An Algorithmic Framework for Fixed-Cardinality Optimization in Sparse Graphs Applied to Dense Subgraph Problems[☆]

Christian Komusiewicz[a], Manuel Sorge[a,1,*]

[a] *Institut für Softwaretechnik und Theoretische Informatik, TU Berlin*

Abstract

We investigate the computational complexity of the Densest k-Subgraph problem, where the input is an undirected graph $G = (V, E)$ and one wants to find a subgraph on exactly k vertices with the maximum number of edges. We extend previous work on Densest k-Subgraph by studying its parameterized complexity for parameters describing the sparseness of the input graph and for parameters related to the solution size k.

On the positive side, we show that, when fixing some constant minimum density μ of the sought subgraph, Densest k-Subgraph becomes fixed-parameter tractable with respect to either of the parameters maximum degree of G and h-index of G. Furthermore, we obtain a fixed-parameter algorithm for Densest k-Subgraph with respect to the combined parameter "degeneracy of G and $|V| - k$".

On the negative side, we find that Densest k-Subgraph is W[1]-hard with respect to the combined parameter "solution size k and degeneracy of G". We furthermore strengthen a previous hardness result for Densest k-Subgraph [Cai, Comput. J., 2008] by showing that for every fixed μ, $0 < \mu < 1$, the problem of deciding whether G contains a subgraph of density at least μ is W[1]-hard with respect to the parameter $|V| - k$.

Our positive results are obtained by an algorithmic framework that can be applied to a wide range of Fixed-Cardinality Optimization problems.

Keywords: clique relaxations, NP-hard problems, fixed-parameter algorithms, subgraph enumeration

[☆] A preliminary version of this work is contained in the *Proceedings of the 7th International Symposium on Parameterized and Exact Computation (IPEC '12)* under the title *Finding Dense Subgraphs of Sparse Graphs* [36]. The current article includes improved running times as well as generalized algorithmic results. This work is to be published in *Discrete Applied Mathematics*, Elsevier, 2015.

[*] Corresponding author. Phone +49 30 314 24166, Fax +49 30 314 23516.

Email addresses: `christian.komusiewicz@tu-berlin.de` (Christian Komusiewicz), `manuel.sorge@tu-berlin.de` (Manuel Sorge)

[1] Supported by DFG projects PABI (NI 369/7-2) and DAPA (NI 369/12).

1. Introduction

Identifying dense regions of graphs is a fundamental computational problem with many important applications, for example in computational biology [43] and social network analysis [5]. There are many different definitions of what a dense subgraph is [23, 38] and for almost all of these formulations, the corresponding computational problems are NP-hard.

In this work, we study the problem of finding subgraphs with a fixed number k of vertices and a maximum number of edges. This problem is known as DENSEST k-SUBGRAPH. For fixed k, maximizing the number of edges is equivalent to maximizing the *density* of a graph $G = (V, E)$ which is defined as $2|E|/(|V|(|V| - 1))$. Using this notion, the NP-hard DENSEST k-SUBGRAPH problem [23, 35] can be defined as follows.

DENSEST k-SUBGRAPH
Input: A graph $G = (V, E)$, and a nonnegative integer k.
Task: Find a vertex set $S \subseteq V$ of size exactly k such that $G[S]$ has maximum density.

DENSEST k-SUBGRAPH is at least as hard as the well-studied CLIQUE problem which asks for finding a complete graph of order exactly k. In this work, our aim is to provide a better picture of when DENSEST k-SUBGRAPH becomes computationally hard or tractable. To this end, we consider how two types of parameters influence the complexity of DENSEST k-SUBGRAPH.

The first type comprises the classic parameter solution size k and its "dual parameterization" $|V| - k$. Parameters of the second type measure the sparseness of the input graph G: maximum degree Δ, h-index,[2] and degeneracy d. Informally, bounded maximum degree means that *all* vertices have few neighbors, bounded h-index means that *most* vertices have few neighbors, and bounded degeneracy means that in every subgraph *there is always* a vertex with few neighbors. By definition, $\Delta \geq h$-index $\geq d$. The study of these three parameters is motivated by two facts: First, many real-world networks such as biological and social networks are relatively sparse since they contain many vertices of low degree and only few vertices of high degree (the network "hubs"). Second, the otherwise notoriously hard CLIQUE problem is much easier on sparse graphs. For example, all maximal cliques can be enumerated in $O(3^{d/3} \cdot d \cdot |V|)$ time on graphs with degeneracy d [21].

We study the complexity of DENSEST k-SUBGRAPH mostly by considering the following problem which can be seen as a decision variant of DENSEST k-SUBGRAPH. Here, one asks whether there is a k-vertex subgraph with density *at least* μ, where $0 \leq \mu \leq 1$, is a fixed rational number. We call such subgraphs μ-cliques, that is, a graph $G = (V, E)$ is a *μ-clique* if the density of G is at least μ.

