an example where no minimum solution is realizable).

2. Iterating through all minimum sequences is often too time-consuming because the same sequence is recomputed multiple times.

To overcome difficulty 1, we improved the lower bound provided by $\mathcal{S}' - \mathcal{S}$ on the k-anonymization cost of G. To this end, the basic observation is that while trying to realize one of the minimum k-anonymous sequences $\mathcal{S}'$ in Phase 2 (failing in almost all cases), we encountered that $\mathcal{S}' - \mathcal{S}$ is not a realizable degree sequence. However, since $\mathcal{S}' - \mathcal{S}$ shall correspond to the graph induced by a k-insertion set, it is necessary for the realization of $\mathcal{S}'$ in Phase 2 that $\mathcal{S}' - \mathcal{S}$ is a realizable degree sequence. This observation allows us to apply a simple characterization due to Erdős and Gallai [EG60] about realizable degree sequences to $\mathcal{S}' - \mathcal{S}$ to rule out nonrealizable solutions produced by the dynamic programming. Thus, for increasing cost c, by iterating through all k-anonymous sequences $\mathcal{S}'$ with $|\mathcal{S}' - \mathcal{S}| = c$ and excluding the possibility that $\mathcal{S}'$ is not realizable in G by the criterion on $\mathcal{S}' - \mathcal{S}$, one can step by step 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 $\mathcal{S}'$ with $|\mathcal{S}' - \mathcal{S}| = c$. 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 $\mathcal{S}' - \mathcal{S}$ (see Figure 6.6 for an example). We thus designed a test which not only checks whether $\mathcal{S}' - \mathcal{S}$ is realizable but also takes already present edges in G into account while preserving that $|\mathcal{S}' - \mathcal{S}|$ is a lower bound on the k-anonymization cost of G. With this further requirement on the resulting sequences $\mathcal{S}'$ of Phase 1, we observed in our experiments that Phase 2 of realizing $\mathcal{S}'$ in G succeeds in 56 out of 58 cases. See Section 6.6.2 for a detailed description of our algorithm for Phase 1.

Phase 1. Although we could (almost) overcome difficulty 1, its solution requires to iterate over all solutions of a particular cost. Unsurprisingly, this had a severe impact on the running time. We thus redesigned the dynamic program for Phase 1 based on the following two observations.

First, we observed that the standard dynamic programming approach using the degree sequence produced many similar solutions (see also difficulty 2): If the degree of one degree-five vertex needs to be increased and there are 100 vertices with degree five, then all the 100 possibilities were considered. When using the block sequence instead of the degree sequence the above decision which degree-five-vertex gets a new neighbor is deferred to Phase 2. In this way the usage of the block sequence allows to store solutions in a compressed way which significantly speeds up Phase 1 but makes Phase 2 more difficult.

Our second observation was that many realizable solutions "contain" the same minimal solution for the number problem in Phase 1 that is not realizable: If, for example, the degree of a vertex v needs to be increased by 20, then this vertex clearly needs 20 further neighbors. If, however, the minimal solution only affects 10 vertices, then the dynamic program with the described characterization of Erdős and Gallai [EG60] will consider all possible combinations to choose 10 further vertices into the solution. To avoid considering this huge amount of similar solutions, we simply pick 10 vertices of degree one into the solution. Since social networks contain a large amount of degree one vertices, there are usually enough degree-one-vertices that are not adjacent to any vertex in the solution, hence picking the degree-one vertices is usually optimal. Combining the two mentioned ideas in the redesigned dynamic program dramatically reduces the number of considered solutions for the number problem and thus improves the running time of Phase 1.

Phase 2. For Phase 2 the task is to decide whether a given k-anonymous degree sequence $\mathcal{S}'$ can be realized in G. As we will show in Section 6.6.3 that

this problem is NP-complete, 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 $d'_i \in \mathcal{S}'$ to a vertex v in G such that $d'_i \geq \deg_G(v)$. Then, the demand of the vertex v is set to $d'_i - \deg_G(v)$. Note that the usage of block sequences makes this degree-vertex mapping necessary. 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 (see Section 6.3.1), we show that already a heuristic refinement of the local exchange heuristic due to Liu and Terzi [LT08] succeeds in most cases. Thus, theoretically and also in our experiments, the "hard part" is to find a good degree-vertex mapping. Roughly speaking, the difficulty is that, according to $\mathcal{S}'$, 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 Section 6.6.3 for a detailed description of our algorithm for Phase 2.

6.6.2. Phase 1: Exact k-Anonymization of Degree Sequences

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

Basic Number Problem

We now show how to solve the problem of making the degree sequence, respectively the block sequence k-anonymous. Recall that in order to avoid reconsidering similar solutions, we only want to compute *minimal* solutions for the number problem which we later try to realize in the graph. Thus, the degree sequence anonymization problem reads as follows.

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

Input:	**Solution:**
$\mathcal{B} = (0, 4, 1, 0, 1), k = 2, s = 2$	$\mathcal{B} = (0, 4, 0, 0, 2)$

Clarkson et al. [CLT10] gave a dynamic programming algorithm that solves a modified version of k-DSA which finds just one minimum-cost solution in $\mathcal{O}(n)$ time and space. Contrasting this modified version, the requirements on $\mathcal{B}'$ in the above problem definition ensure that the solution has cost *exactly* s, that is, $\mathcal{B}'$ can be obtained by performing s increases to the degrees in $\mathcal{B}$. Our dynamic program has two major differences to the one of Clarkson et al. [CLT10]:

1. We use block instead of degree sequences.

2. Our table has size $2k\Delta$ while their table has size n.

Before describing our dynamic programming algorithm, we provide a few useful observations about minimal solutions. First, observe that if a block i is empty, then it never makes sense to raise degrees to block i since raising them to block $i-1$ instead would yield a smaller solution.

Observation 6.25. *Let $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ be a block sequence, $k \in \mathbb{N}$, and let $\mathcal{B}' = (b'_0, b'_1, \ldots, b'_{\Delta'})$ be a minimal k-anonymous block sequence with $\mathcal{B}' \circledcirc \mathcal{B}$. If $b_i = 0$ for some $1 \leq i \leq \Delta$, then $b'_i = 0$ and hence $\Delta = \Delta'$.*

Similarly, if a block i is not empty in a minimal solution, then not all degrees of block i are raised as otherwise the block i would be empty in the minimal solution.

Observation 6.26. *Let $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ be a block sequence, $k \in \mathbb{N}$, and let $\mathcal{B}' = (b'_0, b'_1, \ldots, b'_{\Delta'})$ be a minimal k-anonymous block sequence with $\mathcal{B}' \circledcirc \mathcal{B}$. If $b'_i > 0$ for some $1 \leq i \leq \Delta$, then $\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell < b_i$.*

Here, $\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell$ is the number of degrees that are raised from the blocks $0, 1, \ldots, i$ to the blocks $i+1, i+2, \ldots, \Delta$ in the minimal solution $\mathcal{B}'$.

Next, observe that if a block i is not empty in a minimal solution, then at most $2k-1$ degrees were raised to the block i as otherwise raising k degrees to block $i-1$ and not to block i would yield a smaller solution. To this end, note that the term $b_i - (\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell)$ is the number of degrees that are raised from the block i in the minimal solution $\mathcal{B}'$.

Observation 6.27. *Let $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ be a block sequence, $k \in \mathbb{N}$, and let $\mathcal{B}' = (b'_0, b'_1, \ldots, b'_{\Delta'})$ be a minimal k-anonymous block sequence with $\mathcal{B}' \circledcirc \mathcal{B}$. If $b'_i > 0$ for some $1 \leq i \leq \Delta$, then $b'_i - (b_i - (\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell)) = \sum_{\ell=i}^{\Delta} b'_\ell - b_\ell < 2k$.*

Extending Observation 6.27 gives the following: if we increase $x \geq k$ degrees to a block i, then block i has size less than $2k$ in a minimum solution as otherwise increasing k degrees to the block $i-1$ and $x-k$ degrees to the block i would yield a smaller solution. Formalizing this extension, we arrive at the following.

Observation 6.28. *Let $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ be a block sequence, $k \in \mathbb{N}$, and let $\mathcal{B}' = (b'_0, b'_1, \ldots, b'_{\Delta'})$ be a minimal k-anonymous block sequence with $\mathcal{B}' \oplus \mathcal{B}$. If $b'_i \geq k$ for some $1 \leq i \leq \Delta$, then $b'_i - (b_i - (\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell)) \leq \max\{k, 2k - (b_i - (\sum_{\ell=i+1}^{\Delta} b'_\ell - b_\ell))\}$.*

The proof of the following lemma contains the detailed description of our dynamic programming algorithm for k-DSA.

Lemma 6.29. 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.*

Proof. Let $(\mathcal{B}, k, s)$ be an instance of Degree Sequence Anonymity. We describe a dynamic programming algorithm. It maintains a table T where the entry $T[i, t, c]$ with $0 \leq i \leq \Delta$, $0 \leq c \leq s$, and $0 \leq t < 2k$ is true if and only if for the block sequence $\mathcal{B}(i) = (b_0, b_1, \ldots, b_i)$ minus the last $t \leq \|\mathcal{B}(i)\|$ degrees there exists a minimal k-anonymous block sequence $\mathcal{B}'(i) \oplus \mathcal{B}(i)$ with $s = \|\mathcal{B}'(i) \ominus \mathcal{B}(i)\|$. Formally, $\mathcal{B}(i)$ *minus the last t degrees* is the block sequence $\mathcal{B}(i, t)$ corresponding to the degree sequence $\mathcal{S}$ that is obtained from $\mathcal{S}_{\mathcal{B}(i)}$ by removing the t highest degrees.

Starting with $c = 0$ we stepwise increase the costs until $T[\Delta, 0, c]$ is true. Considering the base case, $T[0, t, c]$ is true if and only if $c = 0$ and either $b_0 - t = 0$ or $b_0 - t \geq k$. The recursion for the computation of T is as follows: First, note that we decrease the cost c for each considered degree i by the amount t of vertices that are raised from $\mathcal{B}(i)$. Thus, when dealing with degree i, we only need to deal with the question of how many degrees are raised from $\mathcal{B}(i-1)$. If block i is an empty block, that is, $b_i = 0$, then, by Observation 6.25, it remains empty and we have $T[i, t, c] = T[i-1, t, c-t]$. Furthermore, if $b_i \leq t$, then, by Observation 6.26, the block i will be empty in the corresponding solution and thus $T[i, t, c] = T[i-1, t-b_i, c-(t-b_i)]$. Hence, it remains to consider the case that $b_i > t$, that is, the block i is not empty in a solution and thus contains at least k degrees. Hence, if $k > b_i - t$, then at least $k - (b_i - t)$ degrees needs to be raised to block i. By Observation 6.28, it follows that at most $\max\{k, 2k - (b_i - t)\}$ degrees are raised to block i. Hence, $T[i, t, c]$ is true if there exists some $\max\{0, k - (b_i - t)\} \leq t' \leq \max\{k, 2k - (b_i - t)\}$ such that $T[i-1, t', c-t'] = \text{true}$.

Summing up, we arrive at the following recursion:

$$T[i,t,c] = \begin{cases} c = 0 \wedge (b_0 - t = 0 \vee b_0 - t \geq k), & \text{if } i = 0 \\ \exists t' \in \mathbb{N}: & \\ \quad \max\{0, k - (b_i - t)\} \leq t' \leq \max\{k, 2k - (b_i - t))\} & \text{if } b_i > t \\ \quad \wedge\, T[i-1, t', c - t'] = \text{true}, & \\ T[i-1, t - b_i, c - (t - b_i)], & \text{if } b_i \leq t. \end{cases}$$

By the discussion above, it follows that the entry $T[\Delta, 0, s]$ is true if and only if $(\mathcal{B}, k, s)$ is a yes-instance. The computation of one table entry requires at most $2k$ table lookups and hence can be done in $\mathcal{O}(k)$ time. As there are $\Delta \cdot k \cdot s$ table entries, the overall running time is $\mathcal{O}(\Delta \cdot k^2 \cdot s)$. □

Recall that there might be multiple minimal solutions for a given k-DSA instance while only one of them is realizable, see Figure 6.5 for an example. Thus, having computed the table T, we iterate via standard traceback through the table to create all minimal solutions.

Criteria on the Realizability of k-DSA Solutions

A problem in the solutions provided by Phase 1 is the following: If a solution raises one degree by some amount, say 100, and the overall number of raised degrees is at most 100, then this solution cannot be realized as one of the vertices would require 100 further neighbors but only 99 further vertices get new neighbors. Lu et al. [LSB12] demonstrated that this effect appears also in real-world networks and we also frequently observed this effect in our experiments. We overcome this problem as follows: For a k-DSA-instance $(\mathcal{B}, k, s)$ 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 $\mathcal{B}' \ominus \mathcal{B}$. To test whether $\mathcal{B}' \ominus \mathcal{B}$ is realizable, we recall the characterization of realizable degree sequence due to Erdős and Gallai [EG60].

Theorem 3.2 (Erdős and Gallai [EG60]). *Let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ be a degree sequence, where $d_1 \geq d_2 \geq \ldots \geq d_n$. Then, $\mathcal{S}$ is realizable if and only if $\sum_{i=1}^{n} d_i$ is even and for all $r \in \{1, 2, \ldots, n-1\}$ it holds that*

$$\sum_{i=1}^{r} d_i \leq r(r-1) + \sum_{i=r+1}^{n} \min\{d_i, r\}. \tag{3.1}$$

As our first implementation of the Erdős-Gallai characterization was a bottleneck for the running time, we used the following result of Tripathi and Vijay [TV03] who showed that it is sufficient to check Inequality (3.1) only for values of r where $d_r > d_{r+1}$ (see also page 33).

Lemma 6.30 (Tripathi and Vijay [TV03]). *Let $\mathcal{S} = \{d_1, \ldots, d_n\}$ be a degree sequence sorted in descending order. Let q be the largest integer such that $d_q \geq q - 1$. Then $\mathcal{S}$ is realizable if and only if the sum of the d_i's is even and for each integer $r \in \{i \in \mathbb{N} \mid d_i > d_{i+1} \wedge i \leq q\}$ the following holds*

$$\sum_{i=1}^{r} d_i \leq r(r-1) + \sum_{i=r+1}^{n} \min(r, d_i). \tag{6.2}$$

Note that the indices r for which Inequality (6.2) in Lemma 6.30 have to be checked corresponds to the "end" of blocks and thus can be quickly accessed in block sequences. Thus, given a block sequence, we can decide whether it is realizable in $\mathcal{O}(\Delta^2)$ time. We call the characterization provided in Theorem 3.2 the *simple Erdős-Gallai test*. Applying the simple Erdős-Gallai test on the solutions provided by the dynamic program yields a significant improvement of the obtained lower bounds—in some of the real-world graphs the first solution of our dynamic program passing the simple Erdős-Gallai test was twice as large as the minimum solution for k-DSA (increase from ≈ 500 edges to $\approx 1,000$ edges).

Unfortunately, there are k-anonymous degree sequences $\mathcal{S}'$ passing the simple Erdős-Gallai test that are still not realizable in the input graph G (see Figure 6.6 for an example). We thus designed an advanced version of the Erdős-Gallai test that also takes the structure of the input graph into account. To explain the basic idea behind, we first discuss how Inequality (3.1) in Theorem 3.2 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 can be at most the number of edges (counting each edge once for each endpoint, that is, twice overall) that can be "obtained" by making V^r a clique (yielding

Figure 6.7.: A graph (left side) with block sequence $\mathcal{B} = (0, 7, 0, 0, 2, 1)$ that can be 2-anonymized by inserting two edges yielding a graph with block sequence $\mathcal{B}' = (0, 5, 2, 0, 0, 3)$ (right side). A minimum-cost solution for the number problem is, however, $\mathcal{B}'' = (0, 7, 0, 0, 0, 3)$. This solution would also pass the Erdős-Gallai test as $\mathcal{B}'' \ominus \mathcal{B} = (8, 2)$ is realizable as graph with nine vertices and one edge.

the term $r(r-1)$) plus 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 simple 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 raised degrees can be uniquely determined, to subtract the corresponding number of edges as they cannot contribute to the right-hand side of Inequality (3.1).

Consider the example given in Figure 6.7: Running the dynamic program on the block sequence $\mathcal{B} = (0, 7, 0, 0, 2, 1)$ would first give the block sequence $\mathcal{B}' = (0, 7, 0, 0, 0, 3)$, that is, the two vertices of degrees four require each one more neighbor. Since the input graph contains just two degree-four vertices which are already adjacent, the advanced Erdős-Gallai test would return that $\mathcal{B}'$ is not realizable. As $\mathcal{B}'$ is the only solution of cost two, this would allow the program to increase the cost and to find the solution depicted on the right side in Figure 6.6. Note that a solution passing the advanced Erdős-Gallai test might still not be realizable as the vertices can often not be uniquely determined. For example, if a solution requires that three degree-five vertices need further neighbors but the input graph contains seven degree-five neighbors, it is not clear which degree-five vertices to choose. Indeed, we later show that the realization problem in Phase 2 is NP-complete (see Theorem 6.31). Thus, unless P = NP, there is no simple characterization like in Theorem 3.2 for the realizability of a solution in the input graph.

The difference between using just the simple Erdős-Gallai test and using the advanced Erdős-Gallai test resulted in rather small differences for the lower bound (at most 10 edges). This small difference was, however, important for some of our instances to succeed in Phase 2 and to optimally solve the instance.

