Win-Win Kernelization for Degree Sequence Completion Problems

Vincent Froese¹, André Nichterlein¹, and Rolf Niedermeier¹

¹Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany, {vincent.froese, andre.nichterlein, rolf.niedermeier}@tu-berlin.de

Abstract

We study the kernelizability of a class of NP-hard graph modification problems based on vertex degree properties. Our main positive results refer to NP-hard graph completion (that is, edge addition) cases while we show that there is no hope to achieve analogous results for the corresponding vertex or edge deletion versions. Our algorithms are based on a method that transforms graph completion problems into efficiently solvable number problems and exploits $f$-factor computations for translating the results back into the graph setting. Indeed, our core observation is that we encounter a win-win situation in the sense that either the number of edge additions is small (and thus faster to find) or the problem is polynomial-time solvable. This approach helps in answering an open question by Mathieson and Szeider [JCSS 2012].

1 Introduction

In this work, we propose a general approach for achieving polynomial-size problem kernels for a class of graph completion problems where the goal graph has to fulfill certain degree properties. Thus, we explore and enlarge results on provably effective polynomial-time preprocessing for these NP-hard graph problems. To a large extent, the initial motivation for our work comes from studying the NP-hard graph modification problem \textsc{Degree Constraint Editing($S$) (DCE($S$))}

Input: An undirected graph $G = (V,E)$, two integers $k, r > 0$, and a “degree list function” $\tau: V \to \{0, \ldots, r\}$.

Question: Is it possible to obtain a graph $G' = (V', E')$ from $G$ using at most $k$ editing operations of type(s) as specified by $S$ such that $\deg_{G'}(v) \in \tau(v)$ for all $v \in V'$?

Mathieson and Szeider [31] originally introduced a weighted version of the problem, where the vertices and edges can have positive integer weights incurring a cost for each editing operation. Here, we focus solely on the unweighted version.
In our work, the set $S$ always consists of a single editing operation. Our studies focus on the two most natural parameters: the number $k$ of editing operations and the maximum allowed degree $r$. We will show that, although all three variants are NP-hard, DCE($e^+$) is amenable to a generic kernelization method we propose. This method is based on dynamic programming solving a corresponding number problem and $f$-factor computations. For DCE($e^-$) and DCE($v^-$), however, we show that there is little hope to achieve analogous results.

Previous Work. There are basically two fundamental starting points for our work. First, there is our previous theoretical work on degree anonymization in social networks [22] motivated and strongly inspired by a preceding heuristic approach due to Liu and Terzi [27]. Indeed, our previous work for degree anonymization very recently inspired empirical work with encouraging experimental results [23]. A fundamental contribution of this work now is to systematically reveal what the problem-specific parts (tailored towards degree anonymization) and what the “general” parts of that approach are. In this way, we develop this approach into a general method of significantly wider applicability for a large number of graph completion problems based on degree properties. The second fundamental starting point is Mathieson and Szeider’s work [31] on DCE($S$). They showed several parameterized preprocessing (also known as kernelization) results and left open whether it is possible to reduce DCE($e^+$) in polynomial time to a problem kernel of size polynomial in $r$—we will affirmatively answer this question. Finally, Golovach [18] achieved a number of kernelization results for closely related graph editing problems; his methods, however, significantly differ from ours.

From a more general perspective, all these considerations fall into the category of “graph editing to fulfill degree constraints”, which recently received increased interest in terms of parameterized complexity analysis [7, 15, 18, 32].

Our Contributions. Answering an open question of Mathieson and Szeider [31], we present an $O(kr^2)$-vertex kernel for DCE($e^+$) which we then transfer into an $O(r^5)$-vertex kernel using a strategy rooted in previous work [22, 27]. A further main contribution of our work in the spirit of meta kernelization [2] is to clearly separate problem-specific from problem-independent aspects of this strategy, thus making it accessible to a much wider class of degree sequence completion problems. We observe that in case that the goal graph shall have “small” maximum degree $r$, then the actual graph structure is in a sense negligible and thus allows for a lot of freedom that can be algorithmically exploited. This paves the way to a win-win situation of either having guaranteed a small number of edge additions or the overall problem being solvable in polynomial-time anyway.

Besides our positive kernelization results, we exclude polynomial-size problem kernels for DCE($e^-$) and DCE($v^-$) subject to the assumption that NP $\not\subseteq$ coNP/poly, thereby showing that the exponential-size kernel results by Mathieson and Szeider [31] are essentially tight. In other words, this demonstrates that in our context edge completion is much more amenable to kernelization than edge deletion or vertex deletion are. We also prove NP-hardness of DCE($v^-$) and DCE($e^+$) for graphs of maximum degree three, implying that the maximum degree is not a useful parameter for kernelization purposes. Last but not least, we develop a general preprocessing approach for Degree Sequence Completion problems which yields a search space size that is polynomially bounded in the
parameter. While this per se does not give polynomial kernels, we derive fixed-parameter tractability with respect to the combined parameter maximum degree and solution size. The usefulness of our method is illustrated by further example degree sequence completion problems.

Notation. All graphs in this paper are undirected, loopless, and simple (that is, without multiple edges). For a graph $G = (V,E)$, we set $n := |V|$ and $m := |E|$. The degree of a vertex $v \in V$ is denoted by $\deg_G(v)$, the maximum vertex degree by $\Delta_G$, and the minimum vertex degree by $\delta_G$. For a finite set $U$, we denote with $\binom{U}{2}$ the set of all size-two subsets of $U$. We denote by $G := (V, \binom{V}{2} \setminus E)$ the complement graph of $G$. For a vertex subset $V' \subseteq V$, the subgraph induced by $V'$ is denoted by $G[V']$. For an edge subset $E' \subseteq \binom{V}{2}$, $V(E')$ denotes the set of all endpoints of edges in $E'$ and $G[E'] := (V(E'), E')$. For a set $E'$ of edges with endpoints in a graph $G$, we denote by $G + E' := (V, E \cup E')$ the graph that results by inserting all edges in $E'$ into $G$. Similarly, we define for a vertex set $V' \subseteq V$, the graph $G - V' := G[V \setminus V']$. For each vertex $v \in V$, we denote by $N_G(v)$ the open neighborhood of $v$ in $G$ and by $N_G[v] := N_G(v) \cup \{v\}$ the closed neighborhood. We omit subscripts if the corresponding graph is clear from the context. A vertex $v \in V$ with $\deg(v) \in \tau(v)$ is called satisfied (otherwise unsatisfied). We denote by $U \subseteq V$ the set of all unsatisfied vertices, formally $U := \{v \in V \mid \deg_G(v) \notin \tau(v)\}$.

Parameterized Complexity. This is a two-dimensional framework for studying computational complexity [12, 16, 34]. One dimension of a parameterized problem is the input size $s$, and the other one is the parameter (usually a positive integer). A parameterized problem is called fixed-parameter tractable (fpt) with respect to a parameter $\ell$ if it can be solved in $f(\ell) \cdot s^{O(1)}$ time, where $f$ is a computable function only depending on $\ell$. This definition also extends to combined parameters. Here, the parameter usually consists of a tuple of positive integers $(\ell_1, \ell_2, \ldots)$ and a parameterized problem is called fpt with respect to $(\ell_1, \ell_2, \ldots)$ if it can be solved in $f(\ell_1, \ell_2, \ldots) \cdot s^{O(1)}$ time.