μ-CLIQUE
Input: A graph $G = (V, E)$, and a nonnegative integer k.
Question: Is there a vertex set $S \subseteq V$ of size at least k such that $G[S]$ is a μ-clique?

[2] The structural graph parameter h-index was introduced by Eppstein and Spiro [20] in the context of triangle counting in dynamic graphs. For a definition, see Section 2.

Throughout this work, we use n to denote the number of vertices in G. As mentioned, we assume furthermore that μ is a fixed rational number. In other words, μ is independent of k and n. Our motivation for this assumption is that the dense subgraphs that one wants to find in many applications should be almost complete graphs.

Densest k-Subgraph is an example of an optimization problem with cardinality constraints. Our positive algorithmic results for Densest k-Subgraph and μ-Clique are applicable to a range of such problems. A generic problem formulation (with graph inputs) can be given as follows.

> Fixed-Cardinality Optimization
> Input: An undirected graph $G = (V, E)$, an objective function $\phi\colon 2^V \to \mathbb{Q}^+$, and an integer k.
> Task: Find the maximum value $\phi(S)$ for any set $S \subseteq V$ such that $|S| = k$.

In this article we assume that ϕ is given as an algorithm that receives G and k and $S \subseteq V$ as input, runs in $T(k, G)$ time and computes $\phi(S)$. Note that the cardinality constraint could also apply to the number of edges of the solution and that, in general, fixed-cardinality optimization is not restricted to graph inputs.

Related Work. For an overview of Fixed-Cardinality Optimization problems, refer to [12]; the parameterized complexity of some special cases is studied by Cai [13]. The random separation method [14] yields fixed-parameter algorithms for a wide range of special cases of Fixed-Cardinality Optimization and the combined parameter (Δ, k) where Δ is the maximum degree of G. For the special case of Densest k-Subgraph, the randomized algorithm takes $\mathrm{O}(2^{(\Delta+1)\cdot k} \cdot (\Delta + k) \cdot n)$ time to achieve a constant error probability. Derandomization of the algorithm adds a factor of $(\Delta k + k)^{\mathrm{O}(\log(\Delta k))} \log n$ to the running time.

The notion of μ-clique, also called μ-dense graph or μ-quasi clique [41] in the literature, is an example of a so-called *clique relaxation*. Surveys of different types of clique relaxations and the computational problems associated with finding such subgraphs were given by Balasundaram and Pajouh [3] and Kosub [38].

Specific results concerning μ-Clique and Densest k-Subgraph are as follows. For $\mu = 1$ the question in μ-Clique becomes to find a complete subgraph of order k, which is W[1]-hard with respect to the parameter k and fixed-parameter tractable with respect to the dual parameter $n - k$ [18, 19]. The μ-Clique problem remains NP-hard for every rational number μ, $0 < \mu < 1$ [41]. Densest k-Subgraph is NP-hard and W[1]-hard with respect to k, as it is a generalization of Clique. Moreover, Densest k-Subgraph is W[1]-hard with respect to the parameter $n - k$ [13]. It is, however, fixed-parameter tractable with respect to the combined parameter "maximum degree Δ and k" [14]. Holzapfel et al. [30] showed that Densest k-Subgraph remains NP-hard, even when looking only for subgraphs with average degree at least $2 + \Omega(1/k^{1-\epsilon})$ for $0 < \epsilon < 2$. Finding k-vertex subgraphs of average degree at least $2 + \mathrm{O}(1/k)$, however, can be done in polynomial time [30]. Furthermore, Densest k-Subgraph is NP-hard even in graphs with maximum degree three and degeneracy two [22]. The "densest subgraph" in the corresponding reduction, however, has very low, non-constant density. Motivated by these algorithmic hardness results, approximation algorithms for Densest k-Subgraph have also received a lot of attention [23, 34, 35]. A trivial exponential-time algorithm solves Densest k-Subgraph in $2^n \cdot \mathrm{poly}(n)$ time by checking all vertex-subsets. Chang

Table 1: Summary of our results and previous results for μ-Clique and Densest k-Subgraph. Note that hardness transfers from μ-Clique to Densest k-Subgraph and tractability transfers in the reverse direction. For fixed-parameter tractability (FPT) results, we write a rough estimate of the exponential running time factor. Herein, k denotes the order of the sought μ-clique, $\ell := n - k$, and d denotes the degeneracy of the input graph.

parameter	μ-Clique	Densest k-Subgraph
max. degree Δ	FPT: $\Delta^{O(\Delta)}$ (Theorem 5), no poly. kernel (Theorem 11)	NP-hard for $\Delta = 3$ [22]
h-index h	FPT: $h^{O(h)}$ (Theorem 6), no poly. kernel (Theorem 11)	NP-hard for $h = 3$ [22]
degeneracy d	in XP (Lemma 4(iii))	NP-hard for $d = 2$ [22]
(k, d)	W[1]-hard (Theorem 10)	W[1]-hard (Theorem 10)
ℓ	W[1]-hard (Theorem 8)	W[1]-hard [13]
(ℓ, d)	FPT: $(\ell + d)^{O(\ell)}$ (Theorem 7)	FPT: $(\ell + d)^{O(\ell)}$ (Theorem 7)

et al. [15] show that the running time can be improved to $1.7315^n \cdot \text{poly}(n)$ and Bourgeois et al. [10] present improved exponential-time algorithms for some special cases of Densest k-Subgraph.