Data Reduction Rule

In our preliminary experiments we observed that for some instances we could not finish Phase 1 even for $k = 2$ within a time limit of one hour. Our investigations revealed that this was mainly due to the frequent occurrence of the following "pattern" within three consecutive blocks: The first block i and the third block $i + 2$ are each of size at least $2k - 1$ and the middle block has size $k - 1$. For example, for $k = 2$ consider the consecutive blocks $4, 1, 4$. The details of the dynamic program (see Observation 6.28) show that in any solution for the entire block sequence the blocks i and $i + 2$ stay full and either the block $i + 1$ is filled by the degrees from block i or the degrees from block i are increased to block $i + 2$. In our example this means that the solution is either $4, 0, 5$ or $3, 2, 4$. Then, if this pattern repeats, say x times, then there are 2^x different solutions in Phase 1. However, in our example, both solutions $4, 0, 4$ and $3, 2, 4$ are equivalent with respect to the simple Erdős-Gallai test because they both increase just one degree by one. We thus designed a data reduction rule to deal with these patterns, where the first and last block are "large" enough to guarantee that the degrees of preceding blocks are not increased to the middle of the pattern and it is not necessary to increase something from the middle of the pattern to succeeding blocks. Hence, the middle of the pattern can be solved "independently" from preceding and succeeding blocks and if there is a minimum solution which is "Erdős-Gallai-optimal" (increasing degrees by at most one), then it is safe to take this solution for the middle of the pattern. Formally, our data reduction rule, generalizing the above ideas, is as follows.

Reduction Rule 6.1. *Let $(\mathcal{B}, k, s)$ be an instance of k-DSA. If there is a block i in $\mathcal{B}$ with $b_i \geq 2k - 1$, a sequence of blocks $j, j + 1, \ldots, j + t$ such that $\sum_{\ell=j}^{j+t} b_\ell \geq (t+1)k+k-1$ and $b_\ell \geq k$ for all $\ell \in \{j, j+1, \ldots, j+t\}$, and if there is a minimum-cost k-anonymization $\mathcal{B}'_{i,j}$ of the block sequence $\mathcal{B}_{i,j} := (b_i, b_{i+1}, \ldots, b_j)$ such that*

i) all blocks ℓ for $\ell \geq 2$ in $\mathcal{B}'_{i,j} \ominus \mathcal{B}_{i,j}$ are empty and

ii) the first block in $\mathcal{B}'_{i,j}$ is of size at least k,

then substitute in $\mathcal{B}$ the subsequence $\mathcal{B}_{i,j}$ by $\mathcal{B}'_{i,j}$ and reduce s by $\|\mathcal{B}'_{i,j} \ominus \mathcal{B}_{i,j}\|$.

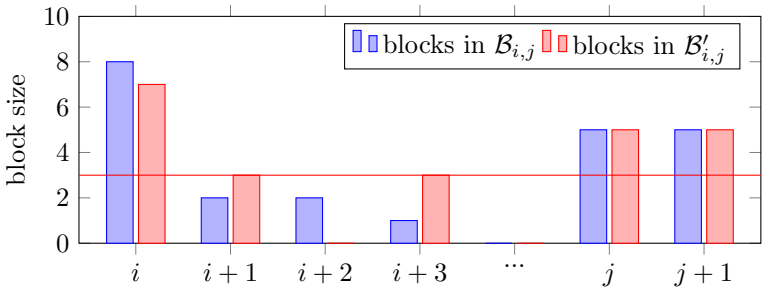


Figure 6.8.: Example for the application of Reduction Rule 6.1 for $k = 3$. For each degree d marked at the x-axis the left bar denotes the block d in $\mathcal{B}_{i,j}$ and the right in $\mathcal{B}'_{i,j}$. The horizontal line denotes the anonymity level $k = 3$. In this example we have $\mathcal{B}'_{i,j} \ominus \mathcal{B}_{i,j} = (0, 3, 0, 0, \ldots, 0)$ as one degree gets raised from degree i to $i+1$ and two degrees from $i+2$ to $i+3$.

See Figure 6.8 for an example of the application of Reduction Rule 6.1.

In our implementation of Reduction Rule 6.1 we use our dynamic programming algorithm (with disabled Erdős-Gallai tests) to check whether there is a k-anonymization $\mathcal{B}'_{i,j}$ for $\mathcal{B}_{i,j}$ fulfilling the required properties.

Although Reduction Rule 6.1 keeps solutions that are optimal with respect to the simple Erdős-Gallai test, some realizable solutions might get skipped when applying Reduction Rule 6.1. Thus, if Reduction Rule 6.1 is applied, then we have to impair the advanced Erdős-Gallai test such that it ignores the degrees that are affected by Reduction Rule 6.1.

Summarizing, the application of Reduction Rule 6.1 significantly accelerates our program at the cost of obtaining a slightly weaker lower bound. Notably, there exists exactly one instance (coAuthorsDBLP with $k = 2$) which we could solve optimally without the data reduction but we could not solve optimally with the data reduction.

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:

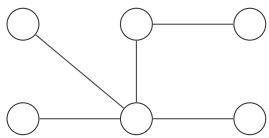
Algorithm 6.3: Pseudocode of our algorithm solving Phase 1

Input: An undirected graph G, its block sequence $\mathcal{B}$, and $k \in \mathbb{N}$.
Output: A set of minimum k-anonymous block sequences $\mathcal{B}'$ such that $\mathcal{B}' \circledcirc \mathcal{B}$ and $\mathcal{B}' \ominus \mathcal{B}$ is realizable.

```
Exhaustively apply Reduction Rule 6.1 on B
c ← 0; upperBound ← ∞; S_upper ← ∅
repeat
    compute table T with cost c according to Lemma 6.29
    if T[Δ, 0, c] = true then
        S ← all solutions of cost exactly c        // computed via traceback in T
        foreach solution B' ∈ S do
            while B' ⊖ B does not pass the advanced Erdős-Gallai test do
                waste costs to increase ‖B'‖                    // see Section 6.6.2
            if ‖B'‖ < upperBound then
                upperBound ← ‖B'‖                // current upper bound improved
                S_upper ← ∅
            if ‖B'‖ = upperBound and |S_upper| < 200 then
                // add B' to the set of maintained upper bounds
                S_upper ← S_upper ∪ {B'}
    c ← c + 1
until c = upperBound
return S_upper
```

Realizable Degree Sequence Anonymity (k-RDSA)
Input: An undirected graph G, its block sequence $\mathcal{B}$, and an integer $k \in \mathbb{N}$.
Task: Compute all k-anonymous block 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 in G.

Input:



$\mathcal{B} = (0, 4, 1, 0, 1), k = 2$

Solution:

$\mathcal{B}' = (0, 2, 2, 0, 2)$

Our strategy to solve k-RDSA is as follows (see Algorithm 6.3 for the pseudocode): We first reduce the given block sequence with respect to Reduction Rule 6.1 (see Line 1). We then iterate for increasing cost through the solutions of k-DSA (see Lines 3 to 6) and run for each of them the advanced Erdős-Gallai test (see Line 8). If the test fails, then we compute how many degrees have to be "wasted" in order to get a sequence passing the test (see Line 9). Wasting means to greedily increase some degrees in $\mathcal{B}'$ (while preserving k-anonymity) until the resulting block sequence passes the advanced Erdős-Gallai test. 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 i and $i+1$ containing many thousands of vertices. Hence, "wasting" edges is easy to achieve by increasing degrees from block i by one to block $i+1$. This is, like the application Reduction Rule 6.1, optimal with respect to the simple Erdős-Gallai characterization, but may destroy realizable solutions. 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 degrees to obtain a realizable sequence. In this way, we always produce a block sequence $\mathcal{B}'$ such that $\mathcal{B}' \ominus \mathcal{B}$ passes the advanced Erdős-Gallai test. If $\mathcal{B}'$ is smaller than a maintained upper bound, then this maintained upper bound is replaced by $\mathcal{B}$ (see Lines 10 to 14). If $\mathcal{B}'$ is of the same cost as the maintained upper bound, then it is added to it (see Lines 13 and 14). However, since for some instances more than 100,000 solutions are stored in this way and trying to realize them all lasts several hours, we keep at most 200 solutions and discard the remaining ones. When the considered costs match the maintained upper bound, then the solutions stored in the upper bound are the result of our heuristic.

Remark. Since we discard many solutions of k-RDSA with Reduction Rule 6.1, with our approach of wasting costs, and with the limit on the number of stored solutions, we might miss realizable solutions with our program. Furthermore, due to Reduction Rule 6.1 and the wasting approach, the advanced Erdős-Gallai test is weakened and, thus, also the lower bound is weakened. We emphasize that the stored lower bound in the dynamic program is, however, still a lower bound for the corresponding Anonym E-Ins instance. Furthermore, our experimental evaluation shows that the obtained lower bounds mostly deliver optimal solutions.

Further acceleration of Algorithm 6.3. The Erdős-Gallai test is designed to detect solutions of the number problem that are not realizable (see Line 8). We also use the Erdős-Gallai characterization to detect and prune in the traceback phase branches that will lead to solutions that are larger than our maintained upper bound.

For the further argumentation we need some notation. Denote by $\mathrm{diff}(r, \mathcal{S})$ the difference between the left-hand side and the right-hand side of Inequality (3.1) considering the degree sequence $\mathcal{S}$, that is,

$$\mathrm{diff}(r, \mathcal{S}) := \sum_{i=1}^{r} d_i - r(r-1) - \sum_{i=r+1}^{n} \min\{d_i, r\}.$$

Observe that if the considered degree sequence $\mathcal{S}$ is not realizable, then for some $1 \leq r \leq n$ it holds that $\mathrm{diff}(r, \mathcal{S}) > 0$. Denote by $r_{\max}$ the value maximizing $\mathrm{diff}(r, \mathcal{S})$, that is, $r_{\max} := \arg\max_{1 \leq r \leq n}\{\mathrm{diff}(r, \mathcal{S})\}$. Since for $r := r_{\max}$ the difference of the left-hand side of Inequality (3.1) to the right-hand side is $\mathrm{diff}(r_{\max}, \mathcal{S})$, it follows that for any realizable degree sequence $\mathcal{S}'$ with $\mathcal{S}' \geq \mathcal{S}$ it holds that $|\mathcal{S}' - \mathcal{S}| \geq \mathrm{diff}(r_{\max}, \mathcal{S})$. Hence, $\mathrm{diff}(r_{\max}, \mathcal{S})$ is a lower bound on the number of degrees that have to be wasted in order to get a realizable block sequence. Thus, if for the degree sequence $\mathcal{S}_{\mathcal{B}'}$ of the block sequence considered in Line 8 it holds that $\mathrm{diff}(r_{\max}, \mathcal{S}_{\mathcal{B}'}) + |\mathcal{S}_{\mathcal{B}'} - \mathcal{S}_{\mathcal{B}}|$ is larger than the stored upper bound, then we can skip $\mathcal{S}_{\mathcal{B}'}$ and continue with the next solution. Here, $\mathcal{S}_{\mathcal{B}}$ denotes the degree sequence of the input block sequence $\mathcal{B}$.

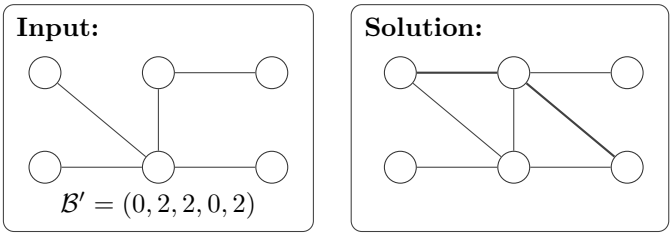
We extended the above idea to further reduce the amount of considered solutions for the k-DSA problem. We implemented a test in the traceback in Line 6 checking whether there is a chance that the current *partial* solution can be extended to a solution that is not larger than the maintained upper bound. If this test fails, then we can abort this branch in the traceback saving a lot of time by discarding multiple solutions at once. This test works as follows: During the traceback the table T is traversed starting from degree Δ in a recursive fashion. When considering the degree i, $0 \leq i \leq \Delta$, we compute the partial solution corresponding to the changes we made in the blocks $i+1, i+2, \ldots, \Delta$. This partial solution is a block sequence $\mathcal{B}' \ominus \mathcal{B}$, where $\mathcal{B}$ is the block sequence provided in the input. We then consider the degree sequence $\mathcal{S}$ corresponding to $\mathcal{B}' \ominus \mathcal{B}$ and compute $\mathrm{diff}(r_{\max}, \mathcal{S})$. If $\mathrm{diff}(r_{\max}, \mathcal{S}) + \|\mathcal{B}' \ominus \mathcal{B}\|$ is larger than the maintained upper bound, then we can prune the current branch in the traceback. Observe that social networks, due to their power-law degree distribution, often contain only few high-degree vertices. Hence, there are often

only a few different possibilities to anonymize these high-degree vertices and in each of these possibilities the raises of the degrees are rather large. This explains the effectiveness of the described test in our experiments on real-world data.

6.6.3. Phase 2: Realizing a k-Anonymous Degree Sequence

Let (G, k) be an instance of Anonym E-Ins 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, we face the following problem.

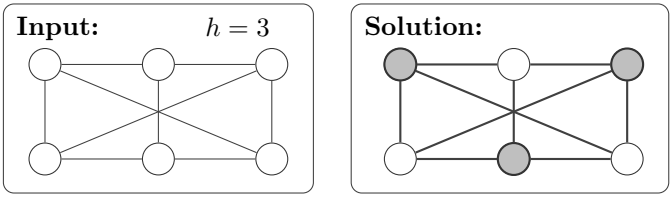
Degree Realization
Input: An undirected graph $G = (V, E)$ and a block sequence $\mathcal{B}'$.
Question: Is there a set $S \subseteq \binom{V}{2} \setminus E$ of edges such that $G + S$ has block sequence $\mathcal{B}'$?



Theorem 6.31. Degree Realization *is NP-complete even on cubic planar graphs.*

Proof. As the containment in NP is obvious, we focus on showing the NP-hardness by a reduction from the Independent Set problem:

Independent Set [GJ79, GT20]
Input: An undirected graph $G = (V, E)$ and a positive integer h.
Question: Is there an independent set $V' \subseteq V$ of size $|V'| \geq h$, that is, a vertex subset of pairwise nonadjacent vertices?



Independent Set remains NP-complete in cubic planar graphs [GJ79, GT20].

The reduction, which is similar to those proving that Anonym E-Ins remains NP-complete on three-colorable graphs (see Theorem 6.1), is as follows: Let G be a cubic planar and h be an integer that together form an instance of Independent Set. The block sequence of the n-vertex graph G is $\mathcal{B} = (0, 0, 0, n)$. We set

$$\mathcal{B}' = (0, 0, 0, n - h, \underbrace{0, 0, \ldots, 0}_{h-2}, h),$$

that is, the target graph is required to have $n - h$ vertices of degree three and h vertices of degree $h + 2$.

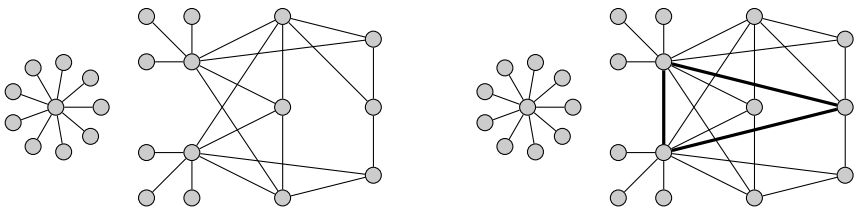
It remains to prove that the Degree Realization-instance $(G, \mathcal{B}')$ is a yes-instance if and only if (G, h) is a yes-instance for Independent Set.

If there is an independent set S (pairwise non-adjacent vertices) of size h in G, then inserting all edges between the vertices in S (making them a clique) results in a graph whose block sequence is $\mathcal{B}'$. Conversely, in a realization $G + S$ of $\mathcal{B}'$, there are exactly h vertices whose degree has been increased by $h - 1$. Hence, these vertices form a clique in $G + S$, implying that they are independent in G. □

We remark that from the proof of Theorem 6.1, it follows that Degree Realization is NP-complete even if $\mathcal{B}'$ is a k-anonymized sequence such that $\|\mathcal{B}' - \mathcal{B}\|$ is minimum.

We next present our heuristics for solving Degree Realization. We split the problem into two parts and then solve each part independently. First, we find a degree-vertex mapping, that is, for $\mathcal{S}' = \{d'_1, d'_2, \ldots, d'_n\}$ 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 $\kappa(v)$ to be the *demand* of v, that is, $\kappa(v) := d'_i - \deg_G(v)$. Second, we try to find, mainly by the local exchange heuristic [LT08], an edge insertion set S such that in $G + S$ each vertex v is incident to $\kappa(v)$ edges in S. The details in the proof of Theorem 6.31 indeed show that already finding a realizable degree-vertex mapping is NP-complete. 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. See Section 6.6.3 for the details of Phase 2.1.

The second part of deciding whether a degree-vertex mapping is realizable can be done in polynomial time by solving an f-Factor instance; see Section 3.2.1



$\mathcal{B} = (0, 15, 0, 3, 2, 1, 0, 2, 0, 1)$ $\mathcal{B}' = (0, 15, 0, 2, 2, 2, 0, 0, 0, 3)$

Figure 6.9.: Our smallest example where "jumps" are necessary to obtain a minimum-cost solution. The only minimum solution to 2-anonymize the left graph is to insert the bold edges in the right graph. Observe that, although there are two degree-four vertices in the left graph, the solution lifts a degree-three vertex to degree five, that is, there is a "jump" of two.

for a formal definition and Section 6.3.1 for the details on how to use the f-Factor problem. However, our preliminary experiments have shown that already a simple heuristic is able to find in almost all cases a realization of a degree-vertex mapping (if possible). We thus concentrated on finding "promising" degree-vertex mappings and implemented a (slightly enhanced) version of the local exchange heuristic for the second part of realizing them. See Section 6.6.3 for the details of Phase 2.2.

Phase 2.1: Finding a Degree-Vertex Mapping

Given an undirected graph G with its block sequence $\mathcal{B}$ and a k-anonymous block sequence $\mathcal{B}' \circledcirc \mathcal{B}$, two difficulties arise when trying to find a best possible (realizable) degree-vertex mapping for $\mathcal{B}'$. The first difficulty is rather obvious: Consider to 2-anonymize a graph consisting of two connected components $\{a, b, c\}$ and $\{d, e\}$ where each component is just a path. Hence, the block sequence is $\mathcal{B} = (0, 4, 1)$ and a minimum solution would be to insert an edge between two degree-one vertices, resulting in the block sequence $\mathcal{B}' = (0, 2, 3)$. Given $\mathcal{B}'$, a degree-vertex mapping has to choose two degree-one vertices where all but the choice $\{d, e\}$ lead to a realization. Hence, the basic problem is that a degree-vertex mapping has to choose x many vertices from block i which is of size more than x and thus the assignment is non-unique. In our experiments we

observed that this difficulty can be solved satisfactorily by randomly selecting the vertices from the blocks.