A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by data reduction [1, 19, 28]. Here, the goal is to transform a given problem instance $I$ with parameter $\ell$ in polynomial time into an equivalent instance $I'$ with parameter $\ell' \leq \ell$ such that the size of $I'$ is upper-bounded by some function $g$ only depending on $\ell$. If this is the case, we call $I'$ a (problem) kernel of size $g(\ell)$. If $g$ is a polynomial, then we speak of a polynomial kernel. Usually, this is achieved by applying polynomial-time executable data reduction rules. We call a data reduction rule $R$ correct if the new instance $I'$ that results from applying $R$ to $I$ is a yes-instance if and only if $I$ is a yes-instance. The whole process is called kernelization. It is well known that a parameterized problem is fixed-parameter tractable if and only if it has a problem kernel.

2 Degree Constraint Editing

Mathieson and Széider [31] showed fixed-parameter tractability of DCE$(S)$ for all non-empty subsets $S \subseteq \{v^-, e^-, e^+\}$ with respect to the combined parameter $(k,r)$ and W[1]-hardness with respect to the single parameter $k$. The fixed-parameter tractability is in a sense tight as Cornuéjols [11] proved that DCE$(e^-)$ is NP-hard on planar graphs with maximum degree three and with $r = 3$ and
thus presumably not fixed-parameter tractable with respect to \( r \). We complement his result by showing that \( \text{DCE}(v^{-}) \) is NP-hard on cubic (that is three-regular) planar graphs, even if \( r = 0 \), and that \( \text{DCE}(e^{+}) \) is NP-hard on graphs with maximum degree three.

**Theorem 1.** \( \text{DCE}(v^{-}) \) is NP-hard on cubic planar graphs, even if \( r = 0 \).

**Sketch.** We provide a polynomial-time many-one reduction from the NP-hard Vertex Cover on cubic planar graphs [17]. Let \((G = (V,E), h)\) be a Vertex Cover instance with the cubic planar graph \( G \). It is not hard to see that \((G, h, 0, \tau)\) with \( \tau(v) = \{0\} \) for all \( v \in V \) is a yes-instance of \( \text{DCE}(v^{-}) \).

**Theorem 2.** \( \text{DCE}(e^{+}) \) is NP-hard on planar graphs with maximum degree three.

**Proof.** We provide a polynomial-time many-one reduction from the NP-hard Independent Set problem on cubic planar graphs [17]. Given an Independent Set instance \((G = (V,E), h)\), we construct a \( \text{DCE}(e^{+}) \) instance \((G', k, h, \tau)\) as follows: Start with \( G' \) as a copy of \( G \), add a new vertex \( v \) to \( G' \) and set \( \tau(v) := \{h\} \). Furthermore, for all other vertices \( u \in V \setminus \{v\} \) set \( \tau(u) = \{3, 3 + h\} \). Finally, set \( k := \left(\frac{h}{2}\right) + h \). It is straightforward to argue that the only way of satisfying \( v \) within the given budget is to connect it to \( h \) vertices forming an independent set.

In contrast to \( \text{DCE}(e^{-}) \) and \( \text{DCE}(v^{-}) \), unless \( P = NP \), \( \text{DCE}(e^{+}) \) cannot be NP-hard for constant values of \( r \) since we later show fixed-parameter tractability for \( \text{DCE}(e^{+}) \) with respect to the parameter \( r \).

**Excluding Polynomial Kernels.** Mathieson and Szeider [31] gave exponential-size problem kernels for \( \text{DCE}(v^{-}) \) and \( \text{DCE}((v^{-}, e^{-})) \) with respect to the combined parameter \((k, r)\). We prove that these results are tight in the sense that, under standard complexity-theoretic assumptions, neither \( \text{DCE}(e^{-}) \) nor \( \text{DCE}(v^{-}) \) admits a polynomial-size problem kernel when parameterized by \((k, r)\).

**Theorem 3.** \( \text{DCE}(e^{-}) \) does not admit a polynomial-size problem kernel with respect to \((k, r)\) unless \( NP \subseteq \text{coNP/poly} \).

**Proof.** We provide a polynomial time and parameter transformation from the CLIQUE problem parameterized by the “vertex cover number”. Given a graph \( G = (V,E) \) and a positive integer \( h \), the CLIQUE problem asks for a subset of at least \( h \) vertices that are pairwise adjacent, whereas the Vertex Cover problem asks for a subset \( V' \) of at most \( h \) vertices such that each edge in \( E \) has at least one endpoint in \( V' \). Given a parameterized problem instance \((I, k)\), a polynomial time and parameter transformation yields an equivalent instance \((I', p(k))\) in time \( g(|I|) \) for polynomials \( p \) and \( q \) [4]. Since CLIQUE parameterized by the “vertex cover number” does not admit a polynomial-size problem kernel [3], the statement of the theorem follows.

To this end, let \((G = (V,E), h)\) be the CLIQUE instance and let \( X \subseteq V \) be a minimum vertex cover of \( G \). If not given explicitly, then \( X \) can be efficiently computed with a simple factor-2 approximation which contains all endpoints of a maximal matching. We assume without loss of generality that each vertex in \( G \) has degree at least \( h \). Note that any clique in \( G \) has size at most \( |X| + 1 \). 

4
Figure 1: An example of the construction. The given graph is displayed at the left side with highlighted vertex cover (VC), independent set (IS), and twin classes in the independent set. The constructed graph is depicted on the right side. The three vertices in the upper half of each subgraph $G_i$, $1 \leq i \leq 4$, correspond to the vertex cover $v_1, v_2, v_3$. The fourth vertex in the lower half of each subgraph correspond to a vertex from the twin class $C_i$.

The precise construction of the DCE($e^-$) instance $(G' = (V', E'), k, r, \tau)$ works as follows: For $i \in \{1, \ldots, \ell\}$, let $v_i$ be an arbitrary vertex of the twin class $C_i$. Furthermore, let $G_i := G[X \cup \{v_i\}]$. Initialize $G'$ as the disjoint union of the graphs $G_i$, $1 \leq i \leq \ell$. Observe that $G'$ contains $\ell$ copies of the vertex cover $X$. Next, add a binary tree of height $\lceil \log \ell \rceil \leq |X|$ with leaves $u_1, \ldots, u_\ell$ to $G'$. For each $i \in \{1, \ldots, \ell\}$, make $u_i$ adjacent to all vertices in $G_i$. This completes the construction of $G'$; see Figure 1 for an example. Set $k := (\binom{h}{2}) + h + \lceil \log \ell \rceil$. For each $i \in \{1, \ldots, \ell\}$ and each vertex $v \in G_i$, set $\tau(v) := \{\deg_{G'}(v), \deg_{G'}(v) - h\}$. For each leaf $u_i$, set $\tau(u_i) := \{\deg_{G'}(u_i), \deg_{G'}(u_i) - h - 1\}$. For each inner vertex $w$ in the binary tree, set $\tau(w) := \{3, 1\}$. Finally, for the root $v$ of the binary tree, set $\tau(v) := \{1\}$. Observe that $r := \max_{u \in V} \max \tau(u) \leq |X| + 2$ and $k \in O(|X|^2)$. Moreover, the above construction requires polynomial time.