A related problem is Minimum Subgraph of Minimum Degree, where the task is to find a subgraph of order at most k such that each vertex has a given minimum degree. Minimum Subgraph of Minimum Degree is W[1]-hard with respect to the parameter k but becomes fixed-parameter tractable on graphs of bounded local treewidth and graphs with excluded minors [2]. A further related problem, referred to as Densest Subgraph, is to find a subgraph that has maximum average degree (without constraint on the order). Densest Subgraph is polynomial-time solvable using network flow techniques [25].

Our Results. We first present a general technique to solve a wide range of fixed-cardinality optimization problems in graphs with maximum degree Δ. We first show that, in case the solutions to these problems are constrained to be connected graphs, we obtain a running time of $O((e(\Delta - 1))^{k-1} \cdot (\Delta + k) \cdot n) \cdot T(k, G)$, where $T(k, G)$ is the time needed to evaluate ϕ on subgraphs of order k. Then we extend this result to a wider class of objective functions that do not demand connectedness of the solution, but have the property that connected components can be evaluated independently. The running time for these problems becomes $O((4.2(\Delta - 1))^{k-1} \cdot (\Delta + k) \cdot n) \cdot T(k, G)$. The algorithm is randomized with only false negatives and error probability at most $1/e$.

We then turn to the specific problems of μ-Clique and Densest k-Subgraph. Table 1 gives an overview of the results for these two problems; note that all negative results that were obtained for μ-Clique immediately transfer to Densest k-Subgraph. Our results can be summarized as follows. Finding dense subgraphs is significantly harder than finding cliques since μ-Clique and Densest k-Subgraph are W[1]-hard with respect to the parameter (d, k). Furthermore, we show that the W[1]-hardness result for Densest k-Subgraph parameterized by $n - k$ [13] can also be generalized to hold for μ-Clique for all μ, $0 < \mu < 1$. Finally, we show that, in contrast to Densest k-

Subgraph, μ-Clique is fixed-parameter tractable for the parameters maximum degree Δ and h-index h of G. In particular, we show that the practically relevant case of finding subgraphs whose density μ deviates not too much from the maximum density—that is, $1/\mu$ is small—is still tractable for bounded Δ or h.

An experimental evaluation of the algorithm presented here for μ-Clique parameterized by the maximum degree Δ was recently carried out by a superset of the authors [37].

2. Preliminaries

We consider simple undirected graphs $G = (V, E)$ where $n := |V|$ and $m := |E|$. The *order* of a graph is its number of vertices. We use $N(v) := \{u \mid \{u, v\} \in E\}$ to denote the *neighborhood of a vertex* v. For a vertex set $S \subseteq V$ we denote by $N(S) := \bigcup_{v\in S} N(v) \setminus S$ the *neighborhood of* S, and by $\deg(v)$ the *degree* of v. We use $G[S] := (S, \{\{u, v\} \in E \mid u, v \in S\})$ to denote the *subgraph induced by* S. The edge set of a graph G is sometimes also denoted by $E(G)$.

The *degeneracy* of a graph G is the smallest integer d such that every induced subgraph of G has at least one vertex with degree at most d. The *h-index* of a graph G is the maximum integer h such that G contains h vertices of degree at least h. The property of being a μ-clique is not hereditary, but has a "nestedness" property [38, 41]: Every μ-clique $G = (V, E)$ has an induced subgraph G' on $|V| - 1$ vertices that is also a μ-clique.

A *parameterized problem* is a language $L \subseteq \Sigma^* \times \Sigma^*$. The second component is the *parameter*; the problem L is *fixed-parameter tractable* if the instance (I, k) can be solved in $f(k) \cdot \text{poly}(n)$ time, where $f(k)$ is a computable function only depending on k. A *parameterized reduction* reduces a problem instance (I, k) in $f(k) \cdot \text{poly}(|I|)$ time to an instance (I', k') such that (I, k) is a yes-instance if and only if (I', k') is a yes-instance and $k' \leq g(k)$, where g is a function only depending on k. A basic class of presumed fixed-parameter intractability is W[1]. A parameterized problem L is W[1]-hard if there is a W[1]-hard problem L' such that there is a parameterized reduction from L' to L. A *problem kernel* for a parameterized problem P is a polynomial-time computable parameterized reduction R from P to itself, such that there is some computable function g with the property that, if (I, k) reduces to (I', k') under R, then $|I'|, k' \leq g(k)$. The function g is called the *size* of the problem kernel. For more details, we refer the reader to the literature [19, 24, 40].