The second difficulty is, however, a more severe problem, also on real-world instances. Assume that $\mathcal{B} = (3, 2, 1)$ is the block sequence of our input graph (three degree-zero vertices and a path of length two) and the result of Phase 1 is the 2-anonymized block sequence $\mathcal{B}' = (2, 2, 2)$. Now, the difficulty arises that there are actually two "interpretations" of $\mathcal{B}'$: The first (natural) one would be to increase a degree zero up to one and a degree two up to three. However, the second would be that one degree zero is increased by two up to three. We call this a *jump* since a degree is increased "over" a non-empty block, while the natural interpretation (making most sense in the majority of the cases) is that a degree is increased to the next non-empty block B and from there, first the vertices originally in B are increased further. While in the example above the second "jump"-interpretation cannot be realized (only one vertex has non-zero demand), Figure 6.9 illustrates an example where the only realizable degree-vertex mapping has such a jump.

In our experiments, against our a-priori intuition, we observed that the k-anonymized sequences $\mathcal{B}'$ (computed in Phase 1) have typically less than ten "jump blocks" (a jump over these blocks is possible) and for each of these blocks up to five degrees can jump "over" it. Since the number of jump blocks is reasonably small and as we try to realize many degree-vertex mappings for each $\mathcal{B}'$, we iterate for increasing α through all possibilities to choose α. Having fixed the jumps, it follows how many degrees from i are increased to j and we randomly select the appropriate number of vertices from block i in G.

Phase 2.2: Realizing a Degree-Vertex Mapping

In the last part of finding a realization of a k-anonymized sequence $\mathcal{B}'$ in a graph $G = (V, E)$, one is given a degree-vertex mapping which provides a non-negative integer demand for each vertex and the task is to decide whether it is realizable, that is, is there an edge insertion set S such that in $G + S$ the amount of incident new edges for each vertex is equal to its demand. Formally, let $\kappa\colon V \to \mathbb{N}$ be the function providing the demand of each vertex. Whether κ is realizable can be decided in polynomial time by solving an f-Factor instance and it has been shown that, for the maximum degree Δ of G, κ is always realizable if $\sum_{v \in V} \kappa(v) \geq (\Delta^2 + 4\Delta + 3)^2$ and $\max_{v \in V}\{\kappa(v)\} \leq \Delta + 2$ (see Lemma 6.8). We have implemented the local exchange heuristic by Liu and Terzi [LT08] which turned out to perform surprisingly well. Indeed, in the next

theorem we present also some theoretical justification for this, formally proving that roughly the same lower bound (as for f-FACTOR) on $\sum_{v \in V} \kappa(v)$ is enough to guarantee that the local exchange heuristic always realizes κ.

In principle, the local exchange heuristic inserts edges between vertices as long as possible to satisfy their demand and if it gets stuck at some point, then it tries to continue by exchanging an already inserted edge. Formally, it works as follows: Let S be the set of new edges which is initialized by $\emptyset$. As long as there are two vertices u and v with non-zero demand, check whether the edge $\{u, v\}$ is insertable, meaning that neither $\{u, v\} \in E$ nor $\{u, v\} \in S$. If it is insertable, then insert $\{u, v\}$ to S and decrease the demand of u and v by one. If this procedure ends with all vertices having demand zero, then S is an insertion set realizing κ. Otherwise, we are left with a set V^κ of vertices with non-zero demand. If there are two vertices $v_1, v_2 \in V^\kappa$, then for each edge $\{u, w\} \in S$ check whether the two edges $\{v_1, u\}$ and $\{v_2, w\}$ or $\{v_1, w\}$ and $\{v_2, v\}$ are insertable. If so, then delete $\{u, w\}$ from S, insert the two edges that are insertable, and decrease the demand of v_1 and v_2 by one. In the special case of V^κ containing only one vertex v, then it holds that the remaining demand of v is at least two, because $\sum_{v \in V} \kappa(v)$ can be assumed to be even (otherwise it is not realizable). In this case perform the following for each edge $\{u, w\} \in S$: Check whether $\{v, u\}$ and $\{v, w\}$ are insertable and if so, then insert them to S, delete $\{u, w\}$ from S, and decrease the demand of v by two.

We have implemented the local exchange heuristic so that it first randomly tries to insert edges and then, if stuck at some point, performs the above described exchange operations (if possible). We conclude with proving a certain lower bound on $\sum_{v \in V} \kappa(v)$ which guarantees the success of the local exchange heuristic. To this end, recall that by Corollary 6.6 we may assume that the maximum degree is not increased by more than $\Delta^2 + 5\Delta + 2 - \Delta$.

Theorem 6.32. *Let $G = (V, E)$ be an undirected graph with maximum degree Δ and let $\kappa : V \to \mathbb{N}$ be a demand function such that $\max_{v \in V} \kappa(v) + \deg_G(v) \leq \Delta^2 + 5\Delta + 2$. The local exchange heuristic always realizes κ if $\sum_{v \in V} \kappa(v) \geq 6(\Delta^2 + 5\Delta + 3)^2$.*

Proof. Towards a contradiction, assume that the local exchange heuristic gets stuck at some point such that no edge is insertable and no further exchange operation can be performed. Denote by V^κ the set of vertices still having a non-zero demand in V^κ. Let S be the set of new edges already inserted at this point. We show that if one of V^κ and S is "large", then the heuristic will

continue working. We then prove that at least one of the two following cases applies.

Case 1 $|V^\kappa| \geq \Delta^2 + 5\Delta + 4$

In this case consider any vertex $v \in V^\kappa$ and observe that it cannot have more than $\Delta^2+5\Delta+2$ neighbors in $G+S$. Hence, there is a vertex $u \in V^\kappa$ such that $\{v, u\}$ is insertable.

Case 2 $|S| > 2(\Delta^2 + 5\Delta + 2)^2$

Consider first the subcase where V^κ consists only of one vertex v. Hence, the demand of v is at least two. However, since v can have at most $\Delta^2 + 5\Delta+2$ neighbors in $G+S$ and each of this neighbors has at most $\Delta^2+5\Delta+2$ incident edges in S, it follows from $|S| > 2(\Delta^2+5\Delta+2)^2 > (\Delta^2+5\Delta+2)^2$ that there is at least one edge $\{u, w\} \in S$ such that v is neither a neighbor of u nor of w. Thus, the exchange operation (inserting $\{u, v\}$ and $\{v, w\}$ and deleting $\{u, w\}$) can be applied.

In the last subcase assume that there are two vertices $v_1, v_2 \in V^\kappa$. For the vertex v_1 (v_2) it holds that there are less than $(\Delta^2 + 5\Delta + 2)^2$ edges in S which contain a neighbor of v_1 (v_2, resp.). Hence, from $|S| > 2(\Delta^2 + 5\Delta + 2)^2$ it follows that there is an edge $\{u, w\}$ in S where both u and w are non-adjacent to each of $\{v_1, v_2\}$ and thus the exchange operation can be applied. This completes Case 2.

Assume that Case 1 does not apply as this would contradict our assumption. Thus, since the demand of each vertex in V^κ is at most $\Delta^2 + 5\Delta + 2$, it follows that

$$\begin{aligned}
|S| &\geq \frac{\sum_{v \in V} \kappa(v)}{2} - |V^\kappa| \cdot (\Delta^2 + 5\Delta + 2) \\
&> \frac{6(\Delta^2 + 5\Delta + 3)^2}{2} - (\Delta^2 + 5\Delta + 3)^2 \\
&= 2(\Delta^2 + 5\Delta + 3)^2 > 2(\Delta^2 + 5\Delta + 2)^2.
\end{aligned}$$

Hence, Case 2 applies and this causes a contradiction to the assumption that the local exchange heuristic got stuck at some point. □

6.6.4. Further Upper-Bound Heuristics

While the focus of the algorithm framework presented in Sections 6.6.1 to 6.6.3 is on obtaining good lower bounds for a minimum-size k-insertion set and trying

to realize the lower bounds in the graph, we present in this section two upper bound heuristics that are based on our algorithm framework. The heuristics follow the two-phase approach by first solving the associated number problem on the degree sequence and then try to realize the computed k-anonymous degree sequence. Both heuristics use the same realization strategy: keep wasting costs until the realization algorithm succeeds. Observe that we use a similar strategy in Line 8 of Algorithm 6.3 to obtain a block sequence passing the advanced Erdős-Gallai test. In fact, we reuse the source code computing a "promising way" of wasting costs (see Section 6.6.2 for further details). We further reuse the realization algorithm for Phase 2 (see Section 6.6.3 for further details) in both heuristics. Hence, the only difference between the two heuristics is the computation of the k-anonymous degree sequence.

The first heuristic, called *Greedy Heuristic*, employs a greedy strategy to heuristically compute a k-anonymous degree sequence. The second heuristic, called *DP Based Heuristic*, uses a slightly improved version of a simple dynamic program due to Clarkson et al. [CLT10]. Note that Clarkson et al. provided a faster but more complicated dynamic program. However, since our implementation of the simple dynamic program could solve all tested instances within seven seconds, we did not implement the complicated one. We now give a detailed description how the two heuristics compute a k-anonymous degree sequence.

Greedy heuristic. Our first heuristic computes the block sequence $\mathcal{B} = (b_0, b_1, \ldots, b_\Delta)$ of the input graph and then step by step computes a k-anonymous block sequence $\mathcal{B}' = (b'_0, b'_1, \ldots, b'_\Delta)$: Starting with degree Δ the heuristic iterates for decreasing degree d over $\mathcal{B}$ and after each iteration loop the considered block d is either empty or full, that is, $b'_d = 0$ or $b'_d \geq k$. Thus, when considering degree d, the blocks $d+1, d+2, \ldots, \Delta$ are already fixed. If the block d is already empty or full, then the heuristic continues with block $d-1$. Otherwise, the heuristic considers two possibilities: either all degrees in block d are raised to the next full block, or the block d gets filled. To this end, denote by d' the degree of the next full block, that is, for each $d < i < d'$ we have $b'_i = 0$. Furthermore let ℓ be the largest number such that $\sum_{i=\ell}^{d} b_i > k$, that is, when filling the block d, then degrees from the blocks $\ell, \ell+1, \ldots, d-1$ are raised to block d. If $d' - d < d - \ell$ and it is cheaper to raise all degrees from block d than to fill block d, then the heuristic raises the degrees in the block d to block d' and the heuristic continues with block $d-1$. Otherwise, it raises degrees from the blocks $\ell, \ell+1, \ldots, d-1$ to block d. If $\sum_{i=\ell}^{d} b_i \geq 2k$, then it holds that $b_\ell \geq k$ and the heuristic will

raise degrees such that the block d contains exactly k degrees and the block ℓ contains at least k degrees. Otherwise, the heuristic raises all degrees from the blocks $\ell, \ell+1, \ldots, d-1$ to block d. After raising degrees to block d, the blocks $\ell+1, \ell+2 \ldots, d-1$ are empty and ℓ is either empty or full. Hence, the heuristic continues with degree $\ell-1$. Finally, let us mention two special cases: If at some point the blocks $0, 1, \ldots d$ contain together less than k degrees, then all degrees in these block are raised to the next full block. If there is no next full block, that is, when $d = \Delta$, then the block Δ gets filled.

DP based heuristic. We briefly show the dynamic program of Clarkson et al. [CLT10] and then explain our modification to it. To this end, let $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ be the degree sequence of the input graph. We denote for $1 \leq i < j \leq n$ with $\text{cost}[i,j]$ the cost of increasing the degrees $d_i, d_{i+1}, \ldots, d_j$ to d_j, that is $\text{cost}[i,j] := \sum_{\ell=i}^{j} d_j - d_\ell$. The dynamic program uses a one-dimensional table T, where for $1 \leq i \leq n$ the entry $T[i]$ stores the minimum cost of making the degree sequence $\{d_1, d_2, \ldots, d_i\}$ k-anonymous. Thus, $T[n]$ stores the cost of making $\mathcal{S}$ k-anonymous. Clarkson et al. [CLT10] showed that T can be computed with the following recurrence:

$$T[i] = \begin{cases} \infty, & \text{if } i < k \\ \text{cost}[1,i], & \text{if } k \leq i < 2k \\ \min\limits_{i-2k+1 \leq t \leq i-k} \{\text{cost}[t+1,i] + T[t]\}, & \text{if } i \geq 2k \end{cases}$$

With this recursion, the table T can be computed in $\mathcal{O}(nk)$ time as each table entry can be computed with at most k table lookups.

Our heuristic uses this dynamic program on a preprocessed degree sequence $\mathcal{S}'$. To this end, let $\mathcal{B}$ the block sequence of the input graph. By Observation 6.27, we raise in a minimum solution at most $2k-1$ degrees from each block. Thus, if a block i contains more than $3k$ degrees, then at least k degrees remain in a minimum solution in block i. Let $\mathcal{B}'$ be the block sequence obtained from $\mathcal{B}$ by setting $b'_i = \min\{3k, b_i\}$ for each $1 \leq i \leq \Delta$, that is, we keep at most $3k$ degrees from each block. The heuristic runs the above dynamic program on the degree sequence $\mathcal{S}'$ corresponding to $\mathcal{B}'$. Note that $\mathcal{S}'$ is not necessarily realizable as graph. However, we can transfer any minimum solution for $\mathcal{S}'$ to $\mathcal{S}$: Let $\mathcal{S}'^{\min} \geq \mathcal{S}'$ be a minimum k-anonymous degree sequence and let $\mathcal{B}'^{\min} = (b'^{\min}_0, b'^{\min}_1, \ldots, b'^{\min}_\Delta)$ the corresponding block sequence. We construct $\mathcal{B}^{\min} = (b^{\min}_0, b^{\min}_1, \ldots, b^{\min}_\Delta)$ by setting $b^{\min}_i := b'^{\min}_i + (b_i - b'_i)$ for

each $1 \leq i \leq \Delta$. Note that if $b_i \neq b'_i$, then $b_i > 3k$ and $b'_i = 3k$. Since at most $2k-1$ degrees are raised from each block in a minimum solution, it follows that if $b_i \neq b'_i$, then ${b'_i}^{\min} > k$. Thus, the block sequence $\mathcal{B}^{\min}$ is k-anonymous and it holds that $\mathcal{B}^{\min} \circledcirc \mathcal{B}$. Furthermore, from the minimality of $\mathcal{B}'^{\min}$, it follows that $\mathcal{B}^{\min}$ is a minimum k-anonymous block sequence.

6.6.5. Experimental Evaluation

We implemented the lower- and upper-bound heuristics described in Sections 6.6.1 to 6.6.4. Here we present the results of our experimental evaluation.

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_55. The source code is freely available.[2]

We tested each graph for $k = 2, 3, 4, 5, 7, 9, 10, 15, 20, 30, 50, 100, 200$. The time limit for one instance (one graph for one value of k) is set to one hour for the lower bounds setup. 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 time limit for the upper bounds heuristics is five minutes.

We compared the results of our upper bound heuristics against an implementation of the so-called *clustering heuristic* due to Lu et al. [LSB12] and against the lower bounds given by our dynamic program.

Real-world data sets. We use datasets from the following four categories for our experimental evaluations. Table 6.2 shows the size of each considered graph and its respective maximum degree.

LWA The website of the Laboratory for Web Algorithmics of the university of Milano [Bol+] provides huge graphs with up to 100 billion vertices. We took three of their "small" graphs that still contain several millions of edges.

SNAP The Stanford Network Analysis Project (SNAP) [Les] provides in the Stanford Large Network Dataset Collection ten networks in the category

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

Table 6.2.: Graph parameters of our real-world networks. The maximum degree is denoted by Δ and the average degree by $\varnothing$.

	graph	n	m	Δ	$\varnothing$
LWA	cnr-2000	325,557	$\approx$ 2.7M	18,236	16.8
	eu-2005	862,664	$\approx$ 16.1M	68,963	37.4
	in-2004	$\approx$ 1,3M	$\approx$ 13.5M	21,869	19.7
SNAP	ego-Facebook	4,039	88,234	1,045	43.7
	ego-Twitter	81,306	$\approx$ 1.3M	3,383	33.0
	ego-Gplus	107,614	$\approx$ 12.2M	20,127	227.4
	soc-Epinions1	75,879	405,740	3,044	10.7
	soc-Slashdot0811	77,360	469,180	2,539	12.1
	soc-Slashdot0902	82,168	504,230	2,552	12.3
	soc-Pokec	$\approx$ 1.6M	$\approx$ 22.3M	14,854	27.3
	soc-LiveJournal1	$\approx$ 4.8M	$\approx$ 42.8M	20,333	17.7
	wiki-Vote	7,115	100,762	1,065	28.3
DIMACS	coPapersDBLP	540,486	$\approx$ 15.2M	3,299	56.4
	coPapersCiteseer	434,102	$\approx$ 16.0M	1,188	73.9
	coAuthorsDBLP	299,067	977,676	336	6.5
	citationCiteseer	268,495	$\approx$ 1.1M	1,318	8.6
	coAuthorsCiteseer	227,320	814,134	1,372	7.2
DBLP	graph_thres_01	715,633	$\approx$ 2.5M	804	7.0
	graph_thres_02	282,831	640,697	201	4.5
	graphConference	5,599	8,492	53	3.0

social networks. We consider nine of these ten networks as the last graph (calles wiki-RfA) is encoded in a different format containing lots of additional text data. Eight of the nine considered networks are directed. Since our model works with undirected graphs, we use for these eight networks the underlying undirected graph as input.

DIMACS We considered all five social networks from the co-author citation category in the 10th DIMACS challenge [DIM12].