It remains to show the correctness of our construction, that is, $I := (G, h)$
is a yes-instance of \textsc{Clique} if and only if \( I' := (G', k, r, \tau) \) is a yes-instance of \textsc{DCE}(e\(^-\)).

\( \Rightarrow \) Let \((G, h)\) be a yes-instance, that is, \( G \) contains a clique \( C \) of size \( h \). If \( C \) contains a vertex from \( V \setminus X \), then let \( C' \) denote its twin class (recall that there is at most one such vertex in \( C \)). Otherwise, set \( i := 1 \). Now, let \( E'' \subseteq E' \) contain all edges between copies of the clique vertices in \( G_i \). Moreover, let \( E'' \) contains all edges between \( u_i \) and the clique vertices in \( G_i \) and all edges along the path from \( u_i \) to the root \( v \). The overall number of edges in \( E'' \) is at most \( k \) and it can easily be verified that, for each \( v \in V' \), it holds \( \text{deg}_{G' - E''}(v) \in \tau(v) \).

Thus, \((G', k, r, \tau)\) is a yes-instance.

\( \Leftarrow \) Let \((G', k, r, \tau)\) be a yes-instance. Note that since \( \text{deg}_{G'}(v) = 2 \) and \( \tau(v) = \{1\} \), one of the two edges incident to \( v \) has to be deleted. Moreover, for all inner nodes \( w \) of the binary tree, we have \( \tau(w) = \{3, 1\} \), which ensures that any solution deletes either 0 or 2 edges incident to \( w \). Consequently, every solution deletes all edges on one particular path from the root \( v \) to some leaf \( u_i \).

This requires \(|X|\) edge deletions. Now consider the leaves of the binary tree. Their degree constraints are chosen in such a way that any solution either deletes no edges or exactly \( h + 1 \) edges incident to a leaf vertex. Thus, for the leaf \( u_i \) at hand, it holds that \( h \) more edges from \( u_i \) to \( G_i \) are deleted in a solution. Finally, we are left with a budget of \( \binom{h}{2} \) edge deletions in order to decrease the degrees of all \( h \) affected vertices in \( G_i \) by \( h - 1 \). This is only possible if they form a clique in \( G_i \), which in turn corresponds to a clique in \( G \). \( \square \)

**Theorem 4.** \textsc{DCE}(v\(^-\)) does not admit a polynomial-size problem kernel with respect to \((k, r)\) unless \( \text{NP} \subseteq \text{coNP/poly} \).

\textbf{Proof.} We adjust our construction from Theorem 3 as follows: In the binary tree connecting all subgraphs \( G_i \), make for each inner vertex \( v \) its two children adjacent. Furthermore, change the degree lists of all inner vertices except the root from \([1, 3]\) to \([2, 4]\). The idea is that if a parent vertex is deleted, then one of its two children also has to be deleted in order to satisfy the remaining child vertex. To ensure that the root \( v \) with \( \text{deg}(v) = 2 \) is deleted, set \( \tau(v) := \{3\} \).

In this way, the selection of the subgraph \( G_i \) via the binary tree works as in the reduction for \textsc{DCE}(e\(^-\)). As edges cannot be deleted any more, we also have to adjust our construction at the subgraphs \( G_i \). For each leaf \( u_i \) of the binary tree, remove the edges between \( u_i \) and the vertices in \( G_i \), and add a new vertex \( u_i' \) that is adjacent to \( u_i \) and all vertices in \( G_i \). Furthermore, add for each \( i \in \{1, \ldots, \ell\} \), a clique \( C_i \) with \(|X|^2\) vertices and make \( u_i' \) adjacent to all vertices in \( C_i \). For each vertex \( w \in C_i \), set \( \tau(w) := \{\text{deg}(w)\} \). Furthermore, set \( \tau(u_i) := \{1, 3\} \) and \( \tau(u_i') := \{\text{deg}(u_i'), |X|^2 + h\} \). For each vertex \( w \in V(G_i) \), set \( \tau(w) := \{\text{deg}(w), h\} \).

Finally, set \( k := \lceil \log \ell \rceil + |X| + 1 - h \). Observe that \( k \in O(|X|) \) and \( r \in O(|X|^2) \). This construction requires polynomial time.

It remains to show the correctness of our construction, that is, \( I := (G, h) \) is a yes-instance of \textsc{Clique} if and only if \( I' := (G', k, r, \tau) \) is a yes-instance of \textsc{DCE}(v\(^-\)).

\( \Rightarrow \) Let \( C \subset V \) be a clique of size \( h \) in \( G \). As \( C \) can contain at most one vertex from \( V \setminus X \), there is a subgraph \( G_i \) in \( G' \) such that the vertices corresponding to \( C \) are also contained in \( G_i \). Now, remove all other vertices in \( G_i \) and all vertices on the shortest path from \( u_i \) to the root \( v \) of the binary tree. Overall, we removed at most \(|X| + 1 - h + |\log \ell| = k \) vertices. Furthermore,
observe that in the remaining graph all vertices are satisfied, implying that $I'$ is a yes-instance.

\textbf{``⇐:``} Assume that $C \subseteq V'$ is a solution for $I'$, that is, each vertex in $G' - C$ is satisfied and $|C| \leq k$. First, observe that the root $v$ of the binary tree is contained in $C$. We now show that we can assume that exactly one of the two children $v_1, v_2$ of $v$ is also contained in $C$: Suppose that neither $v_1$ nor $v_2$ are contained in $C$. Hence, at least one child-vertex $v_1'$ of $v_1$ and one child-vertex $v_2'$ of $v_2$ are contained in $C$ since otherwise $v_1$ or $v_2$ would not be satisfied in $G - C$. Denote with $v_i''$ the second child-vertex of $v_i$. We create a solution $C'$ for $I'$ such that $|C| \geq |C'|$ by setting $C' := (C \cup \{v_1\}) \setminus \{v_2'\}$ and removing from $C$ all vertices in the subtrees with root $v_2$ or with root $v_1''$, that is, all vertices that are in the same connected component with $v_2$ or $v_1''$ in $G' \setminus \{v, v_1, v_1'\}$. As every vertex except $v$ is satisfied in $G'$, $C$ is a solution for $I'$, and $v_2$ and $v_1''$ are satisfied in $G' - C'$, it follows that $C'$ is also a solution for $I'$. By iteratively applying this procedure to all inner vertices of the binary tree, we can assume that in this binary tree exactly the shortest path from $v$ to one leaf, say $u$, is contained in the solution $C$. Since $|C_1| > k$, $\tau(w) = \{\deg(w)\}$ for all $w \in C_1$, and $w'$ is adjacent to all vertices in $C_1$, it follows that $w' \notin C$. As $u \in C$, this implies that all but $h$ vertices in $G_1$ are contained in $C$. Since for each $w$ of these $h$ remaining vertices it holds $\tau(w) = \{\deg(w), h\}$, it follows that they form a clique of order $h$. Thus, $I$ is a yes-instance.