3. A Fixed-Parameter Algorithm for Cardinality-Constrained Optimization Problems

In this section we present our algorithm for Fixed-Cardinality Optimization. As outlined in the introduction, it is divided into two steps: enumeration of connected subgraphs (in Section 3.1) and a dynamic programming procedure (in Section 3.2). Applied to Densest k-Subgraph, we achieve an asymptotic running-time improvement in the exponential part of the running time in comparison to the Random Separation method [14]. This example and further ones are given Section 3.3. In Section 4 we apply our algorithm to μ-Clique.

3.1. Enumerating Connected Graphs

First, we focus on the case that we only have to consider solutions S such that $G[S]$ is connected. We can find these by a simple but efficient enumeration of all possible solutions. Bounding the running time of the enumeration procedure builds on a result of Bollobás [8, Equation (7)].

Theorem 1 (Bollobás [8])**.** *Let G be a graph with maximum degree Δ and let v be a vertex in G. There are at most $(e(\Delta-1))^{k-1}$ subtrees of order k in G that contain v.*

As Bollobás [8] notes, since each connected graph has a spanning tree, it follows that also the number of connected subgraphs of order k in G that contain v is at most $(e(\Delta-1))^{k-1}$. Building on the above bound, we obtain an enumeration algorithm as follows.

Theorem 2. *Let $G=(V,E)$ be a graph with maximum degree Δ that is represented as an adjacency list data structure and let v be a vertex in G. There are $\mathrm{O}((e(\Delta-1))^{k-1})$ connected k-vertex subgraphs of G that contain v and their vertex sets can be enumerated in $\mathrm{O}((e(\Delta-1))^{k-1}\cdot(\Delta+k))$ time.*

We call a vertex subset $V' \subseteq V$ *admissible* if it induces a connected subgraph of G that contains v and is of order at most k. We describe a tree Γ, called *search tree* below, where each of its nodes represents an admissible set and each admissible set is represented by some node in Γ. We then show that the order of Γ is at most $\mathrm{O}((e(\Delta-1))^{k-1})$ and that computing Γ in a depth-first fashion can be implemented to run in $\mathrm{O}((e(\Delta-1))^{k-1}\cdot(\Delta+k))$ time.

Throughout this section vertex v is fixed. We also fix an arbitrary ordering of the vertices in G which assigns a unique *index* to every vertex. Let us describe the search tree $\Gamma=\Gamma(G,v,k)$. Each of its nodes $\mathcal{N}$ is associated with a tuple $\tau(\mathcal{N})=(P,W)$ where $W\subseteq P\subseteq V$. We show below that P is admissible. Intuitively, the subtree of Γ rooted at a node $\mathcal{N}$ associated with (P,W) represents those admissible supersets of P that do not contain a vertex of $N(W)\setminus P$. For this, we will choose a vertex $u\in P\setminus W$ and generate a child of $\mathcal{N}$ for each subset of the neighbors of u that extend P to another suitable admissible set. To avoid adding further neighbors of u to P deeper in the search tree Γ—which would correspond to traversing some parts of the search space multiple times—we use the set W. This set contains all "already processed" vertices that should not contribute any further neighbors to the admissible set P. Hence, when adding neighbors of u we have to avoid any neighbor of a vertex in W.

Continuing the definition of Γ, the root of Γ is associated with $(\{v\},\emptyset)$ and the remaining nodes are defined inductively as follows. Let $\mathcal{N}$ be any node in Γ with its associated tuple (P,W) such that $|W|<|P|<k$ and $N(P)\setminus N(W)\neq\emptyset$, and let u be the vertex with lowest index in $P\setminus W$. For every subset $M\subseteq N(u)\setminus(N(W)\cup P)$ that fulfills $|M|\leq k-|P|$, we add a new child to $\mathcal{N}$ associated with $(P\cup M, W\cup\{u\})$. (Note that $M=\emptyset$ is one of the choices for M.) There are no further nodes in Γ and this concludes its definition.

Lemma 1. *(i) In each tuple (P,W) associated with some node of Γ the set P is admissible. Furthermore, (ii) for each admissible set P, there is a set W such that (P,W) is associated with some node of Γ.*

Proof. To prove (i) it suffices to observe that the set P associated with the root is admissible and that, if P is admissible for some node $\mathcal{N}$, then the set P for each of his children is. The first part is obvious. For the second part, let $\mathcal{N}$ be associated with (P, W) and let one of its children be associated with (P', W'). Since $G[P]$ is connected and since $P' \setminus P \subseteq N(u)$, where u is the vertex with lowest index in P, also $G[P']$ is connected. By definition of Γ moreover $|P' \setminus P| \leq k - |P|$ and hence, P' contains at most k vertices. Thus, we have proved (i).