DBLP We consider coauthor networks derived from the DBLP dataset where the vertices represent authors and the edges represent co-authorship in

at least one paper. The DBLP-dataset was generated on February 2012 following the documentation from http://dblp.uni-trier.de/xml/. As it turned out that this DBLP graph is too large for our exact approach, we derived the following subnetworks: First, we made the graph sparser by making two vertices adjacent if the corresponding two authors are co-authors in at least two papers instead of one paper. We denote this graph by graph_thres_2 (graph_thres_1 denotes the original graph). Second, we just considered papers that appeared in some algorithm conference (the exact conference list here is: AAIM, ALENEX, COCOON, ESA, FAW, ISAAC, SEA, SODA, SWAT, WADS, WALCOM, WEA) and removed all isolated vertices. The resulting graph is denoted by graphConference.

In total, we considered 20 graphs resulting in 260 instances (we tested 13 different values of k).

Upper bounds. Our two upper bound heuristics (see Section 6.6.4) perform well compared to the clustering heuristic of Lu et al. [LSB12]. The solutions found by our greedy heuristic were on average 23 % smaller than the solutions obtained by their clustering heuristic. Further improving this result, the solutions of our DP based heuristic were on average 6 % better than the ones of our greedy heuristic. The solutions of the clustering heuristic were never smaller than the solutions of the DP based heuristic and only eight solutions (from overall 260 instances) of the greedy heuristic were smaller than the DB based heuristic; see Figure 6.10 for a more detailed comparison. Notably, comparing the solutions of our DP heuristic to the optimal solution in the 56 instances that we could solve optimally, it turns out that on 51 instances the DP heuristic actually computed an optimal solution. In the remaining five instances, the solution of the DP heuristic was at most 13 edges (0.75 %) above the optimum.

The running time comparison is not so clear as the picture is not as homogeneous as it was for the comparison of the solution sizes. On average, our greedy heuristic is the fasted and the DP based heuristic is the slowest of the three heuristics. Out of the 260 instances, the greedy heuristic could solve 251 instances, the clustering heuristic 245 instances, and the DP based heuristic 239 instances. All instances solved by the clustering heuristic could also be solved by our greedy heuristic. There is, however, one instance (soc-Pokec with $k = 100$) where only the DP based heuristic produces a solution within a five-minute time limit. Although the greedy heuristic is in many instances more than 20 times faster than the clustering heuristic, there are also quite a number

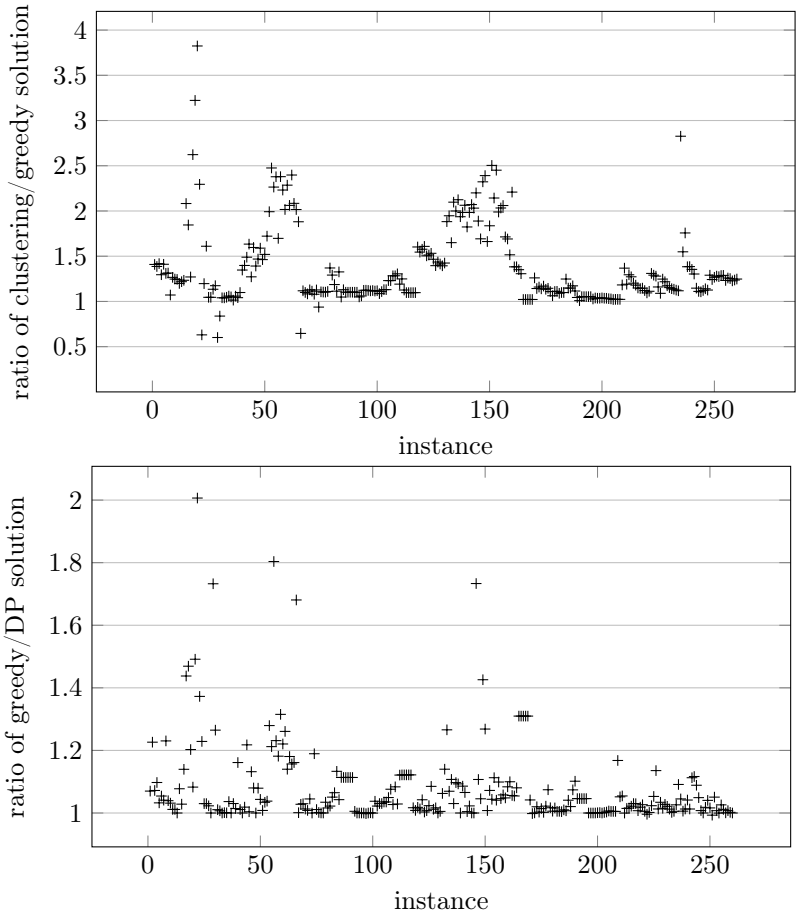


Figure 6.10.: Comparison of the solution sizes of the heuristics. *Top:* Comparison between the clustering heuristic and the greedy heuristic. Values greater than one indicate that greedy heuristic produced a smaller solution for this instance. *Bottom:* Comparison between the DP based heuristic and the greedy heuristic. Values greater than one indicate that the DB based heuristic produced a smaller solution for this instance.

of instances where the clustering heuristic is more than 20 times faster than the greedy heuristic; see Figure 6.11 for a picture of this comparison. The reason for this inhomogeneous behavior is the running time dependence on k. While both our heuristics need in general more time for increasing values of k, the running time of the clustering heuristic has not such a clear dependence on k. For some instances, the clustering heuristic even becomes faster for increasing values of k; see Figure 6.12 for a closer look on two examples.

Results for dynamic programming. Our dynamic program (Phase 1) finishes in 58 out of 260 instances within the time limit of one hour and 56 out of the 58 lower bounds could be realized in Phase 2, that is, 21 % (56 out of 260 instances) could be solved optimally. See Table 6.3 for some representative results on four graphs and Table A.1 (on page 216) for the complete results.

The graphs in the different categories behave very differently. For the three graphs from the Laboratory for Web Algorithmics of the university of Milano (LWA) our implementation rarely proceeds beyond applying the data reduction rule (Reduction Rule 6.1) and never finishes with Phase 1. For larger values of k even the data reduction could not be applied within the time limit. These graphs are simply too big and their maximum degree is simply too large for our exact approach.

Considering the instances from the SNAP-category, the results of the experiments were just a little better. From all nine graphs solely the smallest graph ego-Facebook could be solved optimally for $k = 2$ and $k = 3$. In most cases the dynamic program could improve the lower bound provided by the data reduction rule. After the one hour time limit, the maintained upper bound was, however, still roughly twice as large as the lower bound.

Our algorithm performs better on the DIMACS graphs: 43 % of the corresponding instances could be solved optimally. Interestingly, our exact approach worked best with the coPapersCiteseer graph although this graph was the largest DIMACS-graph (in terms of $n + m$) and fourth largest of all graphs that we consider. We could optimally k-anonymize this graph for all considered values of k. The second largest DIMACS-graph (coPapersDBLP) was, however, very resistant to our dynamic program: It did not finish for any value of k.

Finally, the results for the DBLP graph and subgraphs are as follows. While the whole DBLP coauthor graph (graph_thres_01) could not be solved optimally for any k, its subgraphs graph_thres_02 and graphConference could be solved for all considered values of k—mostly within a few seconds.

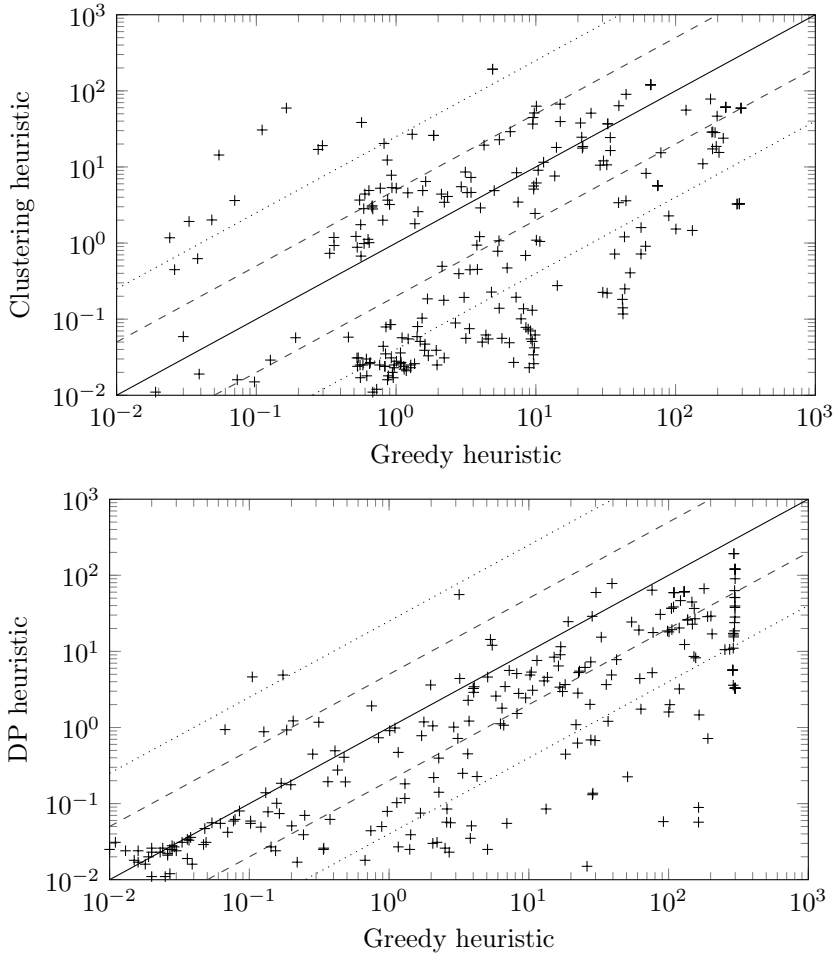


Figure 6.11.: Comparison of the running times of the heuristics. *Top:* Comparison between the clustering heuristic and the greedy heuristic. *Bottom:* Comparison between the DP based heuristic and the greedy heuristic. The solid line denotes the values where both heuristics would have the same running time. Marks above the solid line indicate an instance where the heuristic on the y-axis is faster, marks below the solid line indicate an instance where the heuristic on the x-axis is faster. The dashed lines mark the factor 5 increase/decrease and the dotted lines mark the factor 20 increase/decrease.

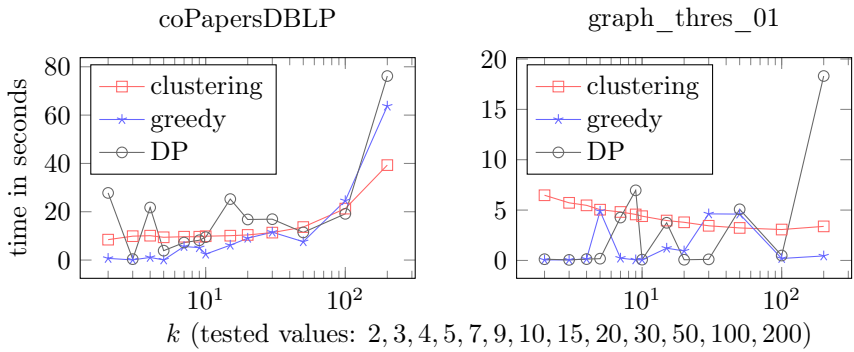


Figure 6.12.: Dependence of the running time on the parameter k for the three heuristics. The left diagram shows the running times for the graph coPapersDBLP, the right diagram for the graph graph_thres_1. Interestingly, the clustering heuristic becomes for increasing value of k slower on the graph coPapersDBLP, but faster on the graph graph_thres_1.

See Table 6.4 for an overview on how many different k-values each graph could be optimally k-anonymized.

Analysis of the results. Our upper bound heuristics perform very well compared to the clustering heuristic of Lu et al. [LSB12]. Furthermore, on instances where we know the optimum solution size, the DP-based heuristic produces optimal or near-optimal solutions. Hence, there is only little room for improvements in terms of the solution quality.

Our exact approach, however, was not overly successful. In general, the approach works better with small k, see Figure 6.13. This comes mainly from the fact that for larger values of k the solution size increases. Since each $(k+1)$-anonymous graph is also k-anonymous, the solution size cannot decrease. Observe that the pruning in the traceback phase of the dynamic program as described in Section 6.6.2 could devitalize this affect a bit. As one can see in Figure 6.14, there is a direct correlation between the lower bound increase speed and the percentage of pruned branches. This is, however, no explanation for the fact that we could solve the graph coPapersCiteseer for $k = 10$ in less than one second (solution size: 958), but could not solve the smaller graph graph_thres_01 for $k = 3$ within one hour although the maintained upper

Table 6.3.: Experimental results on real-world instances with enabled data reduction. For comparison with the optimum and the lower and upper bounds, we display the results of our DP based heuristic. Opt denotes the optimal solution size for the Anonym E-Ins instance. DR denotes the sum of changes in the degree sequence due to the data reduction (Reduction Rule 6.1) divided by two to make the comparison with the solution size (number of inserted edges) easier. If the time entry for Opt 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. Note that the graph graph_thres_1 with $k = 2$ is one of the two instances where Phase 1 is finished, but in Phase 2 the lower bound computed in Phase 1 could not be realized.

graph	k	DP based Heuristic	Opt	DR	DP bounds lower	DP bounds upper	time (in sec)
citation Citeseer	2	457	457	65.5	457.0	457	2.64
	5	1,764		560.0	1,225.0	1,756	
	10	4,775		1,880.5	3,171.0	4,771	
	100	81,464		43,131.0	49,045.5	81,464	
coPapers Citeseer	2	79	79	22.5	79.0	79	263.15
	5	326	326	142.0	326.0	326	0.11
	10	958	958	398.5	958.0	958	0.35
	100	22,006	22,006	10,673.0	22,006.0	22,006	97.36
graph_ thres_01	2	178		69.5	176.0	176	583.21
	5	994		430.5	745.0	991	
	10	2,538		1,137.0	1,853.5	2,538	
	100	38,087		21,173.5	26,298.5	38,087	
ego-Gplus	2	10,017		2,080.5	5,861.0	9,973	
	5	32,031		10,211.0	19,996	31,761	
	10	85,825		30,064.0	51,825.5	84,148	
	100						

bound for the dynamic program is 478. For the graph coPapersCiteseer the pruning was basically useless as the dynamic program itself produced for each k-value except $k = 2$ at most 90 solutions; for other graphs several million solutions are produced for each k-value. Thus, independent from k, the pruning, the solution size, and the graph size, some "hidden" structure of the graph coPapersCiteseer made it accessible for our exact approach.

Table 6.4.: Graph parameters of the real-world networks; see Table 6.2 for more precise values of n and m. The maximum degree is denoted by Δ, the average degree by $\varnothing$, the number of nonempty blocks by $\#{>}0$. We run our exact approach to k-anonymize each graph with 13 different values of k. $\#k$ denotes how many of these attempts were successful.

graph	$\approx n$	$\approx m$	Δ	$\varnothing$	$\#{>}0$	$\#{=}1$	$\#k$
coPapersDBLP	0.5M	15.2M	3,299	56.4	698	46	0
coPapersCiteseer	0.4M	16.0M	1,188	73.9	854	35	13
coAuthorsDBLP	0.2M	0.9M	336	6.5	193	14	4
citationCiteseer	0.2M	1.1M	1,318	8.6	352	39	3
coAuthorsCiteseer	0.2M	0.8M	1,372	7.2	171	13	8
ego-Facebook	4K	88K	1,045	43.7	227	15	2
ego-Twitter	81K	1.3M	3,383	33.0	634	69	0
ego-Gplus	0.1M	12.2M	20,127	227.4	3,424	498	0
soc-Epinions1	75K	0.4M	3,044	10.7	491	54	0
soc-Slashdot0811	77K	0.5M	2,539	12.1	450	41	0
soc-Slashdot0902	82K	0.5M	2,552	12.3	457	50	0
soc-Pokec	1.6M	22.3M	14,854	27.3	803	92	0
soc-LiveJournal1	4.8M	42.8M	20,333	17.7	1,642	170	0
wiki-Vote	7K	0.1M	1,065	28.3	300	40	0
cnr-2000	0.3M	2.7M	18,236	16.8	681	80	0
eu-2005	0.8M	16.1M	68,963	37.4	1,924	248	0
in-2004	1,3M	13.5M	21,869	19.7	1,464	163	0
graph_thres_01	0.7M	2.5M	804	7.0	344	41	0
graph_thres_02	0.2M	0.6M	201	4.5	144	12	13
graphConference	5K	8K	53	3.0	43	3	13

Table 6.4 lists some additional graph parameters for each considered graph: the maximum degree, the average degree, the number of nonempty blocks, and the number of blocks with size one. Unfortunately, none of these parameters yields an explanation for the accessibility of the graph coPapersCiteseer: For almost each parameter we find graphs with smaller parameter value as well as graphs with bigger parameter values such that we could not 2-anonymize these graphs within one hour. There is only one exception: The parameter number of blocks

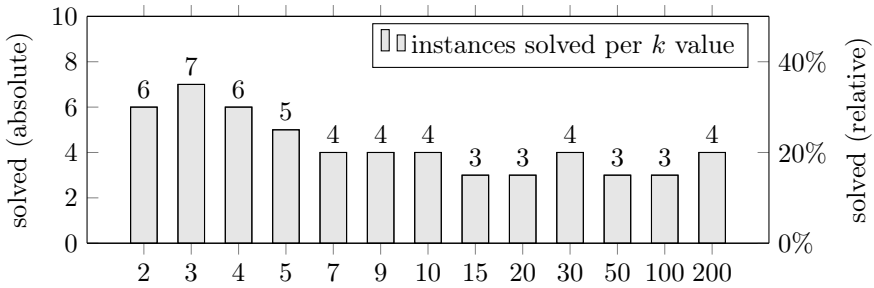


Figure 6.13.: The number of solved instances per k-value out of the 20 graphs. The solved instances are mostly graphs from the two categories DIMACS and DBLP; see Tables 6.2 and 6.4 for the respective graphs and their parameters.

of size one, that is, the number of blocks violating the 2-anonymous property. Although in the graph coPapersCiteseer 535 blocks violate the 100-anonymous property, we could optimally 100-anonymize this graph in less than two minutes. However, no graph contains 535 size-one blocks. Thus, also the parameter number of blocks violating the k-anonymity constraint does not explain the observed results.