Having established these computational lower bounds, we now show that in contrast to $\text{DCE}(e^-)$ and $\text{DCE}(v^-)$, $\text{DCE}(e^+)$ admits a polynomial kernel.

### 2.1 A Polynomial Kernel for $\text{DCE}(e^+)$ with respect to $(k, r)$

In order to describe the kernelization, we need some further notation: For $i \in \{0, \ldots, r\}$, a vertex $v \in V$ is of \textit{type} $i$ if and only if $\deg(v) + i \in \tau(v)$, that is, $v$ can be satisfied by adding $i$ edges to it. The set of all vertices of type $i$ is denoted by $T_i$. Observe that a vertex can be of multiple types, implying that for $i \neq j$ the vertex sets $T_i$ and $T_j$ are not necessarily disjoint. Furthermore, notice that the type-0 vertices are exactly the satisfied ones. We remark that there are instances for $\text{DCE}(e^+)$ where we might have to add edges between two satisfied vertices (though this may seem counter-intuitive): Consider, for example, a three-vertex graph without any edges, the degree list function values are $\{2\}, \{0, 2\}, \{0, 2\}$, and $k = 3$. The two vertices with degree list $\{0, 2\}$ are satisfied. However, the only solution for this instance is to add all edges.

Now, we can describe our kernelization algorithm: The basic strategy is to keep the unsatisfied vertices $U$ and “enough” arbitrary vertices of each type (from the satisfied vertices) and delete all other vertices. The idea behind the correctness is that the vertices in a solution are somehow “interchangeable”. If an unsatisfied vertex needs an edge to a satisfied vertex of type $i$, then it is not important which satisfied type-$i$ vertex is used. We only have to take care not to “reuse” the satisfied vertices to avoid the creation of multiple edges.

Next, we specify what we mean by “enough” vertices: The “magic number” is $\alpha := k(\Delta_G + 2)$. This leads to the definition of $\alpha$-type sets: An $\alpha$-type set $C \subseteq V$ is a vertex subset containing all unsatisfied vertices $U$ and $\min\{\alpha, |T_i \setminus U|\}$ type-$i$ vertices from $T_i \setminus U$ for each $i \in \{1, \ldots, r\}$. We will soon show that for any fixed $\alpha$-type set $C$, deleting all vertices in $V \setminus C$ results in an equivalent instance.
We first prove the correctness of the reduction rule. To this end, the Reduction Rule 1 is correct and can be applied in linear time.

**Proof.**

Let \( (G = (V,E),k,r,\tau) \) be an instance of DCE\((\ell^+)\) and let \( C \subseteq V \) be an \( \alpha \)-type set in \( G \). Then, safely remove all vertices in \( V \setminus C \).

**Lemma 1.** Reduction Rule 1 is correct and can be applied in linear time.

**Proof.** We first prove the correctness of the reduction rule. To this end, the given DCE\((\ell^+)\) instance is denoted by \( I := (G = (V,E),k,r,\tau) \). We fix any \( \alpha \)-type set \( C \subseteq V \). Furthermore, denote by \( I' \) the resulting instance when safely removing \( V \setminus C \), formally, \( I' := (G[C],k,r,\tau_{V \setminus C}) \). As all vertices in \( V \setminus C \) are satisfied, it follows that any edge set that is a solution for \( I' \) is also a solution for \( I \). Hence, if \( I' \) is a yes-instance, then also \( I \) is a yes-instance. To complete the correctness proof, it remains to prove the reverse direction.

Let \( E' \subseteq \binom{V}{2} \setminus E \) be a solution for \( I \), that is, \( \forall v \in V : \deg_{G+E'}(v) \in \tau(v) \). Observe that if \( V(E') \subseteq C \), then \( E' \) is also a solution for \( I' \). Hence, it remains to consider the case \( V(E') \setminus C \neq \emptyset \). Let \( v \in V(E') \setminus C \). We show how to construct from \( E' \) a solution \( E'' \) for \( I' \) such that \( (V(E') \setminus C) \setminus V(E'') = \emptyset \). Let \( i \leq k \) denote the number of edges in \( E' \) with endpoint \( v \). Since \( v \) is not in the \( \alpha \)-type set \( C \), it follows that \( v \notin U \) and \( |C \cap T_i| = \alpha = k(\Delta_G + 2) \). Next, we show that there is a type-\( i \) vertex \( u \in C \) 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 \( E'' := \{u,w\} \cup \{v,w\} \in E' \cup \{(w_1,w_2) \mid \{w_1,w_2\} \in E' \setminus w_1 \neq v \lor w_2 \neq v\} \), it holds that \( E'' \cap E = \emptyset \) and since \( u \) is also of type \( i \), all degree constraints are satisfied in \( G + E'' \). Hence, it remains to show that such a vertex \( u \) exists, that is, \((C \cap T_i) \setminus (V(E') \cup N_G(N_G[E'](v)))\) is indeed non-empty. This is true since \( |C \cap T_i| = k(\Delta_G + 2) \), whereas \( |V(E') \cup N_G(N_G[E'](v))| < 2k + k\Delta_G \). By iteratively applying this procedure, we obtain a solution for \( I' \). Hence, \( I' \) is a yes-instance if \( I \) is a yes-instance. This proves the correctness.

To compute the \( \alpha \)-type set \( C \) in linear time, initialize \( C := \emptyset \), \( U := \emptyset \), and \( r \) counters \( c_1 := c_2 := \ldots := c_r := 0 \) (one for each type) with zero. Then, for
each vertex \( v \), compute the types of \( v \) in \( O(|\tau(v)|) \) time and let \( I \subseteq \{1, \ldots, r\} \) be the set of types of \( v \). If \( v \) is unsatisfied or if \( c_i \leq \alpha \) for some \( i \in I \), then add \( v \) to \( C \). If \( v \) is satisfied, then increase \( c_i \) by one for each \( i \in I \). Now that we computed the vertices in \( C \) in linear time, it remains to compute their correct degree lists. To this end, for each vertex \( v \in C \), compute \( \gamma = \deg_G(v) - \deg_G[C] \) (doable in \( O(\deg(v)) \) time) and set \( \tau_{V\setminus C}(v) := \{ d \geq 0 \mid d + \gamma \in \tau(v) \} \) in \( O(|\tau(v)|) \) time. Overall, we safely removed all vertices in \( V \setminus C \) in linear time, that is, in time \( O(m + |\tau|) \).

As each \( \alpha \)-type set contains at most \( \alpha \) satisfied vertices of each vertex type, it follows that after one application of Reduction Rule 1 the graph contains at most \( |C| = |U| + r\alpha \) vertices. The number of unsatisfied vertices in an \( \alpha \)-type set can always be bounded by \( |U| \leq 2k \) since we can increase the degrees of at most \( 2k \) vertices by adding \( k \) edges. If there are more unsatisfied vertices, then we return a trivial no-instance. Thus, we end up with \( |C| \leq 2k + rk(\Delta_G + 2) \). To obtain a polynomial-size problem kernel with respect to the combined parameter \((k, r)\), we need to bound \( \Delta_G \). However, this can easily be achieved: Since we only allow edge additions, for each vertex \( v \in V \), we have \( \deg(v) \leq \max \tau(v) \leq r \).