It remains to prove (ii). Assume for the sake of contradiction that for some admissible set P there is no node in Γ associated with the set (P, W) for any set W. Consider a node $\mathcal{N}'$ with associated tuple (P', W') in Γ with the longest path to the root such that $P' \subseteq P$ and $N(W') \cap (P \setminus P') = \emptyset$; clearly, such a node exists. Consider the vertex $u' \in P' \setminus W'$ with lowest index and $C = N(u') \cap (P \setminus P')$. Note that $C \cap N(W') = \emptyset$ as $N(W') \cap (P \setminus P') = \emptyset$. Then, the child of $\mathcal{N}'$ with associated tuple $(P' \cup C, W' \cup \{u'\})$ also fulfills our condition on its tuple but has a longer path to the root. This contradicts our choice of $\mathcal{N}'$. □

In order to bound the running time of constructing Γ, we divide the nodes of Γ into different types. The first type are *interesting leaves* which are leaves $\mathcal{N}$ of Γ with $\tau(N) = (P, W)$ such that $|P| = k$. The second type are *boring leaves* which are the remaining leaves. Note that for boring leaves we have $|P| < k$ and $P = W$. The third type are the parents of the interesting leaves and the fourth type are all remaining inner nodes which we call *deep inner nodes.*

Lemma 2. *The search tree Γ is of order at most $\left(3 + \frac{2}{2^{\Delta-1}-2}\right)(e(\Delta-1))^{k-1}$. Moreover, the number of interesting leaves and number of parents of interesting leaves is at most $(e(\Delta-1))^{k-1}$ each, there are at most $\frac{1}{2^{\Delta-1}-2}(e(\Delta-1))^{k-1}$ deep inner nodes, and there are at most $\left(1 + \frac{1}{2^{\Delta-1}-2}\right)(e(\Delta-1))^{k-1}$ boring leaves.*

Proof. By definition, each node $\mathcal{N}$ with associated tuple $\tau(\mathcal{N}) = (P, W)$ and vertex u with lowest index in $P \setminus W$ has one child for every subset of $R(\mathcal{N}) := N(u) \setminus (P \cup N(W))$ of size at most $k - |P|$. Hence, node $\mathcal{N}$ has at most $\sum_{i=0}^{\min\{|R(\mathcal{N})|, |P|-k\}} \binom{|R(\mathcal{N})|}{i}$ children. In particular, the root of Γ has at most 2^{Δ} children and each deep inner node has at most $2^{\Delta-1}$ children (as some neighbor of u has to be in P already). Assume for the moment that a search tree $\Gamma^* = \Gamma^*(G^*, v^*, k)$ exists that achieves the maximum possible number of children in each node. Clearly, search tree Γ^* contains at least as many nodes as Γ, because search tree Γ can be embedded into Γ^* in the natural way. Search tree Γ^* also contains at least as many deep inner nodes. Furthermore, every boring leaf $\mathcal{N}$ of search tree Γ can be mapped to a unique boring leaf of Γ^*: take the natural embedding of Γ into Γ^*, the node $\mathcal{N}^*$ to which $\mathcal{N}$ corresponds in Γ^* and then recursively follow the (unique) child of $\mathcal{N}^*$, $\tau(\mathcal{N}^*) = (P^*, W^*)$ such that no vertex is added to P^*. Hence, search tree Γ^* contains at least as many boring leaves than Γ. Similarly, search tree Γ^* also contains at least as many interesting leaves.

It thus suffices to bound the numbers of nodes of Γ^* instead of the corresponding numbers of nodes of Γ. Let us fix a concrete G^* and v^* in $\Gamma^* = \Gamma^*(G^*, v^*, k)$ (in particular, let us show that Γ^* as defined above exists). Take as G^* any tree that is rooted at some vertex v^*, such that each vertex in G^* has degree exactly Δ and that has depth at least $k-1$. As there are no two distinct paths from v^* to any other vertex, in each node $\mathcal{N}$

of Γ^* with associated tuple $\tau(\mathcal{N}) = (P, W)$ and vertex u with lowest index in $P \setminus W$ we indeed have $|N(u) \setminus (P \cup N(W))| = |N(u) \setminus P| = |N(u)| - 1 = \Delta - 1$, except for the root, where we have $|N(u) \setminus (P \cup N(W))| = |N(u)| = \Delta$. Hence, Γ^* achieves the maximum possible number of children in each node.

We first bound the number of interesting leaves in Γ^*. Since $|P| = k$ in an interesting leaf $\mathcal{N}$ with associated tuple (P, W), graph $G^*[P]$ induces an order-k subtree of G^* that contains v^*. This mapping from the leaves to the set of order-k subtrees of G^* that contain v is injective: Assume the contrary, that is, there are two interesting leaves $\mathcal{N}_1, \mathcal{N}_2$ of Γ^* that are mapped to the same tree. Consider the node $\mathcal{N}$ in Γ^* with the longest path to the root such that $\mathcal{N}$ lies on both paths from the root to $\mathcal{N}_1$ and $\mathcal{N}_2$. Consider the tuple $\tau(\mathcal{N}) = (P, W)$ associated with $\mathcal{N}$, the successors $\mathcal{N}_1', \mathcal{N}_2'$ of $\mathcal{N}$ on the corresponding paths, and their associated tuples $\tau(\mathcal{N}_1') = (P_1, W_1)$, $\tau(\mathcal{N}_2') = (P_2, W_2)$. There is a vertex w in exactly one of P_1, P_2 that is not in P, say $w \in P_1 \setminus P_2$. Thus, in the tree induced by leaf $\mathcal{N}_1$, the vertex w is present whereas in the tree induced by $\mathcal{N}_2$ the vertex w is missing. This is a contradiction to our assumption.