Another attempt to explain experimental results is to check whether the efficiently solvable graphs have a special degree sequence. To this end, we depict in Figure 6.15 the degree sequences of six graphs and the number of k-values for which we can solve these graphs optimally. As one can see, the degree sequences of the two best handled graphs, coPapersCiteseer and graph_thres_02 behave very differently. While for increasing degree d the number of vertices with degree d falls rapidly for the graph graph_thres_02, this is not true for the graph coPapersCiteseer: from all degree sequences in Figure 6.15, the one of the coPapersCiteseer graph is in a sense the most "balanced" one.

Summarizing, none of the parameters that we considered can explain all observed results of our experiments. Our theoretical findings in Section 6.3 only provide some evidence why the graph graph_thres_02 is accessible to our exact approach as the parameter maximum degree is comparatively small. Thus, a reasonable next step is to analyze whether particular combinations of parameter values make the instances tractable. Due to the large range of possible combinations, this needs either an automated preprocessing of combinations or further theoretical results that lead to promising parameters and parameter combinations.

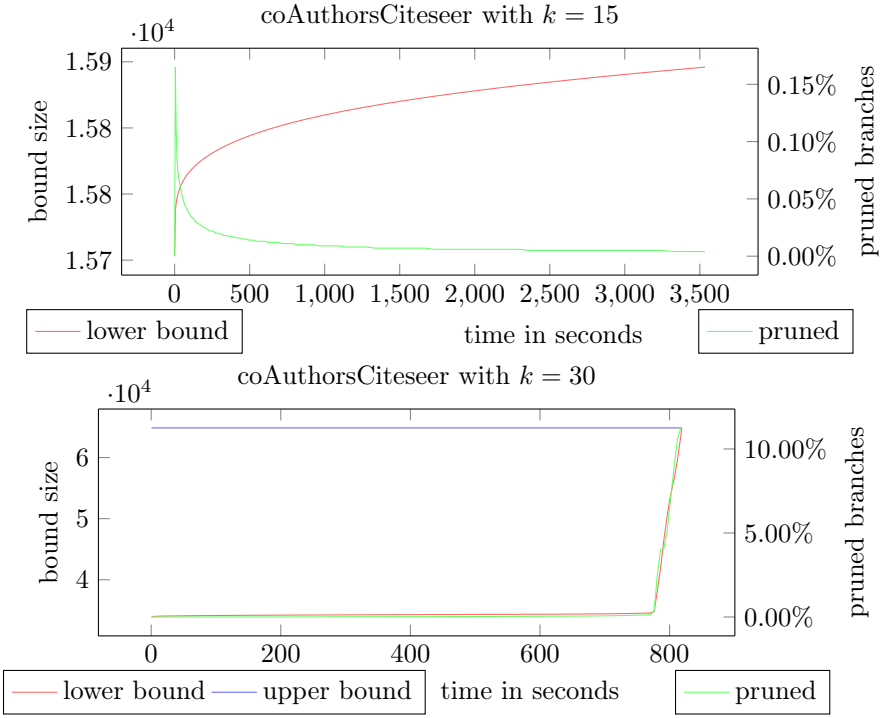


Figure 6.14.: The development of the lower and upper bounds in the dynamic program over time for the graph coAuthorsCiteseer for $k = 15$ and $k = 30$. The exact approach did not finish for $k = 15$ but for $k = 30$ it did. The upper bound, as depicted in the case $k = 30$, is usually quite accurate from the beginning (when the dynamic program finds the first nonrealizable solution). For $k = 15$, we do not show the upper bound to have a better view on the lower bound, which increases rather slowly. Note that the pruning plot is cumulative over time. Thus the absolute values here are somewhat misleading. However, this cumulative plotting demonstrates the correlation to the rate of increase for the lower bound very well. From a certain point, most solutions produced by the dynamic program can be quickly determined as to costly. This can be observed for the case $k = 30$: after about 770 seconds, the lower bound increases very quickly as most branches in the traceback can be pruned.

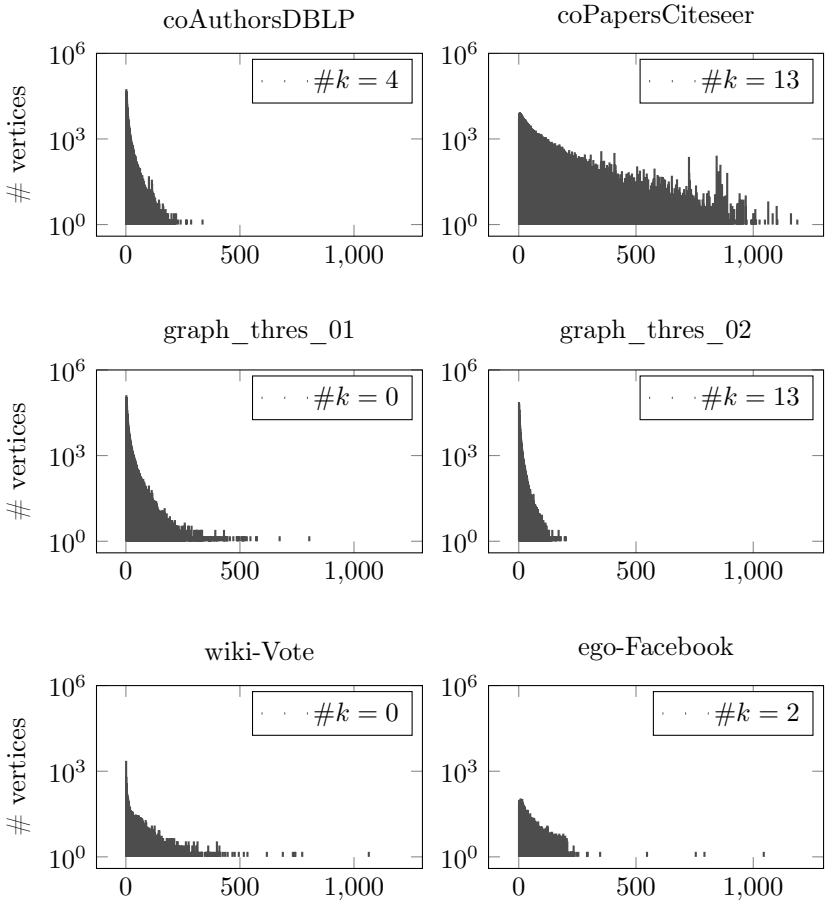


Figure 6.15.: The degree distribution of several real-world graphs; the number $\#k$ denotes for how many of the 13 considered values of k the graph could be k-anonymized. The smallest visible bars depict blocks of size one, that is, vertices with unique degree.

6.7. Conclusion

In this chapter, we contributed to a better theoretical and practical understanding of a basic problem in graph anonymization. On the one side we partially explain the quality of a successful heuristic approach [LT08] and tuned this heuristic approach to optimally solve large scale real-world networks. On the other side we proposed a method for deriving efficient and effective preprocessing algorithms for degree sequence completion problems. Our work just being one of the first steps in the so far underdeveloped field of studying the computational complexity of graph anonymization [Che+13b], there are numerous challenges for future research. First, the upper bounds proven in our results ask for improvement (which we consider as promising). Second, we would like to know the computational complexity of the following variant of the Realizable Degree Sequence Anonymity problem that asks to make a given degree sequence k-anonymous such that the simple Erdős-Gallai test is passed (see Section 6.6.2).

EG-Realizable Degree Sequence Anonymity
Input: A degree sequence $\mathcal{S} = \{d_1, d_2, \ldots, d_n\}$ and integers $k, s \in \mathbb{N}$.
Question: Is there a k-anonymous degree sequence $\mathcal{S}' = \{d'_1, d'_2, \ldots, d'_n\}$ such that

1. $d'_i \geq d_i$ for all $i \in \{1, 2, \ldots, n\}$,
2. $\sum_{i=1}^{n} d'_i - d_i \leq s$,
3. $\mathcal{S}' - \mathcal{S}$ is realizable.

A polynomial-time algorithm for EG-Realizable Degree Sequence Anonymity would most likely lead to an improved implementation as we currently use an algorithm with exponential worst-case running time. Third, it would be interesting to perform a data-driven analysis of parameter values on real-world networks in order to gain parameterizations that can be exploited in a broadband multivariate complexity analysis [FJR13, KN12, Nie10] of Anonym E-Ins. The hope is that this would lead to theoretical results that help explaining the outcome of our experiments. Finally, with Anonym E-Ins we focused on a very basic problem of graph anonymization; there are numerous other models (partially mentioned in the introductory section) that ask for similar studies.

Chapter 7.

Conclusion and Outlook

In this thesis, we studied degree-constrained graph modification problems with a special focus on parameterized complexity and the parameter maximum degree. If we allow vertex or edge deletions, then the parameter maximum degree is not very useful; see Chapter 5 for the intractability results. The parameter maximum degree is, however, useful in the studied completion problems (completion means only edge insertions are allowed). Both DAG Realization and Anonym E-Ins are fixed-parameter tractable with respect to the maximum degree; see Chapters 4 and 6 for the corresponding results. A key observation in both chapters is that if some measurement of the solution exceeds a certain bound that depends only on the maximum degree, then the problems become in some sense "easy": In case of Anonym E-Ins, we can find solutions of size at least $2\Delta^4$ in polynomial time; see Lemma 6.9. In case of DAG Realization, we show that if the topological ordering contains two positions where the so-called potential has value at least Δ^2, then the vertices between these two positions can be arranged almost arbitrarily; see Section 4.3 for the details. The reason for this behavior seems to be the high degree of freedom when inserting edges in small-degree graphs. Hence, for approaching degree-constrained graph completion problems we propose to check whether computing "large" solutions—large compared to the maximum degree or similar parameters—is tractable. Herein, polynomial-time computable degree factors (see Chapter 3) might be a key to handle these solutions.

Next, we briefly discuss the challenges for further research that is motivated by results from this thesis.

Theoretical challenges. The graph modification problems that we studied in this thesis have global degree constraints, see Chapter 1 for an overview. We believe that our methods that led to the fixed-parameter algorithms with

respect to the parameter maximum degree can be used in further graph completion problems. Indeed, we already used the methods which we presented in Sections 6.3 and 6.5 to provide a polynomial problem kernel for the variant of Degree Constraint Editing that only allows edge insertions [FNN14]; see Section 3.2.2 for the formal problem definition. Thus, applying our methods to further graph completion problems that have local as well as global degree constraints is a natural task for future research.

We showed in Chapter 5 that the Degree Anonymity variant where we allow the four modification operations vertex/edge insertion/deletion is fixed-parameter tractable with respect to the combined parameter solution size and maximum degree of the input graph (Theorem 5.23). We believe that the applied method can be extended to general degree sequence modification problems similar to the ones studied in Section 6.5; this remains to be investigated in future work.

Practical challenges. One of the grand challenges of theoretical research on computationally hard problems is to gain a better understanding of when and why heuristic algorithms work [Kar11]. For the DAG Realization problem, Berger and Müller-Hannemann [BM11, BM12] performed experimental studies showing that their heuristics solve the NP-complete problem in (almost) all considered cases when the resulting DAGs are dense or sparse. Our fixed-parameter algorithms for the parameters $\binom{n}{2} - m$ and $m - n + 1$ (see Section 4.4) show that DAG Realization becomes tractable on very dense and very sparse graphs and thus partially explain these results.

In Chapter 6, we gave theoretical evidence for the success of Liu and Terzi's heuristic for Anonym E-Ins [LT08]: we proved that if a minimum solution is large ($\geq 2\Delta^2$), then their two-phase approach solves Anonym E-Ins to optimality. Thus, our theoretical results partially explain the success of their heuristic. Inspired by our theoretical results, we enhanced Liu and Terzi's heuristic and described in Section 6.6 new algorithms for both phases. Furthermore, in Section 6.6.5 we experimentally evaluated our implementation. On the one hand, large networks with more than 16 million edges can be optimally k-anonymized for all tested k. On the other hand, some instances that are significantly smaller could not be solved for any k. This motivates the following question: What makes these instances hard? The measured parameters do not explain this hardness as for example the maximum degree is also small in the hard instances. Thus, our theoretical findings are still insufficient to explain this particular

outcome. So again the following question arises: Is there a hidden structure in large, “easy” instances that could be exploited? Parameterized algorithmics provides the theoretical framework to measure the effect of different structures. However, finding the “right” structure or combination of structures remains a very challenging but also promising endeavor.

Bibliography

[AK85] J. Akiyama and M. Kano. "Factors and factorizations of graphs—a survey". In: *Journal of Graph Theory* 9 (1985), pp. 1–42 (cited on p. 41).

[AL96] S. Arora and C. Lund. "Hardness of approximations". In: *Approximation Algorithms for NP-hard Problems.* PWS Publishing Company, 1996, pp. 399–446 (cited on pp. 25, 26).

[ALY11] C. Aggarwal, Y. Li, and P. S. Yu. "On the hardness of graph anonymization". In: *Proceedings of the 11th IEEE International Conference on Data Mining (ICDM '11).* IEEE, 2011, pp. 1002–1007 (cited on p. 9).

[Aus+99] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.* Springer, 1999 (cited on p. 24).

[BA99] A. Barabási and R. Albert. "Emergence of scaling in random networks". In: *Science* 286 (1999), p. 509 (cited on p. 92).

[Baz+14a] C. Bazgan, M. Chopin, A. Nichterlein, and F. Sikora. "Parameterized approximability of maximizing the spread of influence in networks". In: *Journal of Discrete Algorithms* 27 (2014), pp. 54–65 (cited on p. ix).

[Baz+14b] C. Bazgan, M. Chopin, A. Nichterlein, and F. Sikora. "Parameterized inapproximability of target set selection and generalizations". In: *Computability* 3 (2014), pp. 135–145 (cited on p. ix).

[BBC04] N. Bansal, A. Blum, and S. Chawla. "Correlation clustering". In: *Machine Learning* 56 (2004), pp. 89–113 (cited on p. 1).

[BBD06] P. Burzyn, F. Bonomo, and G. Durán. "NP-completeness results for edge modification problems". In: *Discrete Applied Mathematics* 154 (2006), pp. 1824–1844 (cited on p. 3).

[BCP11] A. Borri, T. Calamoneri, and R. Petreschi. "Recognition of unigraphs through superposition of graphs". In: *Journal of Graph Algorithms and Applications* 15 (2011), pp. 323–343 (cited on p. 162).

[BDVD11] P. Bonizzoni, G. Della Vedova, and R. Dondi. "Anonymizing binary and small tables is hard to approximate". In: *Journal of Combinatorial Optimization* 22 (2011), pp. 97–119 (cited on p. 10).

[Ber11] A. Berger. “Directed Degree Sequences”. PhD thesis. Institut für Informatik, Martin-Luther-Universität Halle-Wittenberg, 2011 (cited on pp. 5, 48, 61).

[Ber14] A. Berger. “A note on the characterization of digraphic sequences”. In: *Discrete Mathematics* 314 (2014), pp. 38–41 (cited on p. 39).

[Bet+12] N. Betzler, R. Bredereck, R. Niedermeier, and J. Uhlmann. “On bounded-degree vertex deletion parameterized by treewidth”. In: *Discrete Applied Mathematics* 160 (2012), pp. 53–60 (cited on p. 44).

[Bev+13] R. van Bevern, R. Bredereck, M. Chopin, S. Hartung, F. Hüffner, A. Nichterlein, and O. Suchý. “Parameterized complexity of dag partitioning”. In: *Proceedings of the 8th International Conference on Algorithms and Complexity (CIAC ’13)*. Vol. 7878. LNCS. Springer, 2013, pp. 49–60 (cited on p. ix).

[Bev+14] R. van Bevern, S. Hartung, A. Nichterlein, and M. Sorge. “Constant-factor approximations for Capacitated Arc Routing without triangle inequality”. In: *Operations Research Letters* 42 (2014), pp. 290–292 (cited on p. ix).

[BH65] L. Beineke and F. Harary. “Local restrictions for various classes of directed graphs”. In: *Journal of the London Mathematical Society* 1 (1965), pp. 87–95 (cited on pp. 39, 47).

[BHL14] H. L. Bodlaender, P. Heggernes, and D. Lokshtanov. “Graph modification problems (Dagstuhl seminar 14071)”. In: *Dagstuhl Reports* 4 (2014), pp. 38–59 (cited on p. 1).

[BLS99] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: a Survey*. Vol. 3. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 1999 (cited on pp. 104, 106, 114, 117, 162).

[BM11] A. Berger and M. Müller-Hannemann. “Dag realizations of directed degree sequences”. In: *Proceedings of the 18th International Symposium on Fundamentals of Computation Theory (FCT ’11)*. Vol. 6914. LNCS. Springer, 2011, pp. 264–275 (cited on pp. ix, 4, 5, 47, 48, 54–56, 89, 200).

[BM12] A. Berger and M. Müller-Hannemann. “How to attack the NP-complete DAG realization problem in practice”. In: *Proceedings of the 11th International Symposium on Experimental Algorithms (SEA ’12)*. Vol. 7276. LNCS. Springer, 2012, pp. 51–62 (cited on pp. 5, 48, 49, 56, 61, 84, 200).

[BN14] C. Bazgan and A. Nichterlein. “Parameterized inapproximability of degree anonymization”. In: *Proceedings of the 9th International Symposium on Parameterized and Exact Computation (IPEC ’14)*. Vol. 8894. LNCS. Springer, 2014, pp. 75–84 (cited on p. x).

[BNN13] R. Bredereck, A. Nichterlein, and R. Niedermeier. “Pattern-guided k-anonymity”. In: *Algorithms* 6 (2013), pp. 678–701 (cited on pp. ix, x).

[Bod09] H. L. Bodlaender. "Kernelization: new upper and lower bound techniques". In: *Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC '09)*. Vol. 5917. LNCS. Springer, 2009, pp. 17–37 (cited on p. 22).

[Bol+] P. Boldi, B. Codenotti, M. Rosa, M. Santini, and S. Vigna. *University of Milano, Laboratory of Web Algorithms. Datasets,* URL: http://law.di.unimi.it/datasets.php (cited on p. 186).