Formalized as a data reduction rule, this reads as follows:

**Rule 2.** Let \((G = (V, E), k, r, \tau)\) be an instance of \( \text{DCE}(e^+) \). If \( G \) contains more than \( 2k \) unsatisfied vertices or if there exists a vertex \( v \in V \) with \( \deg(v) > \max \tau(v) \), then return a trivial no-instance.

Having applied Reduction Rules 1 and 2 once, it holds that \( \Delta_G \leq r \) and thus the graph contains at most \( 2k + rk(r + 1) \) vertices. Lemma 1 ensures that we can apply Reduction Rule 1 in linear time. Note that linear time means \( O(m + |\tau|) \) time, where \( |\tau| \geq n \) denotes the encoding size of \( \tau \). Clearly, Reduction Rule 2 can be applied in linear time too. This leads to the following.

**Theorem 5.** \( \text{DCE}(e^+) \) admits a problem kernel containing \( O(kr^2) \) vertices computable in \( O(m + |\tau|) \) time.

### 2.2 A Polynomial Kernel for \( \text{DCE}(e^+) \) with respect to \( r \)

In this subsection, we show how to extend the polynomial-size problem kernel provided in Theorem 5 to a polynomial-size problem kernel for the single parameter \( r \). To this end, among other things, we adapt some ideas of Hartung et al. [22] to show how to bound \( k \) in a polynomial of \( r \). The general strategy, inspired by a heuristic of Liu and Terzi [27], will be as follows: First, remove the graph structure and solve the problem on the degree sequence of the input graph by using dynamic programming. The solution to this number problem will indicate the demand for each vertex, that is, the number of added edges incident to that vertex. Then, using a result of Katerinis and Tsikopoulos [24], we prove that either \( k \leq r(r + 1)^2 \) or we can find a set of edges satisfying the specified demands in polynomial time.

We start by formally defining the corresponding number problem and showing its polynomial-time solvability.
**Number Constraint Editing (NCE)**

**Input:** A function \( \phi : \{1, \ldots, n\} \rightarrow 2^{\{0, \ldots, r\}} \) and positive integers \( d_1, \ldots, d_n, k, r \).

**Question:** Are there \( n \) positive integers \( d'_1, \ldots, d'_n \) such that \( \sum_{i=1}^{n} (d'_i - d_i) = k \) and for all \( i \in \{1, \ldots, n\} \) it holds that \( d'_i \geq d_i \) and \( d'_i \in \phi(i) \)?

**Lemma 2.** NCE is solvable in \( O(n \cdot k \cdot r) \) time.

**Proof.** We provide a simple dynamic programming algorithm for NCE. To this end, we define a two-dimensional table \( T \) as follows: For \( i \in \{1, \ldots, n\} \) and \( j \in \{1, \ldots, k\} \), the entry \( T[i, j] \) is true if and only if the instance \((d_1, \ldots, d_i, j, r, \phi)\) is a yes-instance. Hence, \( T[n, k] \) stores the answer to the instance \((d_1, \ldots, d_n, k, r, \phi)\).

In order to compute \( T \), we use the following recurrence:

\[
T[i, j] = \text{true} \iff \exists x \in \phi(i) : x \geq d_i \land T[i-1, j-(x-d_i)] = \text{true}. \tag{1}
\]

The recursion terminates with

\[
T[1, j] := \begin{cases} 
\text{true}, & \text{if } d_1 + j \in \phi(1), \\
\text{false}, & \text{else.}
\end{cases}
\]

The correctness follows from the fact that at position \( i \) all possibilities for \( d'_i \) are considered. Also the running time is not hard to see: There are \( n \cdot k \) entries and the computation of one entry requires to check at most \( r \) possibilities for the value \( x \) in (1). As each check is doable in \( O(1) \) time, the overall running time sums up to \( O(n \cdot k \cdot r) \).

Lemma 2 can be proved with a dynamic program that specifies the demand for each vertex, that is, the number of added edges incident to each vertex. Given these demands, the remaining problem is to decide whether there exists a set of edges that satisfy these demands and are not contained in the input graph \( G \). This problem is closely related to the polynomial-time solvable \( f \)-Factor problem \([29]\), a special case of DCE\((e^-)\) where \( |\tau(v)| = 1 \) for all \( v \in V \); it is formally defined as follows:

**\( f \)-Factor**

**Input:** A graph \( G = (V, E) \) and a function \( f : V \rightarrow \mathbb{N}_0 \).

**Question:** Is there an \( f \)-factor, that is, a subgraph \( G' = (V, E') \) of \( G \) such that \( \deg_G(v) = f(v) \) for all \( v \in V \)?

Observe that our problem of satisfying the demands of the vertices in \( G \) is essentially the question whether there is an \( f \)-factor in the complement graph \( \overline{G} \) where the function \( f \) stores the demand of each vertex. Using a result of Katerinis and Tsikopoulos \([24]\), we can show the following lemma about the existence of an \( f \)-factor:

**Lemma 3.** Let \( G = (V, E) \) be a graph with \( n \) vertices, \( \delta_G \geq n-r-1, r \geq 1 \), and let \( f : V \rightarrow \{1, \ldots, r\} \) be a function such that \( \sum_{v \in V} f(v) \) is even. If \( n \geq (r+1)^2 \), then \( G \) has an \( f \)-factor.

**Proof.** We use the following result to prove the statement.
Lemma 4 (Katerinis and Tsikopoulos [24]). Let $G = (V, E)$ be a graph with minimum vertex degree $\delta_G$ and let $a \leq b$ be two positive integers. Suppose further that

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

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

Set $a = 1$ and $b = r$. Then, $\delta_G \geq n - r - 1 \geq \frac{r}{r+1} n$ holds if $n \geq (r+1)^2$, which is true by assumption. Also, $\frac{r+1}{r}(b+a-3) = (r+1)(r-2) = r^2 - r - 2 < (r+1)^2 \leq n$ holds, and thus all conditions of Lemma 4 are fulfilled. □

We now have all ingredients to show that we can upper-bound $k$ by $r(r+1)^2$ or solve the given instance of DCE(e+) in polynomial time. The main technical statement towards this is the following.

Lemma 5. Let $I := (G = (V, E), k, r, \tau)$ be an instance of DCE(e+) with $k \geq r(r+1)^2$ and $V = \{v_1, \ldots, v_n\}$. If there exists a $k' \in \{r(r+1)^2, \ldots, k\}$ such that $(\text{deg}(v_1), \ldots, \text{deg}(v_n), 2k', r, \phi)$ with $\phi(i) := \tau(v_i)$ is a yes-instance of NCE, then $I$ is a yes-instance of DCE(e+).