Hence, there is an injective mapping from the interesting leaves of Γ^* to the set of order-k subtrees of G^* that contain v^* and, invoking Theorem 1, their number is at most $(e(\Delta - 1))^{k-1}$. Clearly, the set of parents of interesting leaves also has at most this cardinality.

We next aim to bound the number of deep inner nodes of Γ^*. First note that any inner node of Γ^* has at most one boring leaf as child. Hence, if we remove the leaves from Γ^*, we obtain a tree Γ' in which each inner node has at least $2^{\Delta-1} - 1$ children. Furthermore, as in each non-root node $\mathcal{N}$ of Γ' with associated tuple (P, W) we have $|N(u) \setminus (P \cup N(W))| = |N(u)| - 1$, each node $\mathcal{N}$ is the ancestor of a parent of a interesting leaf in Γ^*. Thus the leaves in Γ' are a subset of the parents of interesting leaves in Γ^*, that is, there are at most $(e(\Delta - 1))^{k-1}$ leaves in Γ'. Using that each inner node in Γ' has at least $2^{\Delta-1} - 1$ children, the number of inner nodes in Γ' is at most $\frac{1}{2^{\Delta-1}-2}(e(\Delta - 1))^{k-1}$ which is also the number of deep inner nodes in Γ^*.

Finally, as noted above, each inner node of Γ^* has at most one boring leaf as child. Hence, the number of boring leaves is at most $\left(1 + \frac{1}{2^{\Delta-1}-2}\right)(e(\Delta - 1))^{k-1}$. □

A corollary of Lemma 2 is that the number of subgraphs of G that contain v and are of order *at most* k is at most $3(e(\Delta - 1))^{k-1}$. Lemma 2 now yields a proof for Theorem 2.

Proof of Theorem 2. We construct Γ in a depth-first fashion, computing the associated tuples of each node as follows. Clearly, the associated tuple of the root is given. For every node, we will compute the associated tuples of all children. Hence, when processing a node of Γ, we may assume that its tuple is given. We represent P and W in a tuple (P, W) as well as all neighbors $N(W)$ of vertices in W as a set data structure that allows for addition and lookup of single elements in $\mathrm{O}(1)$ time [11].

Let us describe the procedure for a certain node $\mathcal{N}$ associated with tuple $\tau(\mathcal{N}) = (P, W)$. We first report P as admissible set, which is correct by Lemma 1(i) and can be done in $\mathrm{O}(k)$ time. Then we create a list containing the elements of $N(u) \setminus (P \cup N(W))$ in arbitrary order, where u is the vertex with lowest index in $P \setminus W$; vertex u is retrievable in $\mathrm{O}(k)$ time. It is possible to create the list in $\mathrm{O}(\Delta)$ time, because of the set data structures used for P and $N(W)$. Once we created the list of $N(u) \setminus (P \cup N(W))$, we iterate over all its sublists L corresponding to some set M with $|M| \leq k - |P|$ and, for each of them, make a recursive call for the child with associated tuple $(P \cup M, W \cup \{u\})$. The recursive call includes computing the tuple, which is possible in $\mathrm{O}(k)$ time, and updating the set data structure for the vertices in $N(W)$, possible in $\mathrm{O}(\Delta)$ time. As creating the sublists is possible in time linear in the number of sublists, processing each node $\mathcal{N}$ thus takes $\mathrm{O}((\Delta + k)) \cdot (C(\mathcal{N}) + 1)$ time where $C(\mathcal{N})$ is the number of children of $\mathcal{N}$ in Γ.

We now derive the overall running time for constructing Γ. Clearly, each leaf of Γ contributes $\mathrm{O}(\Delta + k)$ processing time, amounting to $\mathrm{O}((e(\Delta-1))^{k-1}(\Delta + k))$ overall by Lemma 2. The overall processing time contributed by the parents of the leaves is $\mathrm{O}(\Delta + k)$ times the number of leaves, hence yielding the same asymptotic upper bound. Next, each remaining node in Γ has at most 2^{Δ} children and hence contributes $\mathrm{O}(2^{\Delta}(\Delta + k))$ processing time. Using the bound on their number from Lemma 2, their overall contribution is $\mathrm{O}(2^{\Delta}(\Delta + k)) \cdot \frac{1}{2^{\Delta-1}-2} \cdot (e(\Delta-1))^{k-1} = \mathrm{O}((e(\Delta-1))^{k-1} \cdot (\Delta + k))$. □

While the above enumeration scheme seems quite simple, its running time is asymptotically almost optimal. It is not hard to show that choosing G to be a $(\Delta - 1)$-ary tree yields many admissible sets, closely matching the upper bound of Lemma 2. We show in Appendix A that G has at least $\frac{1}{(\Delta-2)k+1}\binom{(\Delta-1)k}{k}$ admissible sets. Let us derive bounds for the binomial coefficient occurring here (the upper bound is also needed below in Section 4).