[Bon+13] P. Bonizzoni, G. Della Vedova, R. Dondi, and Y. Pirola. "Parameterized complexity of k-anonymity: hardness and tractability". In: *Journal of Combinatorial Optimization* 26 (2013), pp. 19–43 (cited on p. 10).

[Bre+13a] R. Bredereck, S. Hartung, A. Nichterlein, and G. J. Woeginger. "The complexity of finding a large subgraph under anonymity constraints". In: *Proceedings of the 24th International Symposium on Algorithms and Computation (ISAAC '13)*. Vol. 8283. LNCS. Springer, 2013, pp. 152–162 (cited on p. x).

[Bre+13b] R. Bredereck, T. Köhler, A. Nichterlein, R. Niedermeier, and G. Philip. "Using patterns to form homogeneous teams". In: *Algorithmica* (2013). Online Available. (cited on pp. ix, x).

[Bre+14a] R. Bredereck, J. Chen, A. Nichterlein, P. Faliszewski, and R. Niedermeier. "Prices matter for the parameterized complexity of shift bribery". In: *Proceedings of the 28th Conference on Artificial Intelligence (AAAI '14)*. 2014, pp. 552–558 (cited on p. ix).

[Bre+14b] R. Bredereck, V. Froese, S. Hartung, A. Nichterlein, R. Niedermeier, and N. Talmon. "The complexity of degree anonymization by vertex addition". In: *Proceedings of the International Conference on Algorithmic Aspects of Information and Management (AAIM '14)*. Vol. 8546. LNCS. Springer, 2014, pp. 44–55 (cited on pp. ix, 1, 9, 10, 125).

[Bre+14c] R. Bredereck, A. Nichterlein, R. Niedermeier, and G. Philip. "The effect of homogeneity on the computational complexity of combinatorial data anonymization". In: *Data Mining and Knowledge Discovery* 28 (2014), pp. 65–91 (cited on pp. ix, x, 10).

[Cai08] L. Cai. "Parameterized complexity of cardinality constrained optimization problems". In: *The Computer Journal* 51 (2008), pp. 102–121 (cited on p. 25).

[Cai96] L. Cai. "Fixed-parameter tractability of graph modification problems for hereditary properties". In: *Information Processing Letters* 58 (1996), pp. 171–176 (cited on pp. 3, 162).

[CCC06] L. Cai, S. M. Chan, and S. O. Chan. "Random separation: a new method for solving fixed-cardinality optimization problems". In: *Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC '06)*. Vol. 4169. LNCS. Springer, 2006, pp. 239–250 (cited on p. 127).

[CGG06] Y. Chen, M. Grohe, and M. Grüber. "On parameterized approximability". In: *Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC '06)*. Vol. 4169. LNCS. 2006, pp. 109–120 (cited on p. 25).

[Cha+89] G. Chartrand, L. Lesniak, C. M. Mynhardt, and O. R. Oellermann. "Degree uniform graphs". In: *Annals of the New York Academy of Sciences* 555 (1989), pp. 122–132 (cited on p. 162).

[Che+12] S. Chester, J. Gaertner, U. 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 '12)*. IEEE Computer Society, 2012, pp. 418–422 (cited on pp. 9, 10).

[Che+13a] S. Chester, B. M. Kapron, G. Ramesh, G. Srivastava, A. Thomo, and S. Venkatesh. "Why Waldo befriended the dummy? k-anonymization of social networks with pseudo-nodes". In: *Social Network Analysis and Mining* 3 (2013), pp. 381–399 (cited on pp. 1, 9, 10, 125).

[Che+13b] S. Chester, B. Kapron, G. Srivastava, and S. Venkatesh. "Complexity of social network anonymization". In: *Social Network Analysis and Mining* 3 (2013), pp. 151–166 (cited on pp. 9, 11, 144, 198).

[Che66] W.-K. Chen. "On the realization of a (p, s)-digraph with prescribed degrees". In: *Journal of The Franklin Institute* 281 (1966), pp. 406–422 (cited on pp. 5, 47).

[Cho+14] M. Chopin, A. Nichterlein, R. Niedermeier, and M. Weller. "Constant thresholds can make target set selection tractable". In: *Theory of Computing Systems* 55 (2014), pp. 61–83 (cited on p. ix).

[CHT13] J. Casas-Roma, J. Herrera-Joancomartí, and V. Torra. "An algorithm for k-degree anonymity on large networks". In: *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM '13)*. ACM Press, 2013, pp. 671–675 (cited on p. 9).

[CLT10] K. L. Clarkson, K. Liu, and E. Terzi. "Towards identity anonymization in social networks". In: *Link Mining: Models, Algorithms, and Applications*. Springer, 2010, pp. 359–385 (cited on pp. 6, 131, 168, 184, 185).

[Cor88] G. Cornuéjols. "General factors of graphs". In: *Journal of Combinatorial Theory. Series B* 45 (1988), pp. 185–198 (cited on p. 43).

[DF13] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer, 2013 (cited on pp. 2, 21, 107, 110, 122).

[DFM06] R. G. Downey, M. R. Fellows, and C. McCartin. "Parameterized approximation problems". In: *Proceedings of the 2nd International Workshop on Parameterized and Exact Computation (IWPEC '06).* Vol. 4169. LNCS. Springer, 2006, pp. 121–129 (cited on p. 25).

[Die10] R. Diestel. *Graph Theory.* 4th. Vol. 173. Graduate Texts in Mathematics. Springer, 2010 (cited on pp. 13, 152).

[DIM12] DIMACS. *Graph partitioning and graph clustering. 10th DIMACS challenge.* 2012. URL: http://www.cc.gatech.edu/dimacs10/ (cited on p. 187).

[DLS14] M. Dom, D. Lokshtanov, and S. Saurabh. "Kernelization lower bounds through colors and ids". In: *ACM Transactions on Algorithms* 11 (2014), p. 13 (cited on p. 107).

[Dwo11] C. Dwork. "A firm foundation for private data analysis". In: *Communications of the ACM* 54 (2011), pp. 86–95 (cited on p. 8).

[EC88] E. S. El-Mallah and C. J. Colbourn. "The complexity of some edge deletion problems". In: *IEEE Transactions on Circuits and Systems* 35 (1988), pp. 354–362 (cited on p. 3).

[Edm64] J. Edmonds. "Existence of k-edge connected ordinary graphs with prescribed degrees". In: *Journal of National Bureau of Standards* 68 (1964), pp. 73–74 (cited on pp. 35, 47).

[Edm65] J. Edmonds. "Paths, trees, and flowers". In: *Canadian Journal of Mathematics* 17 (1965), pp. 449–467 (cited on p. 40).

[EG60] P. Erdős and T. Gallai. "Graphs with prescribed degrees of vertices (in Hungarian)". In: *Matematikai Lapok* 11 (1960), pp. 264–274 (cited on pp. 5, 30, 33, 47, 165, 166, 170, 171).

[EMT10] P. L. Erdős, I. Miklós, and Z. Toroczkai. "A simple Havel-Hakimi type algorithm to realize graphical degree sequences of directed graphs". In: *Electronic Journal of Combinatorics* 17 (2010), R66 (cited on pp. 35, 38).

[ES12] D. Eppstein and E. S. Spiro. "The h-index of a graph and its application to dynamic subgraph statistics". In: *Journal of Graph Algorithms and Applications* 16 (2012), pp. 543–567 (cited on p. 162).

[EWC09] P. A. Evans, T. Wareham, and R. Chaytor. "Fixed-parameter tractability of anonymizing data by suppressing entries". In: *Journal of Combinatorial Optimization* 18 (2009), pp. 362–375 (cited on p. 10).

[Fel+09] M. Fellows, D. Hermelin, F. Rosamond, and S. Vialette. "On the parameterized complexity of multiple-interval graph problems". In: *Theoretical Computer Science* 410 (2009), pp. 53–61 (cited on p. 137).

[Fel+11] M. R. Fellows, J. Guo, H. Moser, and R. Niedermeier. "A generalization of Nemhauser and Trotter's local optimization theorem". In: *Journal of Computer and System Sciences* 77 (2011), pp. 1141–1158 (cited on pp. 3, 44, 95).

[FG01] M. Frick and M. Grohe. "Deciding first-order properties of locally tree-decomposable structures". In: *Journal of the ACM* 48 (2001), pp. 1184–1206 (cited on pp. 44, 127).

[FG06] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006 (cited on pp. 2, 21).

[FJR13] M. R. Fellows, B. M. P. Jansen, and F. A. Rosamond. "Towards fully multivariate algorithmics: parameter ecology and the deconstruction of computational complexity". In: *European Journal of Combinatorics* 34 (2013), pp. 541–566 (cited on pp. 21, 57, 198).

[FKW04] F. V. Fomin, D. Kratsch, and G. J. Woeginger. "Exact (exponential) algorithms for the dominating set problem". In: *Proceedings of the 30th International Workshop on Graph-Theoretic Conecpts in Computer Science (WG '04)*. Vol. 3353. LNCS. Springer, 2004, pp. 245–256 (cited on p. 107).

[FNN14] V. Froese, A. Nichterlein, and R. Niedermeier. "Win-win kernelization for degree sequence completion problems". In: *Proceedings of the 14th Scandinavian Workshop on Algorithm Theory (SWAT '14)*. Vol. 8503. LNCS. Springer, 2014, pp. 194–205 (cited on pp. xi, 44, 200).

[FT87] A. Frank and É. Tardos. "An application of simultaneous diophantine approximation in combinatorial optimization". In: *Combinatorica* 7 (1987), pp. 49–65 (cited on p. 82).

[Ful60] D. Fulkerson. "Zero-one matrices with zero trace". In: *Pacific Journal of Mathematics* 10 (1960), pp. 831–836 (cited on pp. 5, 38, 47).

[Fun+10] B. C. M. Fung, K. Wang, R. Chen, and P. S. Yu. "Privacy-preserving data publishing: a survey of recent developments". In: *ACM Computing Surveys* 42 (2010), 14:1–14:53 (cited on pp. 5, 8, 131).

[Gab83] H. N. Gabow. "An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems". In: *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC '83)*. ACM, 1983, pp. 448–456 (cited on pp. 40, 42, 139, 147, 148, 151, 155).

[Gal57] D. Gale. "A theorem on flows in networks". In: *Pacific Journal of Mathematics* 7 (1957), pp. 1073–1082 (cited on pp. 5, 38, 47).

[GHP13] P. A. Golovach, P. van 't Hof, and D. Paulusma. "Obtaining planarity by contracting few edges". In: *Theoretical Computer Science* 476 (2013), pp. 38–46 (cited on p. 1).

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979 (cited on pp. 2, 20, 23, 50, 99, 100, 109, 116, 120, 122, 135, 136, 178, 179).

[GN07] J. Guo and R. Niedermeier. "Invitation to data reduction and problem kernelization". In: *ACM SIGACT News* 38 (2007), pp. 31–45 (cited on p. 22).

[Gol+13] P. A. Golovach, M. Kaminski, D. Paulusma, and D. M. Thilikos. "Increasing the minimum degree of a graph by contractions". In: *Theoretical Computer Science* 481 (2013), pp. 74–84 (cited on p. 1).

[Gol14a] P. A. Golovach. "Editing to a connected graph of given degrees". In: *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS '14)*. Vol. 8635. LNCS. Springer, 2014, pp. 324–335 (cited on p. 35).

[Gol14b] P. A. Golovach. "Editing to a graph of given degrees". In: *Proceedings of the 9th International Symposium on Parameterized and Exact Computation (IPEC '14)*. Vol. 8894. LNCS. Springer, 2014, pp. 196–207 (cited on pp. 44, 128).

[Gol+95] P. W. Goldberg, M. C. Golumbic, H. Kaplan, and R. Shamir. "Four strikes against physical mapping of DNA". In: *Journal of Computational Biology* 2 (1995), pp. 139–152 (cited on p. 3).

[Gra+04] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. "Automated generation of search tree algorithms for hard graph modification problems". In: *Algorithmica* 39 (2004), pp. 321–347 (cited on p. 3).

[GS13] H. N. Gabow and P. Sankowski. "Algebraic algorithms for b-matching, shortest undirected paths, and f-factors". In: *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS '13)*. IEEE Computer Society, 2013, pp. 137–146 (cited on p. 40).

[Hak62] S. Hakimi. "On realizability of a set of integers as degrees of the vertices of a linear graph. I". In: *Journal of SIAM* 10 (1962), pp. 496–506 (cited on pp. 5, 30, 32, 47).

[Har+13] S. Hartung, A. Nichterlein, R. Niedermeier, and O. Suchý. "A refined complexity analysis of degree anonymization on graphs". In: *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP '13)*. Vol. 7966. LNCS. Springer, 2013, pp. 594–606. Journal version to appear in *Information and Computation* (cited on pp. x, xi).

[Hav55] V. Havel. “A remark on the existence of finite graphs. (in Hungarian)”. In: *Časopis pro pěstování matematiky* 80 (1955), pp. 477–480 (cited on pp. 5, 30, 32, 47).

[HHN14] S. Hartung, C. Hoffmann, and A. Nichterlein. “Improved upper and lower bound heuristics for degree anonymization in social networks”. In: *Proceedings of the 13th International Symposium on Experimental Algorithms (SEA ’14)*. Vol. 8504. LNCS. Springer, 2014, pp. 376–387 (cited on p. xii).

[HIS75] P. L. Hammer, T. Ibaraki, and B. Simeone. “Degree sequences of threshold graphs”. In: *Congressus Numerantium* 21 (1975), pp. 329–355 (cited on p. 162).

[HKN12] S. Hartung, C. Komusiewicz, and A. Nichterlein. “Parameterized algorithmics and computational experiments for finding 2-clubs”. In: *Proceedings of the 7th International Symposium on Parameterized and Exact Computation (IPEC’12)*. Vol. 7535. Springer, 2012, pp. 231–241 (cited on p. ix).

[HKN13] S. Hartung, C. Komusiewicz, and A. Nichterlein. “On structural parameterizations for the 2-club problem”. In: *Proceedings of the 39th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM ’13)*. Vol. 7741. LNCS. Springer, 2013, pp. 233–243 (cited on p. ix).

[HN12] S. Hartung and A. Nichterlein. “NP-hardness and fixed-parameter tractability of realizing degree sequences with directed acyclic graphs”. In: *Proceedings of the 8th International Conference on Computability in Europe (CiE 2012)*. Vol. 7318. LNCS. Springer, 2012, pp. 283–292 (cited on p. ix).

[HN13] S. Hartung and A. Nichterlein. “On the parameterized and approximation hardness of metric dimension”. In: *Proceedings of the 28th IEEE Conference on Computational Complexity (CCC ’13)*. IEEE, 2013, pp. 266–276 (cited on p. ix).

[HS78] S. Hakimi and E. Schmeichel. “Graphs and their degree sequences: a survey”. In: *Theory and Applications of Graphs* (1978), pp. 225–235 (cited on p. 34).

[HS81] P. L. Hammer and B. Simeone. “The splittance of a graph”. In: *Combinatorica* 1 (1981), pp. 275–284 (cited on pp. 3, 162).

[Kan87] R. Kannan. “Minkowski’s convex body theorem and integer programming”. In: *Mathematics of Operations Research* 12 (1987), pp. 415–440 (cited on p. 82).

[Kar11] R. M. Karp. “Heuristic algorithms in computational molecular biology”. In: *Journal of Computer and System Sciences* 77 (2011), pp. 122–128 (cited on p. 200).

[Kar72] R. M. Karp. "Reducibility among combinatorial problems". In: *Complexity of Computer Computations*. Ed. by R. E. Miller and J. W. Thatcher. Plenum Press, 1972, pp. 85–103 (cited on p. 4).

[KL75] D. J. Kleitman and S. Y. Li. "A note on unigraphic sequences". In: *Studies in Applied Mathematics* 4 (1975), pp. 283–287 (cited on p. 162).

[KN12] C. Komusiewicz and R. Niedermeier. "New races in parameterized algorithmics". In: *Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science (MFCS '12)*. Vol. 7464. LNCS. Springer, 2012, pp. 19–30 (cited on pp. 2, 21, 90, 161, 198).

[KNU11] C. Komusiewicz, R. Niedermeier, and J. Uhlmann. "Deconstructing intractability—a multivariate complexity analysis of interval constrained coloring". In: *Journal of Discrete Algorithms* 9 (2011), pp. 137–151 (cited on pp. 57, 89).

[Kra14] S. Kratsch. "Recent developments in kernelization: a survey". In: *Bulletin of EATCS* 113 (2014), pp. 58–97 (cited on p. 22).

[KST99] H. Kaplan, R. Shamir, and R. E. Tarjan. "Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs". In: *SIAM Journal on Computing* 28 (1999), pp. 1906–1922 (cited on p. 3).

[KT00] P. Katerinis and N. Tsikopoulos. "Minimum degree and f-factors in graphs". In: *New Zealand Journal of Mathematics* 29 (2000), pp. 33–40 (cited on pp. 41, 140).

[KW73] D. Kleitman and D. Wang. "Algorithms for constructing graphs and digraphs with given valences and factors". In: *SIAM Journal on Discrete Mathematics* 6 (1973), pp. 79–88 (cited on pp. 5, 35–38, 47, 62).

[Len83] H. W. Lenstra. "Integer programming with a fixed number of variables". In: *Mathematics of Operations Research* 8 (1983), pp. 538–548 (cited on p. 82).

[Les] J. Leskovec. *Stanford Network Analysis Package (SNAP)*. URL: http://snap.stanford.edu/index.html (cited on p. 186).

[LH08] J. Leskovec and E. Horvitz. "Planetary-scale views on a large instant-messaging network". In: *Proceedings of the 17th International Conference on World Wide Web (WWW '08)*. ACM Press, 2008, pp. 915–924 (cited on p. 2).

[Lov72] L. Lovász. "The factorization of graphs. II". In: *Acta Mathematica Academiae Scientiarum Hungarica* 23 (1972), pp. 223–246 (cited on pp. 42, 43).