Proof. Assume that $(\text{deg}(v_1), \ldots, \text{deg}(v_n), 2k', r, \phi)$ is a yes-instance of NCE. Let $d'_1, \ldots, d'_n$ be integers such that $d'_i \in \tau(v_i)$, $\sum_{i=1}^n d'_i - \text{deg}(v_i) = 2k'$, and $d'_i \geq d_i$. Hence, we know that the degree constraints can numerically be satisfied, giving rise to a new target degree $d'_i$ for each vertex $v_i$. Let $A := \{v_i \in V \mid d'_i > \text{deg}(v_i)\}$ denote the set of affected vertices containing all vertices which require addition of some edges. It remains to show that the degree sequence of the affected vertices can in fact be realized by adding $k'$ edges to $G[A]$. To this end, it is sufficient to prove the existence of an $f$-factor in the complement graph $\overline{G[A]}$ with $f(v_i) := d'_i - \text{deg}(v_i) \in \{1, \ldots, r\}$ for all $v_i \in A$ since such an $f$-factor contains exactly the $k'$ edges we want to add to $G$. Thus, it remains to check that all conditions of Lemma 3 are indeed satisfied to conclude the existence of the sought $f$-factor. First, note that $\delta_{\overline{G[A]}} \geq |A| - r - 1$ since $\Delta_{\overline{G[A]}} \leq r$. Moreover, $\sum_{v_i \in A} (d'_i - \text{deg}(v_i)) = 2k' \leq |A|r$, and thus $|A| \geq 2k'/r \geq 2(r+1)^2$. Finally, $\sum_{v_i \in A} f(v_i) = 2k'$ is even and thus Lemma 3 applies. □

As NCE is polynomial-time solvable, Lemma 5 states a win-win situation: either the solution is bounded in size or can be found in polynomial time. From this and Theorem 5, we obtain the polynomial-size problem kernel.

Theorem 6. DCE(e+) admits a problem kernel containing $O(r^5)$ vertices computable in $O(k^2 \cdot r \cdot n + m + |\tau|)$ time.

Proof. Let $I := (G, k, r, \tau)$ be an instance of DCE(e+). We distinguish two cases concerning the size of $k$.

Case 1 $k > r(r+1)^2$: We solve for all $k' \in \{r(r+1)^2, \ldots, k\}$ the corresponding NCE formulation. If for one $k'$, we encounter a yes-instance of the NCE formulation, then, due to Lemma 5, we return a trivial yes-instance of constant size. By Lemma 2, this can be done in polynomial time. Otherwise, as each solution for DCE(e+) can be transferred to a solution of NCE, it follows that there is no solution for $I$ of size $k'$ for any $k' \in \{r(r+1)^2, \ldots, k\}$. Thus, $I$ is a yes-instance if and only if $(G, r(r+1)^2, r, \tau)$ is a yes-instance. Hence, set $k := r(r+1)^2$ and proceed as in the Case 2.
Case 2 $k \leq r(r+1)^2$: We simply run the kernelization algorithm from Theorem 5 on $I$ to obtain an $O(r^5)$-vertex problem kernel.

Concerning the running time, observe that we have to solve at most $k$ times an instance of NCE. By Lemma 2, we can determine in $O(k \cdot r \cdot n)$ time for each of these at most $k$ instances whether it is a yes- or no-instance. If one instance is a yes-instance, due to Lemma 5, then the kernelization algorithm can return a trivial yes-instance in constant time. Otherwise, we apply Theorem 5 in $O(m + |\tau|)$ time. Overall, this gives a running time of $O(k^2 \cdot r \cdot n + m + |\tau|)$.  

3 A General Approach for Degree Sequence Completion

In the previous section, we dealt with the problem DCE($e^+$), where one only has to locally satisfy the degree of each vertex. In this section, we show how the presented ideas for DCE($e^+$) can also be used to solve more globally defined problems where the degree sequence of the solution graph $G'$ has to fulfill a given property. For example, consider the problem of adding a minimum number of edges to obtain a regular graph, that is, a graph where all vertices have the same degree. In this case the degree of a vertex in the solution is a priori not known but depends on the degrees of the other vertices.

The degree sequence of a graph $G$ with $n$ vertices is an $n$-tuple containing the vertex degrees. Then, for some tuple property $\Pi$, we consider the following problem:

\[ \Pi\text{-Degree Sequence Completion (}\Pi\text{-DSC)} \]

**Input:** A graph $G = (V, E)$, an integer $k \in \mathbb{N}$.

**Question:** Is there a set of edges $E' \subseteq \binom{V}{2} \setminus E$ with $|E'| \leq k$ such that the degree sequence of $G + E'$ fulfills $\Pi$?

Note that $\Pi$-DSC is not a generalization of DCE($e^+$) since in DCE($e^+$) one can require for two vertices $u$ and $v$ of the same degree that $u$ gets two more incident edges and $v$ not. This cannot be expressed in $\Pi$-DSC. We remark that the results stated in this section can be extended to hold for a generalized version of $\Pi$-DSC where a “degree list function” $\tau$ is given as additional input and the vertices in the solution graph $G'$ also have to satisfy $\tau$, thus generalizing DCE($e^+$). For simplicity, however, we stick to the easier problem definition as stated above and defer the details to a full version.

3.1 Fixed-Parameter Tractability of $\Pi$-DSC

In this subsection, we first generalize the ideas behind Theorem 5 to show fixed-parameter tractability of $\Pi$-DSC with respect to the combined parameter $(k, \Delta_G)$. Then, we present an adjusted version of Lemma 5 and apply it to show fixed-parameter tractability for $\Pi$-DSC with respect to the parameter $\Delta_{G'}$. Clearly, a prerequisite for both these results is that the following problem has to be fixed-parameter tractable with respect to the parameter $\Delta_T := \max\{d_1, \ldots, d_n\}$.

\[ \Pi\text{-Decision} \]

**Input:** An integer tuple $T = (d_1, \ldots, d_n)$.

**Question:** Does $T$ fulfill $\Pi$?
Theorem 7. Let $\Pi$ be some tuple property. If II-Decision is fixed-parameter tractable with respect to $\Delta_T$, then II-DSC is fixed-parameter tractable with respect to $(k, \Delta_G)$.

Bounding the Solution Size $k$ in $\Delta_G$. We now show how to extend the ideas of Section 2.2 to the context of II-DSC in order to bound the solution size $k$ by a polynomial in $\Delta_G$. The general procedure still is the one inspired by Liu and Terzi [27]: Solve the number problem corresponding to II-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:

**II-Number Sequence Completion (II-NSC)**

**Input:** Positive integers \(d_1,\ldots,d_n,k,\Delta\).

**Question:** Are there \(n\) nonnegative integers \(x_1,\ldots,x_n\) with \(\sum_{i=1}^{n} x_i = k\) such that \((d_1 + x_1,\ldots,d_n + x_n)\) fulfills \(\Pi\) and \(d_i + x_i \leq \Delta\)?

With these problem definitions, we can now generalize Lemma 5.

**Lemma 7.** Let \(I := (G,k)\) be an instance of II-DSC with \(V = \{v_1,\ldots,v_n\}\) and \(k \geq \Delta_G(\Delta_{G'} + 1)^2\). If there exists a \(k' \in \{\Delta_G(\Delta_{G'} + 1)^2,\ldots,k\}\) such that the corresponding II-NSC instance \(I' := (\deg(v_1),\ldots,\deg(v_n),2k',\Delta_{G'})\) is a yes-instance, then \(I\) is a yes-instance.