Proposition 1.

$$\frac{1}{e^2}\sqrt{\frac{\ell}{2\pi(\ell-1)k}}\left(\frac{\ell}{\ell-1}\right)^{\ell k}(\ell-1)^k \le \binom{\ell k}{k} \le e\sqrt{\frac{\ell}{2\pi(\ell-1)k}}\left(\frac{\ell}{\ell-1}\right)^{\ell k}(\ell-1)^k.$$

The proof is given in Appendix B. Using this estimate we obtain that in the worst case the number of admissible sets can be at least

$$c\left(\frac{\Delta-1}{\Delta-2}\right)^{(\Delta-1)k} \cdot \frac{(\Delta-2)^k}{\Delta k^{3/2}}$$

for some constant $c > 0$ independent of k and Δ. Let us consider the ratio ρ "upper bound divided by lower bound" for fixed k. The kth root of the ratio ρ is

$$\Theta\left(\frac{e(\Delta-1)}{\left(\frac{\Delta-1}{\Delta-2}\right)^{\Delta-1}(\Delta-2)}\right) = \Theta\left(\frac{e}{\left(\frac{\Delta-1}{\Delta-2}\right)^{\Delta-2}}\right) = \Theta\left(\frac{e}{\left(1+\frac{1}{\Delta-2}\right)^{\Delta-2}}\right)$$

and, as $\lim_{\Delta\to\infty}(1 + 1/(\Delta-2))^{\Delta-2} = e$, the ratio approaches some constant.

Even though the number of connected subgraphs of a fixed order of a graph is a fundamental issue in graph theory, we are not aware of an algorithmic treatment as above. Katrenic and Schiermeyer [33], Bonnet et al. [9] and Hermelin et al. [28, Lemma 5] use connected subgraph enumeration as a subroutine but they give a worse upper bound of $\mathrm{O}(\Delta^{2k})$. Using Theorem 2, we can achieve the following by simply starting the enumeration process from each vertex in G.

Theorem 3. *Let (G, ϕ, k) be an instance of* Fixed-Cardinality Optimization *such that*

1. G has maximum degree Δ,
2. $\phi(S) = 0$ if $G[S]$ is not connected,
3. and $\phi(S)$ can be evaluated in $T(k, G)$ time.
Then, (G, ϕ, k) can be solved in $\mathrm{O}((e(\Delta - 1))^{k-1} \cdot (\Delta + k) \cdot n) \cdot T(k, G)$ time.

There are many natural Fixed-Cardinality Optimization problems that fulfill the conditions of Theorem 3. For example, if we take $\phi_G(S)$ to be the diameter of $G[S]$, we arrive at an optimization version of the s-Club problem [3, 4, 44] which asks to find a subgraph of order k that has diameter at most s. Note that this problem is nontrivial only if $k > \Delta$. As a consequence, the above theorem improves on the previously reported overall running time of $\mathrm{O}((k-2)^k \cdot k! \cdot kn + nm)$ for s-Club parameterized by k [44].

We note that for this specific set of special cases of Fixed-Cardinality Optimization, we obtain, compared to the random separation method, an improved exponential part of the running time. Moreover, our algorithm is deterministic and runs in linear time for fixed values of Δ and k if $T(k, G) \leq f(k, \Delta)$ for some function f.

3.2. Combining Solutions from Different Connected Graphs

In the case of general objective functions ϕ, the optimal solution for ϕ could be a vertex set S such that $G[S]$ is not connected. This is in particular the case for Densest k-Subgraph: If the input graph consists for example of two cliques of order $k/2$ that are connected by a long path of degree-two vertices, then, for sufficiently large k, the optimal solution consists exactly of the two cliques and it is thus disconnected. However, the objective function of Densest k-Subgraph has, as many objective functions for Fixed-Cardinality Optimization, the useful property that connected components can be considered independently when evaluating the function. This can be formalized in a general way as follows.

Definition 1. *Let $G = (V, E)$ be a graph, let $\phi \colon 2^V \to \mathbb{Q}$ be an objective function, let $S, T \subseteq V$ be arbitrary disjoint vertex sets, and let $\chi \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ such that χ is nondecreasing in both arguments individually. We call ϕ* component sub-χ *if the functions ϕ and χ fulfill that $\phi(S) \leq \chi(\phi(W), \phi(S \setminus W))$ where W is an arbitrary connected component of $G[S]$. Furthermore, we call ϕ* super-χ *if $\phi(S \cup T) \geq \chi(\phi(S), \phi(T))$. If both properties hold for ϕ, we call it* component χ-linear.