[LP86] L. Lovász and M. D. Plummer. *Matching Theory*. Vol. 29. Annals of Discrete Mathematics. North-Holland, 1986 (cited on pp. 31, 35, 40, 42, 47, 139).

[LSB12] X. Lu, Y. Song, and S. Bressan. "Fast identity anonymization on graphs". In: *Proceedings of the 23rd International Conference on Database and Expert Systems Applications (DEXA '12), Part I*. Vol. 7446. LNCS. Springer, 2012, pp. 281–295 (cited on pp. 9, 134, 170, 186, 188, 192, 215).

[LT08] K. Liu and E. Terzi. "Towards identity anonymization on graphs". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data*. SIGMOD '08. ACM, 2008, pp. 93–106 (cited on pp. x, xi, 6, 8, 9, 11, 12, 91, 92, 95, 131, 133, 138, 140, 141, 144, 159, 162–164, 167, 179, 181, 198, 200).

[LY80] J. M. Lewis and M. Yannakakis. "The node-deletion problem for hereditary properties is NP-complete". In: *Journal of Computer and System Sciences* 20 (1980), pp. 219–230 (cited on p. 3).

[Man08] F. Mancini. "Graph modification problems related to graph classes". PhD thesis. University of Bergen, 2008 (cited on p. 3).

[Mar08] D. Marx. "Parameterized complexity and approximation algorithms". In: *The Computer Jorunal* 51 (2008), pp. 60–78 (cited on pp. 24, 25).

[MP94] F. Maffray and M. Preissmann. "Linear recognition of pseudo-split graphs". In: *Discrete Applied Mathematics* 52 (1994), pp. 307–312 (cited on p. 162).

[MS12] L. Mathieson and S. Szeider. "Editing graphs to satisfy degree constraints: a parameterized approach". In: *Journal of Computer and System Sciences* 78 (2012), pp. 179–191 (cited on pp. xi, 3, 43, 44).

[MT09] H. Moser and D. M. Thilikos. "Parameterized complexity of finding regular induced subgraphs". In: *Journal of Discrete Algorithms* 7 (2009), pp. 181–190 (cited on pp. 44, 95).

[MV80] S. Micali and V. V. Vazirani. "An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs". In: *Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science (FOCS '80)*. IEEE, 1980, pp. 17–27 (cited on p. 40).

[NDN11] A. Nichterlein, M. Dom, and R. Niedermeier. "Aspects of a multivariate complexity analysis for rectangle tiling". In: *Operations Research Letters* 39 (2011), pp. 346–351 (cited on p. ix).

[New10] M. E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2010 (cited on p. 21).

[Nic+13] A. Nichterlein, R. Niedermeier, J. Uhlmann, and M. Weller. "On tractable cases of target set selection". In: *Social Network Analysis and Mining* 3 (2013), pp. 233–256 (cited on p. ix).

[Nie06] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006 (cited on pp. 2, 21).

[Nie10] R. Niedermeier. “Reflections on multivariate algorithmics and problem parameterization”. In: *Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS ’07)*. Vol. 5. LIPIcs. IBFI Dagstuhl, Germany, 2010, pp. 17–32 (cited on pp. 21, 57, 198).

[NS09] A. Narayanan and V. Shmatikov. “De-anonymizing social networks”. In: *Proceedings of the 30th IEEE Symposium on Security and Privacy*. IEEE, 2009, pp. 173–187 (cited on p. 9).

[NSS01] A. Natanzon, R. Shamir, and R. Sharan. “Complexity classification of some edge modification problems”. In: *Discrete Applied Mathematics* 113 (2001), pp. 109–128 (cited on pp. 1, 3, 162).

[Per11] A. Perez. “Algorithmes de noyau pour des problèmes d’édition de graphes et autres structures”. PhD thesis. Laboratoire d’Informatique de Robotique et de Microélectronique de Montpellier, 2011 (cited on p. 3).

[Plu07] M. D. Plummer. “Graph factors and factorization: 1985-2003: a survey”. In: *Discrete Mathematics* 307 (2007), pp. 791–821 (cited on p. 41).

[PW96] C. Phillips and T. J. Warnow. “The asymmetric median tree—a new model for building consensus trees”. In: *Discrete Applied Mathematics* 71 (1996), pp. 311–335 (cited on p. 136).

[Rao81] S. Rao. “A survey of the theory of potentially p-graphic and forcibly p-graphic degree sequences”. In: *Combinatorics and Graph Theory*. Springer, 1981, pp. 417–440 (cited on pp. 34, 39).

[RS97] R. Raz and S. Safra. “A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP”. In: *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC ’97)*. ACM Press, 1997, pp. 475–484 (cited on p. 108).

[Rys57] H. Ryser. “Combinatorial properties of matrices of zeros and ones”. In: *Canadian Journal of Mathematics* 9 (1957), pp. 371–377 (cited on pp. 5, 38, 47).

[Sal+11] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao. “Sharing graphs using differentially private graph models”. In: *Proceedings of the 11th ACM SIGCOMM Conference on Internet Measurement*. ACM, 2011, pp. 81–98 (cited on p. 9).

[Sam01] P. Samarati. “Protecting respondents identities in microdata release”. In: *IEEE Transactions on Knowledge and Data Engineering* 13 (2001), pp. 1010–1027 (cited on pp. 8, 131).

[Sha02] R. Sharan. “Graph Modification Problems and their Applications to Genomic Research”. PhD thesis. Sackler Faculty of Exact Sciences, School of Computer Science, Tel-Aviv University, 2002 (cited on pp. 1, 3).

[SS98] P. Samarati and L. Sweeney. "Generalizing data to provide anonymity when disclosing information". In: *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '98)*. ACM Press, 1998, pp. 188–188 (cited on pp. 8, 131).

[Swe02] L. Sweeney. "k-anonymity: a model for protecting privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10 (2002), pp. 557–570 (cited on pp. 8, 131).

[Tut54] W. T. Tutte. "A short proof of the factor theorem for finite graphs". In: *Canadian Journal of Mathematics* 6 (1954), pp. 347–352 (cited on p. 40).

[TV03] A. Tripathi and S. Vijay. "A note on a theorem of Erdös & Gallai". In: *Discrete Mathematics* 265 (2003), pp. 417–420 (cited on pp. 34, 171).

[Vas12] V. Vassilevska Williams. "Multiplying matrices faster than Coppersmith-Winograd". In: *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*. ACM, 2012, pp. 887–898 (cited on p. 41).

[Vaz01] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001 (cited on pp. 25, 108).

[WS11] D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011 (cited on p. 25).

[Wu+10] X. Wu, X. Ying, K. Liu, and L. Chen. "A survey of privacy-preservation of graphs and social networks". In: *Managing and Mining Graph Data*. Springer, 2010, pp. 421–453 (cited on pp. 6, 91).

[Yan81a] M. Yannakakis. "Computing the minimum fill-in is NP-complete". In: *SIAM Journal on Algebraic and Discrete Methods* 2 (1981), pp. 77–79 (cited on p. 3).

[Yan81b] M. Yannakakis. "Edge-deletion problems". In: *SIAM Journal on Computing* 10 (1981), pp. 297–309 (cited on p. 3).

[ZP11] B. Zhou and J. Pei. "The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks". In: *Knowledge and Information Systems* 28 (2011), pp. 47–77 (cited on p. 9).

Appendix A.

Complete List of Experimental Results

Table A.1 (see next page) contains the full list of experimental results on real-world instances with data reduction. Therein, we use the following abbreviations: "Cluster" for the clustering-heuristic of Lu et al. [LSB12], "Greedy" for our greedy heuristic, "DP" for our dynamic programming-based heuristic, "Opt" for optimum solution size computed by our exact approach, "Preprocessed" for the degree changes made by the data reduction, "lower" ("upper") for the lower (upper) bound maintained in the dynamic program. The preprocessed changes due to the data reduction rule as well as the lower and upper bound maintained in the dynamic program are just sums over degrees (see Section 6.6.2 for the details). To make comparison with the solution size (counted in the number of added edges) easier, we divided the values by two (as every edge affects two vertices). Thus, the lower bound and the preprocessed changes may not be integer values any more. If an entry corresponding to a heuristic is empty, then this heuristic could not solve the instance within five minutes. If the time entry for the optimum is empty, then our exact approach could not solve the instance within one hour. In this case the maintained lower and upper bounds of the dynamic program are returned. If the entry for the optimum value is empty but the corresponding time entry is not empty, then our implementation could not realize the provided lower bound.

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
cnr-2000	2	5,627	4,560	4,232		11.5	2,397.5	4,232	4.135	0.05	0.886	
	3	10,590	5,085	4,945		13.0	3,044.5	4,936	3.157	0.056	2.751	
	4	17,604	9,539	8,370		16.5	16.5		3.349	0.075	1.677	
	5	15,766	12,407	8,629		8.0	8.0		2.659	0.089	163.933	
	7	51,961	19,810	13,484		13.0	13.0		7.468	3.452	6.764	
	9	64,908	20,143	16,747		31.5	31.5		8.161	0.137	28.738	
	10	70,558	18,452	17,041		42.0	42.0		9.426	0.131	28.523	
	15	114,434	49,863	33,433		1,102.5	1,102.5		14.244	0.276	0.428	
	20	120,026	190,353	94,867		2,164.0	2,164.0		10.68	1.05	2.054	
	30	244,656	204,361	148,883		5,165.0	5,165.0		43.089	1.206	36.92	
	50	523,925	325,372	264,810		42,251.0	42,251.0		100.517	1.529	8.236	
	100	885,531	846,884	822,213					275.599	3.257	4.007	
	200											
eu-2005	2	47,160	34,420	33,274		8,039.5	8,039.5		39.087	3.39	16.669	
	3	79,425	61,501	60,524		16,532.0	16,532.0		44.157	3.593	290.295	
	4	115,539	97,349	95,243		27,312.5	27,312.5		61.505	8.232	156.363	
	5	157,656	141,566	134,724		38,408.0	38,408.0		78.501	15.38	33.001	
	7	280,635	211,536	198,648		61,090.0	61,090.0		131.885	1.464	165.439	
	9	330,796	315,439	278,367		86,098.0	86,098.0		156.714	11.013	290.441	
	10	388,252	343,640	329,576		98,609.0	98,609.0		191.751	28.275	298.715	
	15	628,701	570,546	512,368		161,934.5	161,934.5		286.537	3.286	299.268	
	20		748,756			261,173.5	261,173.5			32.986		
	30		1,482,233	1,311,212						84.065	293.962	
	50											
	100		5,259,502							22.408		
	200											

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
in-2004	2	8,073	4,711	4,504		660.5	3,075.0	4,504	32.368	0.22	2.087	
	3	13,843	8,188	7,740		1,703.5	1,703.5		30.148	0.225	50.897	
	4	34,807	22,964	21,907		3,661.0	3,661.0		30.483	12.035	5.469	
	5	58,842	26,640	24,583		5,773.0	5,773.0		32.196	10.655	273.715	
	7	62,338	45,089	40,993		13,141.5	13,141.5		28.851	10.548	253.044	
	9	85,512	61,813	58,533		24,369.0	24,369.0		34.154	16.429	291.839	
	10	102,126	75,578	71,677		30,367.5	30,367.5		36.654	0.717	191.174	
	15	195,462	149,588	138,453		60,508.0	60,508.0		56.626	1.598	100.401	
	20	277,159	271,273	207,122		90,832.5	90,832.5		74.486	5.668	287.807	
	30	451,600	485,956			151,688.5	151,688.5		114.279	72.405		
	50	752,596	765,469			274,067.5	274,067.5		184.649	11.955		
	100		1,636,519			589,441.0	589,441.0			31.142		
	200											
ego-Facebook	2	583	555	552	552	130.0	552.0	552	0.015	0.003	0.015	0.375
	3	1,132	1,067	1,067	1,066	279.0	1,066.0	1,066	0.019	0.011	0.02	0.696
	4	1,404	1,250	1,250		524.0	691.5	1,250	0.03	0.059	0.077	
	5	2,109	1,876	1,876		879.0	1,026.0	1,876	0.039	0.019	0.036	
	7	3,707	3,304	3,304			1,811.5	3,312	0.073	0.016	0.018	
	9	5,423	4,854	4,864			2,650.0	4,879	0.097	0.015	26.093	
	10	6,329	5,648	5,648			3,083.5	5,673	0.126	0.029	0.047	
	15	10,804	9,688	9,681			5,303.0	9,696	0.191	0.057	163.81	
	20	15,358	13,729	13,733			7,583.5	13,785	0.277	16.975	204.938	
	30	24,709	22,741	21,917			12,134.0	21,998	0.456	0.058	91.825	
	50	43,538	39,259	38,409			21,446.5	38,809	0.865	12.318	130.075	
	100	92,632	81,936	79,532			45,058.0	81,316	1.857	25.948	136.857	
	200	190,718	161,888				92,770.0	167,676	3.996	49.795		

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
ego-Twitter	2	1,775	628	612		81.0	491.5	612	0.516	1.222	0.206	
	3	3,316	2,141	1,962		264.5	1,354.0	1,952	0.803	1.997	102.102	
	4	4,894	2,785	2,665		579.0	1,903.0	2,657	0.898	3.212	119.446	
	5	6,364	4,599	4,572		1,029.5	3,021.0	4,560	0.548	3.671	18.201	
	7	10,213	7,372	7,279		1,923.0	4,796.0	7,269	0.87	3.664	35.818	
	9	13,273	9,799	9,409		3,172.0	6,174.0	9,364	0.82	20.312	119.22	
	10	14,951	11,471	11,326		3,972.0	7,286.5	11,310	0.926	7.791	42.45	
	15	26,429	23,042	20,707		8,326.0	12,963.0	20,682	1.213	4.567	13.591	
	20	37,914	34,129	30,565		13,120.0	18,695.5	30,515	1.628	6.441	16.295	
	30	63,458	57,348	52,676		23,560.5	31,369.0	52,652	2.925	5.504	23.034	
	50	115,267	102,780	97,967			57,168.0	99,249	5.486	22.759	148.076	
	100	253,602	222,423	220,716					14.042	18.032	100.662	
	200	543,858	483,026	483,014					34.096	24.351	54.394	
ego-Gplus	2	12,759	10,370	10,017		2,080.5	5,861.0	9,973	4.904	192.493	293.423	
	3	21,881	16,431			4,164.5	9,634.5	15,344	7.864	90.569		
	4	35,821	27,907	26,594		7,145.5	16,153.5	26,364	9.75	51.202	297.564	
	5	44,248	34,465	32,031		10,211.0	19,996.0	31,761	10.091	62.908	296.905	
	7	71,602	54,864	53,418		17,671.5	32,309.0	52,330	14.957	39.476	298.708	
	9	96,721	81,180	74,916		25,776.0	45,788.0	74,482	20.88	37.802	298.508	
	10	110,292	88,298	85,825		30,064.0	51,825.5	84,148	24.833	50.924	298.627	
	15	176,209	156,431	139,519					44.25	90.034	299.333	
	20	250,151	227,942	203,186					66.332	120.244	299.05	
	30	386,742	396,142						116.445	112.597		
	50	698,224	683,352						237.063	183.1		
	100		1,473,346							290.047		
	200		3,080,726							235.812		

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

		solution size				Pre-	DP bounds		time (in seconds)			
graph	k	Cluster	Greedy	DP	Opt	processed	lower	upper	Cluster	Greedy	DP	Opt
soc-Epinions1	2	2,013	1,597	1,533		607.0	881.0	1,529	0.359	1.184	1.774	
	3	3,855	3,373	3,379		1,296.5	1,912.0	3,372	0.558	1.744	63.439	
	4	6,247	5,389	5,387		2,006.5	3,046.0	5,387	0.598	4.407	3.204	
	5	7,975	6,999	6,862		2,736.0	3,895.5	6,842	0.639	4.905	39.075	
	7	12,150	10,387	10,357		4,422.5	5,900.5	10,355	0.771	5.253	76.364	
	9	16,844	14,849	14,641		6,250.5	8,266.5	14,593	0.934	5.348	10.424	
	10	19,073	16,651	16,616		7,189.5	9,382.0	16,602	1.003	5.27	22.693	
	15	30,392	27,398	26,811		12,160.5	15,008.0	26,797	1.551	4.895	10.198	
	20	42,753	40,227	37,444		17,432.0	21,045.0	37,410	2.111	4.4	62.251	
	30	66,894	60,250	59,253		28,592.0	33,027.0	59,231	3.435	7.278	27.749	
	50	119,350	107,344	106,724			59,347.5	108,164	6.566	29.078	200.973	
	100	253,120	232,126	228,413			126,105.5	230,682	14.94	66.872	179.575	
	200	531,938	484,146	481,556					32.709	36.853	104.9	
soc-Slashdot0811	2	1,341	1,134	971		115.5	587.5	971	0.335	0.733	0.84	
	3	1,921	1,404	1,335		261.5	865.0	1,335	0.555	0.017	0.22	
	4	3,637	3,055	2,897		661.5	1,764.0	2,890	0.561	0.674	29.747	
	5	6,035	4,661	4,662		1,076.5	2,735.0	4,660	0.552	0.025	5.067	
	7	6,771	5,321	5,244		2,156.5	3,235.0	5,244	0.527	0.88	0.127	
	9	10,457	8,698	8,562		3,691.0	5,124.0	8,556	0.63	1.12	6.265	
	10	12,625	10,697	10,469		4,480.0	6,175.5	10,465	0.681	2.804	8.46	
	15	22,529	19,608	19,034		8,544.5	11,018.5	19,034	0.906	0.085	2.596	
	20	31,963	27,959	27,150		13,081.5	15,640.5	27,148	1.436	2.586	5.812	
	30	53,441	46,573	46,232		22,463.5	26,315.5	46,220	2.336	4.091	12.863	
	50	96,084	85,739	84,170			47,578.5	84,619	4.26	19.32	105.61	
	100	207,857	189,899	184,998			103,401.0	185,835	9.558	44.594	146.937	
	200	437,421	393,248	391,176					21.7	18.561	98.263	