**Proof.** Let \(I' := (\deg(v_1),\ldots,\deg(v_n),2k',\Delta_{G'})\) with \(k' \in \{\Delta_G(\Delta_{G'} + 1)^2,\ldots,k\}\) be a yes-instance of II-NSC and let \(x_1,\ldots,x_n\) denote a solution for \(I'\). Defining the function \(f: V \rightarrow \mathbb{N}\) as \(f(v_i) := x_i\), we now prove that \(G\) contains an \(f\)-factor which forms a solution \(E'\) for \(I\). Denote by \(A\) the set of affected vertices, formally, \(A := \{v_i \in V \mid 0 < x_i\}\). Observe that \(|A| \geq 2k'/\Delta_{G'} \geq 2(\Delta_{G'} + 1)^2\) as \(k' \geq \Delta_G(\Delta_{G'} + 1)^2\). Furthermore, as the maximum degree in \(G\) is \(\Delta_G \leq \Delta_{G'}\), it follows that \(G[A]\) has minimum degree at least \(|A| - \Delta_{G'} - 1\). Finally, observe that \(f(v_i) \in \{1,\ldots,\Delta_{G'}\}\) for each \(v_i \in A\) and that \(\sum_{v_i \in A} f(v_i) = 2k'\) is even. Hence, by Lemma 3, \(G[A]\) contains an \(f\)-factor. Thus, \(G\) also contains an \(f\)-factor \(G' = (V,E')\) and since \((\deg(v_1) + x_1,\ldots,\deg(v_n) + x_n)\) fulfills \(\Pi\), it follows that \(E'\) is a solution for \(I\), implying that \(I\) is a yes-instance. \(\square\)

Let function \(g(|I|)\) denote the running time for solving the II-NSC instance \(I\). Clearly, if there is a solution for an instance of II-DSC, then there also exists a solution for the corresponding II-NSC instance. It follows that we can decide whether there is a large solution for II-DSC (with at least \(\Delta_G(\Delta_{G'} + 1)^2\) edges) in \(k \cdot g(n \log(n))\) time. Hence, we arrive at the following win-win situation:

**Corollary 1.** Let \(I := (G,k)\) be an instance of II-DSC. Then, either one can decide in \(k \cdot g(n \log(n))\) time that \(I\) is a yes-instance, or \(I\) is a yes-instance if and only if \((G,\min\{k,\Delta_{G'}(\Delta_{G'} + 1)^2\})\) is a yes-instance.

Using Corollary 1, we can transfer fixed-parameter tractability of II-NSC with respect to \(\Delta\) to fixed-parameter tractability of II-DSC with respect to \(\Delta_{G'}\). Notice that \(\Delta_{G'} \leq k + \Delta_G\), that is, \(\Delta_{G'}\) is a smaller and thus “stronger” parameter [26]. Also, showing II-NSC to be fixed-parameter tractable with respect to \(\Delta\) is a significantly easier task than proving fixed-parameter tractability for II-DSC with respect to \(\Delta_{G'}\) directly since the graph structure can be completely ignored.

**Theorem 8.** If II-NSC is fixed-parameter tractable with respect to \(\Delta\), then II-DSC is fixed-parameter tractable with respect to \(\Delta_{G'}\).

**Proof.** Let \(I := (G,k)\) be a II-DSC instance. First, note that \(\Delta_G \leq \Delta_{G'}\) always holds since we are only adding edges to \(G\). Thus, if \(k \leq \Delta_{G'}(\Delta_{G'} + 1)^2\), then the fixed-parameter tractability with respect to \((k,\Delta_G)\) from Theorem 7 yields fixed-parameter tractability with respect to \(\Delta_{G'}\). Otherwise, we use Corollary 1 to check whether there exists a large solution of size at least \(\Delta_{G'}(\Delta_{G'} + 1)^2\).
Hence, by assumption, in time $f(\Delta G') \cdot n^{O(1)}$ for some computable function $f$, we either find that $I$ is a yes-instance or we can assume that $k \leq \Delta G'(\Delta G' + 1)^2$, which overall yields fixed-parameter tractability with respect to $\Delta G'$.

If II-NSC can be solved in polynomial time, then Corollary 1 shows that we can assume that $k \leq \Delta G'(\Delta G' + 1)^2$. Thus, as in the DCE($n^+$) setting (Theorem 6), polynomial kernels with respect to $(k, \Delta G)$ transfer to the parameter $\Delta G'$, leading to the following.

**Theorem 9.** If II-NSC is polynomial-time solvable and II-DSC admits a polynomial kernel with respect to $(k, \Delta G)$, then II-DSC also admits a polynomial kernel with respect to $\Delta G'$.

### 3.2 Applications

As our general approach is inspired by ideas of Hartung et al. [22], it is not surprising that it can be applied to “their” Degree Anonymity problem:

**DEGREE ANONYMITY**

**Input:** An undirected graph $G = (V,E)$ and two positive integers $k$ and $s$.

**Question:** Is there an edge set $E'$ over $V$ of size at most $s$ such that $G' := G + E'$ is $k$-anonymous, that is, for each vertex $v \in V$, there are at least $k - 1$ other vertices in $G'$ having the same degree?

The property II of being $k$-anonymous clearly can be decided in polynomial time for a given degree sequence and thus, by Theorem 7, we immediately get fixed-parameter tractability with respect to $(s, \Delta G)$. Theorem 9 then basically yields the kernel results obtained by Hartung et al. [22]. There are more general versions of Degree Anonymity as proposed by Chester et al. [10]. For example, just a given subset of the vertices has to be anonymized or the vertices have labels. As in each of these generalizations one can decide in polynomial time whether a given graph satisfies the particular anonymity requirement, Theorem 7 applies also in these scenarios. However, checking in which of these more general settings the conditions of Theorem 8 or Theorem 9 are fulfilled is future work.

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 $D$, then being $k$-anonymous translates into $p_i(D_{G'}) \geq k$ for all degrees $i$ occurring in the degree sequence $D_{G'}$ of the modified graph $G'$. Now, it is natural to consider not only a lower bound $k \leq p_i(D)$, but also an upper bound $p_i(D) \leq u$ or maybe even a set of allowed frequencies $p_i(D) \in F_i \subseteq \mathbb{N}$. 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 $\ell$ times for some $\ell \in \mathbb{N}$ [9]. To obtain some sort of “robustness” it might be useful to ask for an $h$-index of $\ell$, that is, in the solution graph are at least $\ell$ vertices with degree at least $\ell$ [13].

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;
one of the first is the well-known result by Erdős and Gallai [14] showing which degree sequences are in fact graphic, that is, realizable by a graph.

**Theorem 10** (Erdős and Gallai [14]). Let $D := (d_1, d_2, \ldots, d_n)$ be a degree sequence, where $d_1 \geq d_2 \geq \ldots \geq d_n$. Then, $D$ is a realizable (or graphical) if and only if $\sum_{i=1}^{n} d_i$ is even and for all $r \in \{1, \ldots, n-1\}$, it holds that

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

Based on this characterization researchers characterized for example pseudo-split, split and threshold graphs completely by their degree sequences.