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

		solution size				Pre-	DP bounds		time (in seconds)			
graph	k	Cluster	Greedy	DP	Opt	processed	lower	upper	Cluster	Greedy	DP	Opt
soc-Slashdot0902	2	1,323	1,010	1,014		101.5	609.5	1,011	0.36	0.929	0.185	
	3	2,082	1,613	1,610		229.0	998.5	1,592	0.642	1.012	2.902	
	4	3,310	2,586	2,527		576.5	1,576.5	2,527	0.579	0.025	0.339	
	5	5,329	4,594	4,363		989.5	2,587.0	4,363	0.59	0.985	1.102	
	7	7,833	7,188	6,330		1,998.0	3,823.0	6,324	0.587	2.831	22.437	
	9	10,176	8,160	8,056		3,346.5	4,914.5	8,034	0.675	3.086	10.643	
	10	11,862	9,740	9,417		4,104.5	5,682.0	9,415	0.671	2.973	17.437	
	15	22,878	19,499	19,209		8,013.0	11,229.5	19,187	0.918	0.085	13.295	
	20	30,366	26,335	25,627		12,279.0	14,955.0	25,601	1.363	1.795	6.43	
	30	52,363	45,890	44,927		21,590.5	25,835.0	44,913	2.214	3.432	4.069	
	50	95,270	83,902	82,874		41,070.0	47,123.0	82,702	4.015	2.896	4.032	
	100	205,851	182,946	182,579			102,455.0	183,496	9.504	36.63	152.627	
	200	436,917	390,811	388,738					21.53	17.688	77.467	
soc-Pokec	2	10,542	10,260	10,260		2,580.0	5,243.0	10,260	42.01	0.117	1.297	
	3	17,972	17,309	17,309		5,564.0	8,923.0	17,309	41.886	0.141	2.272	
	4	29,568	28,552	28,552		9,646.0	14,699.0	28,552	41.628	0.182	1.299	
	5	39,879	38,505	38,501		14,158.5	19,840.5	38,501	43.324	0.25	3.36	
	7	56,828	54,683	54,636		24,938.0	24,938.0		47.318	0.406	0.476	
	9	83,716	80,838	80,724		37,383.0	41,642.0	80,724	56.566	0.718	3.105	
	10	95,029	91,811	91,808		43,804.5	43,804.5		61.005	0.908	1.01	
	15	161,970	157,173	156,910		77,385.0	80,792.5	156,909	89.431	2.275	3.68	
	20	231,246	224,905	224,199		111,425.5	115,393.5	224,199	118.843	55.83	3.174	
	30	371,071	363,291	360,968		180,350.5	185,194.5	360,959	177.905	78.036	39.389	
	50	655,687	640,907	637,647		318,856.0	327,251.5	637,588	293.881	59.243	109.263	
	100			1,331,159		667,210.5	683,354.5	1,331,159			47.981	
	200											

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
soc-LiveJournal1	2	10,027	8,804			2,578.5	4,846.0	8,535	230.565	36.642		
	3	24,631	19,739	19,654		5,791.0	11,020.0	19,652	219.469	23.877	296.322	
	4	32,165	28,023	27,744		9,240.5	15,631.0	27,742	204.556	15.501	292.405	
	5	53,213	46,368	46,041		12,820.0	25,404.0	46,031	195.452	18.608	296.7	
	7	79,848	68,090	65,412		20,549.0	36,233.0	65,404	184.527	28.904	28.398	
	9	96,795	86,786	85,041		30,793.5	47,175.0	85,023	181.88	28.62	190.217	
	10	116,645	111,733	104,036		36,621.5	57,502.5	104,030	183.757	17.251	290.605	
	15	179,424	177,827	161,429		68,561.5	88,980.0	161,407	198.304	46.469	121.693	
	20	263,242	249,969	238,966		103,848.0	130,180.5	238,936	228.193	60.917	129.567	
	30	427,405	417,653			178,951.0	212,452.0	392,838	294.362	270.837		
	50					339,230.0	339,230.0					
	100					765,096.5	765,096.5					
	200					1,658,724.0	1,658,724.0					
wiki-Vote	2	578	448	440		152.5	277.5	437	0.026	0.447	0.284	
	3	1,184	953	916		312.0	560.0	911	0.024	1.173	0.314	
	4	1,901	1,500	1,476		474.0	898.5	1,472	0.038	0.624	22.317	
	5	2,586	2,020	2,034		640.0	1,233.5	2,018	0.033	1.921	0.75	
	7	3,454	2,711	2,580		1,058.0	1,649.5	2,563	0.048	2.012	27.412	
	9	4,861	3,778	3,750		1,591.5	2,344.0	3,740	0.054	14.341	5.311	
	10	5,589	4,338	4,339		1,877.0	2,706.5	4,335	0.07	3.62	1.984	
	15	9,183	7,452	7,263		3,416.0	4,492.0	7,216	0.11	30.621	87.281	
	20	13,521	10,738	10,614			6,477.0	10,944	0.164	59.4	30.25	
	30	22,097	17,637	17,627			10,772.5	18,258	0.297	19.08	61.347	
	50	38,855	31,652	31,412			19,234.0	32,274	0.566	38.413	108.537	
	100	85,022	68,738	68,590			41,467.5	70,593	1.304	26.851	154.542	
	200	179,040	143,506	143,504			87,912.5	149,307	3.129	8.583	151.873	

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
coPapersDBLP	2	1,890	2,923	1,739		702.5	957.0	1,734	8.446	0.687	27.791	
	3	3,415	3,055	3,052		1,396.5	1,682.0	3,032	9.887	0.062	0.378	
	4	6,242	5,677	5,523		2,205.0	2,993.0	5,506	10.088	1.091	21.759	
	5	9,085	8,383	8,142		3,014.5	4,414.0	8,134	9.428	0.051	3.886	
	7	12,166	10,780	10,644		4,861.5	5,897.0	10,630	9.661	5.618	7.24	
	9	16,793	15,073	14,949		6,913.0	8,203.5	14,946	9.653	5.118	8.194	
	10	19,631	18,184	17,402		7,945.5	9,557.0	17,351	9.871	2.448	9.487	
	15	31,645	28,071	28,073		13,438.5	15,402.5	28,057	10.088	6.158	25.244	
	20	43,637	46,542	39,130		19,175.0	21,469.5	39,112	10.397	9.046	16.795	
	30	70,567	63,786	63,089		30,939.5	34,714.0	63,036	11.362	11.527	16.92	
	50	122,202	110,694	110,503		55,359.5	60,525.5	110,495	13.654	7.598	11.437	
	100	258,230	234,102	234,064		119,477.0	128,146.0	234,018	21.297	24.581	19.135	
	200	540,505	486,865	486,841		251,792.0	268,413.5	486,775	39.303	63.725	76.172	
coPapersCiteseer	2	203	82	79	79	22.5	79.0	79	6.941	0.027	0.143	263.146
	3	394	174	136	136	39.5	136.0	136	8.974	0.023	2.693	0.047
	4	666	280	231	231	86.0	231.0	231	9.674	0.026	0.341	0.062
	5	998	588	326	326	142.0	326.0	326	9.612	0.034	0.036	0.107
	7	1,925	809	657	657	233.0	657.0	657	9.762	0.042	0.07	0.203
	9	2,265	1,015	859	859	344.0	859.0	859	9.223	0.055	0.102	0.290
	10	2,541	1,260	958	958	398.5	958.0	958	9.16	0.07	0.249	0.347
	15	5,147	2,252	1,845	1,845	731.5	1,845.0	1,845	8.815	0.074	0.164	1.079
	20	6,829	3,312	2,627	2,627	1,121.5	2,627.0	2,627	8.489	0.078	0.136	1.767
	30	11,679	4,871	4,273	4,273	2,039.5	4,273.0	4,273	7.842	0.101	0.157	3.898
	50	22,910	10,991	9,310	9,310	4,458.5	9,310.0	9,310	7.221	0.194	0.365	17.459
	100	51,318	25,456	22,006	22,006	10,673.0	22,006.0	22,006	6.235	0.471	1.162	97.355
	200	112,177	59,642	51,361	51,361	25,439.0	51,361.0	51,361	5.451	1.071	6.612	541.497

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

		solution size				Pre-	DP bounds		time (in seconds)			
graph	k	Cluster	Greedy	DP	Opt	processed	lower	upper	Cluster	Greedy	DP	Opt
coAuthorsDBLP	2	97	72	62		27.0	61.0	61	1.076	0.026	0.02	0.350
	3	253	181	179	179	60.0	179.0	179	1.355	0.026	0.024	0.071
	4	344	231	231	231	95.5	231.0	231	1.271	0.023	0.02	194.462
	5	531	325	319	317	144.5	317.0	317	1.273	0.025	0.027	266.860
	7	818	643	528		258.0	410.0	528	1.183	0.021	0.026	
	9	1,199	751	748		379.0	572.5	748	1.135	0.024	0.03	
	10	1,372	986	871		444.5	696.0	871	1.142	0.022	0.026	
	15	2,357	1,605	1,486		786.5	1,144.5	1,486	1.086	0.027	1.164	
	20	3,323	2,091	2,091		1,169.0	1,618.0	2,091	1.006	0.028	0.028	
	30	5,375	3,675	3,405		1,892.5	2,580.5	3,403	0.924	0.031	2.204	
	50	9,661	6,356	6,091		3,585.0	4,746.0	6,085	0.84	0.035	3.814	
	100	21,273	12,355	12,255		8,222.5	10,581.0	12,253	1.103	0.057	2.582	
	200	45,242	22,711	21,960	21,960	18,531.0	21,960.0	21,960	0.846	0.079	0.971	9.271
citationCiteseer	2	689	489	457	457	65.5	457.0	457	2.133	0.495	0.41	2.637
	3	1,187	856	698	698	154.5	698.0	698	2.21	0.031	0.033	3.274
	4	1,689	1,191	1,111	1,111	342.5	1,111.0	1,111	1.965	0.025	0.028	5.219
	5	2,508	1,936	1,764		560.0	1,225.0	1,756	1.938	0.039	0.244	
	7	3,866	2,740	2,654		1,017.0	1,814.5	2,654	1.701	0.033	0.037	
	9	5,854	4,467	4,237		1,583.5	2,820.5	4,229	1.598	0.039	1.427	
	10	6,532	4,971	4,775		1,880.5	3,171.0	4,771	1.61	0.047	0.048	
	15	10,491	9,794	7,960		3,497.5	5,117.0	7,932	1.488	0.051	0.201	
	20	15,879	12,540	12,065		5,270.0	7,770.0	12,065	1.412	0.059	0.102	
	30	24,191	19,393	18,800		9,346.0	11,835.5	18,800	1.425	0.08	0.085	
	50	45,288	36,418	36,004		18,360.0	22,320.0	36,004	1.677	0.185	0.169	
	100	98,804	82,393	81,464		43,131.0	49,046.5	81,464	2.813	0.396	2.266	
	200	212,152	174,114	174,114		96,784.5	105,146.0	174,114	5.34	0.78	1.709	

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

		solution size				Pre-	DP bounds		time (in seconds)			
graph	k	Cluster	Greedy	DP	Opt	processed	lower	upper	Cluster	Greedy	DP	Opt
coAuthorsCiteseer	2	1,163	1,025	1,002	1,002	448.0	1,002.0	1,002	0.617	0.018	0.015	0.937
	3	2,156	1,836	1,836	1,836	890.0	1,836.0	1,836	0.868	0.016	0.039	13.832
	4	3,103	5,160	2,978	2,978	1,382.5	2,978.0	2,978	0.954	0.017	0.016	153.027
	5	4,415	5,267	4,164	4,164	1,876.0	4,164.0	4,164	0.88	0.018	0.673	179.957
	7	6,304	6,059	5,995	5,982	2,864.0	5,982.0	5,982	0.941	0.02	0.019	1,638.210
	9	8,478	8,156	8,099	8,097	3,966.5	8,097.0	8,097	0.97	0.025	1.406	1,747.073
	10	9,716	9,286	9,286	9,286	4,531.0	9,286.0	9,286	0.968	0.023	0.023	1,482.391
	15	15,844	14,954	14,937		7,393.5	7,923.0	14,937	1.075	0.036	0.038	
	20	21,640	20,602	20,589		10,384.0	10,847.5	20,589	1.22	0.055	0.062	
	30	34,108	33,613	32,429	32,429	16,451.5	32,429.0	32,429	1.535	0.103	1.136	817.614
	50	59,307	56,014	56,014		28,572.5	29,522.0		2.194	0.177	0.198	
	100	122,996	118,059	114,586		59,632.0	61,296.0	114,586	3.818	0.451	3.656	
	200	252,440	229,745	226,517		122,772.0	125,961.0	226,511	7.284	8.397	15.177	
graph_thres_01	2	290	181	178		69.5	176.0	176	6.472	0.049	0.121	583.207
	3	748	484	480		185.0	357.5	478	5.72	0.056	0.054	
	4	1,153	731	718		304.5	546.5	715	5.47	0.139	0.131	
	5	1,620	1,007	994		430.5	745.0	991	5.038	4.888	0.174	
	7	2,518	1,674	1,606		710.0	1,186.5	1,606	4.811	0.227	4.259	
	9	3,350	2,213	2,206		992.0	1,635.5	2,206	4.566	0.055	6.959	
	10	3,918	2,557	2,538		1,137.0	1,853.5	2,538	4.393	0.062	0.079	
	15	6,458	4,404	4,303		1,900.5	3,153.0	4,303	3.96	1.215	3.737	
	20	8,773	6,286	5,793		2,760.0	4,227.5	5,793	3.786	0.939	0.067	
	30	13,643	9,502	9,389		4,652.5	6,635.5	9,354	3.439	4.622	0.105	
	50	24,449	17,290	17,010		8,898.5	11,960.0	17,010	3.226	4.601	5.075	
	100	53,101	38,087	38,087		21,173.5	26,298.5	38,087	3.067	0.193	0.487	
	200	115,381	80,992	80,581		49,481.5	56,962.5	80,571	3.381	0.446	18.312	

Table A.1.: Full list of experimental results on real-world instances with enabled data reduction, see page 215 for the description of the values.

graph	k	solution size Cluster	Greedy	DP	Opt	Pre-processed	DP bounds lower	upper	time (in seconds) Cluster	Greedy	DP	Opt
graph_thres_02	2	32	17	16	16	4.0	16.0	16	0.525	0.031	0.011	0.016
	3	111	57	50	50	16.0	50.0	50	0.833	0.024	0.013	0.015
	4	165	100	79	79	28.5	79.0	79	0.83	0.024	0.016	0.015
	5	258	123	115	115	43.0	115.0	115	0.764	0.009	0.026	0.028
	7	392	196	177	177	78.0	177.0	177	0.763	0.025	0.01	0.022
	9	580	273	265	263	119.0	263.0	263	0.726	0.012	0.027	0.047
	10	612	316	288	288	143.0	288.0	288	0.681	0.011	0.025	0.042
	15	1,095	552	505	505	286.0	505.0	505	0.646	0.026	2.516	3.132
	20	1,564	758	758	758	455.0	758.0	758	0.653	0.027	0.029	11.533
	30	2,584	1,417	1,305	1,305	834.5	1,305.0	1,305	0.613	0.03	2.056	383.908
	50	5,094	2,568	2,410	2,410	1,695.5	2,410.0	2,410	0.536	0.031	0.049	0.350
	100	11,274	5,443	5,419	5,419	4,180.0	5,419.0	5,419	0.529	0.024	0.154	1.109
	200	24,771	12,192	11,921	11,921	9,950.5	11,921.0	11,921	0.809	0.044	0.74	3.446
graphConference	2	11	5	5	5	3.0	5.0	5	0.009	0.001	0.001	0.001
	3	17	9	9	9	5.0	9.0	9	0.006	0.002	0	0.002
	4	44	26	15	15	7.5	15.0	15	0.009	0.001	0.001	0.001
	5	72	31	28	28	10.5	28.0	28	0.009	0.002	0	0.005
	7	110	46	44	44	17.5	44.0	44	0.003	0.002	0.002	0.008
	9	128	77	54	54	25.5	54.0	54	0.009	0.003	0.001	0.003
	10	191	104	82	82	31.5	82.0	82	0.008	0.001	0.002	0.003
	15	323	129	128	128	65.0	128.0	128	0.008	0.003	0.004	0.004
	20	478	223	208	208	108.0	208.0	208	0.009	0.002	0.005	0.018
	30	875	357	342	342	206.5	342.0	342	0.01	0.004	0.007	0.021
	50	1,432	720	647	647	456.5	647.0	647	0.012	0.003	0.009	0.039
	100	3,308	1,627	1,563	1,563	1,224.0	1,563.0	1,563	0.04	0.003	0.033	0.078
	200	7,960	3,863	3,515	3,515	3,127.0	3,515.0	3,515	0.156	0.003	0.109	0.187

Universitätsverlag der TU Berlin

Degree-Constrained Editing of Small-Degree Graphs

This thesis deals with two degree-constrained graph modification problems, namely DAG Realization and Degree Anonymity. The DAG Realization problem is, given a multiset of positive integer pairs, to decide whether there is a realizing directed acyclic graph (DAG). That is, pairs are one-to-one assigned to vertices such that the indegree and the outdegree of every vertex coincides with the two integers of the assigned pair. The Degree Anonymity problem is, given an undirected graph G and two positive integers k and s, to decide whether at most s graph modification operations can be performed in G in order to obtain a k-anonymous graph. A graph is k-anonymous if for each vertex there are k-1 other vertices with the same degree. Both problems are shown to be NP-complete, that is, there are presumably no polynomial-time algorithms that can solve every instance of these problems. Confronted with this worst-case intractability, the goal is to develop fixed-parameter algorithms where the super-polynomial dependency in the running time is confined to a parameter of the input. If the parameter is small, then the corresponding fixed-parameter algorithm is fast. Using the parameter maximum vertex degree, fixed-parameter algorithms are shown for DAG Realization and the version of Degree Anonymity that allows only edge insertions.

ISBN 978-3-7983-2761-0 (print)
ISBN 978-3-7983-2762-7 (online)

ISBN 978-3-7983-2761-0

http://verlag.tu-berlin.de