**Theorem 11.** Let $G$ be a graph with non-increasing degree sequence $(d_1, d_2, \ldots, d_n)$ and $\omega = \max\{i \mid d_i \geq i - 1\}$.

1. (Hammer et al. [21]) $G$ is a threshold graph if and only if for all $r \in \{1, \ldots, \omega\}$

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

2. (Hammer and Simeone [20]) $G$ is a split graph if and only if

$$\sum_{i=1}^{\omega} d_i = \omega(\omega - 1) + \sum_{i=\omega+1}^{n} d_i.$$ 

3. (Maffray and Preissmann [30]) $G$ is a pseudo-split graph if and only if $G$ is a split graph or

$$\sum_{i=1}^{q} d_i = q(q + 4) + \sum_{i=q+6}^{n} d_i$$

and $d_{q+1} = d_{q+2} = d_{q+3} = d_{q+4} = d_{q+5} = q + 2$ where $q = \max\{i \mid d_i \geq i + 4\}$.

The NP-hard Split Graph Completion [33] problem, for example, is known to be fixed-parameter tractable with respect to $k$ [8]. Note, however, that for the graph classes from Theorem 11 polynomial kernels with respect to the parameter $\Delta G$ trivially exist because here we always have $\sqrt{n} \leq \Delta G$. Nevertheless, we mention this (trivial) implication as further evidence that our framework applies to many problems.

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 [6]. Given a degree sequence $D = (d_1, \ldots, d_n)$, one can decide in linear time whether $D$ defines a unigraph [5, 25]. Again, by Theorem 8, we conclude fixed-parameter tractability for the unigraph completion problem with respect to the parameter $\Delta G$. 

16
4 Conclusion

We proposed a method for deriving efficient preprocessing algorithms for degree sequence completion problems. DCE($e^+$) served as our main illustrating example. Roughly speaking, the core of the approach (as basically already used in previous work [22, 27]) consists of extracting the degree sequence from the input graph, efficiently solving a simpler number editing problem, and translating the obtained solution back into a solution for the graph problem using $f$-factors. While previous work [22, 27] was specifically tailored towards an application for graph anonymization, we generalized the approach by filtering out problem-specific parts and “universal” parts. Thus, whenever one can solve these problem-specific parts efficiently, we can automatically obtain efficient preprocessing and fixed-parameter tractability results.

Our approach seems promising for future empirical investigations concerning its practical usefulness, a very recent experimental work has been performed for Degree Anonymity [23]. Another line of future research could be to study polynomial-time approximation algorithms for the considered degree sequence completion problems. Perhaps parts of our preprocessing approach might find use here as well. A more specific open question concerning our work would be how to deal with additional connectivity requirements for the generated graphs.

Acknowledgments. Vincent Froese was supported by the DFG, project DAMM (NI 369/13).

References

[1] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In Proc. 4th IWPEC, volume 5917 of LNCS, pages 17–37. Springer, 2009.

[2] H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) kernelization. In Proc. 50th FOCS, pages 629–638. IEEE, 2009.

[3] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Cross-composition: A new technique for kernelization lower bounds. In Proc. 28th STACS, volume 9 of LIPIcs, pages 165–176. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[4] H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. Theor. Comput. Sci., 412(35):4570–4578, 2011.

[5] A. Borri, T. Calamoneri, and R. Petreschi. Recognition of unigraphs through superposition of graphs. J. Graph Algorithms Appl., 15(3):323–343, 2011.

[6] A. Brandstädt, V. B. Le, and J. P. Spinrad. Graph Classes: a Survey, volume 3 of SIAM Monographs on Discrete Mathematics and Applications. SIAM, 1999.

[7] R. Bredereck, S. Hartung, A. Nichterlein, and G. J. Woeginger. The complexity of finding a large subgraph under anonymity constraints. In Proc. 24th ISAAC, volume 8283 of LNCS, pages 152–162. Springer, 2013.
[8] L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Inf. Process. Lett.*, 58(4):171–176, 1996.

[9] G. Chartrand, L. Lesniak, C. M. Mynhardt, and O. R. Oellermann. Degree uniform graphs. *Ann. N. Y. Acad. Sci.*, 555(1):122–132, 1989.

[10] S. Chester, B. Kapron, G. Srivastava, and S. Venkatesh. Complexity of social network anonymization. *Social Netw. Analys. Mining*, 3(2):151–166, 2013.

[11] G. Cornuéjols. General factors of graphs. *J. Combin. Theory Ser. B*, 45(2):185–198, 1988.

[12] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013.

[13] D. Eppstein and E. S. Spiro. The h-index of a graph and its application to dynamic subgraph statistics. *J. Graph Algorithms Appl.*, 16(2):543–567, 2012.

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

[15] M. R. Fellows, J. Guo, H. Moser, and R. Niedermeier. A generalization of Nemhauser and Trotter’s local optimization theorem. *J. Comput. System Sci.*, 77(6):1141–1158, 2011.

[16] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[17] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

[18] P. A. Golovach. Editing to a connected graph of given degrees. *CoRR*, abs/1308.1802, 2013.

[19] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.

[20] P. L. Hammer and B. Simeone. The splittance of a graph. *Combinatorica*, 1(3):275–284, 1981.

[21] P. L. Hammer, T. Ibaraki, and B. Simeone. Degree sequences of threshold graphs. *Congres. Numer.*, 21:329–355, 1975.

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

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

[24] P. Katerinis and N. Tsikopoulos. Minimum degree and $f$-factors in graphs. *New Zealand J. Math.*, 29(1):33–40, 2000.
[25] D. J. Kleitman and S. Y. Li. A note on unigraphic sequences. *Stud. Appl. Math.*, 4:283–287, 1975.

[26] C. Komusiewicz and R. Niedermeier. New races in parameterized algorithms. In *Proc. 37th MFCS*, volume 7464 of *LNCS*, pages 19–30. Springer, 2012.

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

[28] D. Lokshtanov, N. Misra, and S. Saurabh. Kernelization - preprocessing with a guarantee. In *The Multivariate Algorithmic Revolution and Beyond*, pages 129–161, 2012.

[29] L. Lovász and M. D. Plummer. *Matching Theory*, volume 29 of *Annals of Discrete Mathematics*. North-Holland, 1986.

[30] F. Maffray and M. Preissmann. Linear recognition of pseudo-split graphs. *Discrete Appl. Math.*, 52(3):307–312, 1994.

[31] L. Mathieson and S. Szeider. Editing graphs to satisfy degree constraints: A parameterized approach. *J. Comput. System Sci.*, 78(1):179–191, 2012.

[32] H. Moser and D. M. Thilikos. Parameterized complexity of finding regular induced subgraphs. *J. Discrete Algorithms*, 7(2):181–190, 2009.

[33] A. Natanzon, R. Shamir, and R. Sharan. Complexity classification of some edge modification problems. *Discrete Appl. Math.*, 113:109–128, 2001.

[34] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.