Informally, the idea behind component χ-linearity is as follows. The value of $\phi(S)$ can be evaluated by first evaluating subsets $S_1, S_2 \subseteq S$ of S. If ϕ is component χ-linear, the only possibility that the value of $\phi(S)$ is much larger than a "combination" of $\phi(S_1)$ and $\phi(S_2)$ is if there are edges between S_1 and S_2. Hence, connected components of $G[S]$ can be first evaluated separately and then be combined.

It is easy to check that, taking $\phi(S)$ as the number of edges in $G[S]$ and $\chi(a, b) = a + b$ we obtain an objective function corresponding to Densest k-Subgraph which is component $+$-linear. In the following, we extend our enumeration algorithm to Fixed-Cardinality Optimization with component χ-linear objective functions. We then demonstrate in Section 3.3 why we think that the above general definition of component χ-linearity is useful. Our algorithm for Fixed-Cardinality Optimization, based on the *color coding* technique [1], is randomized with false negatives. It can be derandomized with an additional running time factor of $2^{O(k)} \cdot \log n$ [1].

Let S be a vertex set of order k such that $\phi(S)$ is maximum. The basic idea of color coding is to color the vertices of the input graph uniformly at random with a set C of k colors and to hope that S is *colorful*, that is, for each color in C there is exactly one vertex in S that has received this color. With some nonzero probability, the coloring will satisfy this property. Assuming the graph is colored this way, first use the enumeration algorithm to find all connected components of $G[S]$. Then combine these connected graphs by applying dynamic programming. The color-coding/enumeration/dynamic programming routine is repeated sufficiently enough to achieve constant error probability. The details are as follows.

Apply the coloring to the vertex set. After the coloring, first compute for every subset C' of C the connected subgraph $G[S']$ that maximizes $\phi(S')$ among all connected subgraphs whose vertices have color set C'. This can be easily achieved by adapting the enumeration algorithm of Theorem 2 to only report colorful connected graphs and then evaluating ϕ for each enumerated colorful graph $G[S']$. We fill a table $\mathcal{D}$ storing the currently best value for each color subset as follows. We initialize $\mathcal{D}$ by setting $\mathcal{D}(C') := -\infty$ for all $C' \subseteq C$. If during the enumeration we find some S' with color set C' and $\mathcal{D}(C') \leq \phi(S')$, then we set $\mathcal{D}(C') \leftarrow \phi(S')$. After the enumeration of all connected subgraphs the entry $\mathcal{D}(C')$ contains exactly the maximum objective value among all connected subgraphs with color set C'. Afterwards, we find the maximum objective value of any graph with color set C using another table $\mathcal{T}$. Here, the entry in $\mathcal{T}(C')$ for some color set $C' \subseteq C$ contains the maximum of $\phi(S)$ for all vertex sets S with color set C'. We can fill $\mathcal{T}(C')$ by the following recurrence:

Lemma 3.

$$\mathcal{T}(C') = \max\{\mathcal{D}(C'), \max_{C'' \subset C'} \chi(\mathcal{T}(C''), \mathcal{T}(C' \setminus C''))\}.$$

Proof. We first prove that the left hand side is at most as large as the right hand side and then the other direction.

"$\leq$": Let $S \subseteq V$ be a "witness" for $\mathcal{T}(C')$, that is, $G[S]$ is colorful with color set C' and $\phi(S) = \mathcal{T}(C')$. If $G[S]$ is connected, then $\mathcal{T}(C') = \mathcal{D}(C')$, as required. If $G[S]$ is disconnected, then for an arbitrary connected component W of S we have $\mathcal{T}(C') \leq \chi(\phi(W), \phi(S \setminus W))$ because ϕ is component sub-χ. Since χ is nondecreasing we have

$$\chi(\phi(W), \phi(S \setminus W)) \leq \chi(\mathcal{D}(C''), \mathcal{T}(C' \setminus C'')) \leq \chi(\mathcal{T}(C''), \mathcal{T}(C' \setminus C''))$$

where C'' is the set of colors in $G[W]$. Thus $\mathcal{T}(C') \leq \chi(\mathcal{T}(C''), \mathcal{T}(C' \setminus C''))$.

"$\geq$": We have $\mathcal{T}(C') \geq \mathcal{D}(C')$ by definition. Let S and T, $S \cap T = \emptyset$, be witnesses for $\mathcal{T}(C'')$ and $\mathcal{T}(C' \setminus C'')$, respectively, for a set $C'' \subseteq C'$. Since ϕ is super-χ, we have $\phi(S \cup T) \geq \chi(\phi(S), \phi(T))$ and hence also $\mathcal{T}(C') \geq \chi(\mathcal{T}(C''), \mathcal{T}(C' \setminus C''))$. □

Thus, after filling $\mathcal{T}$ according to the above recurrence, the maximum objective value of any colorful solution of order k is stored in $\mathcal{T}(C)$.

By repeating the above algorithm $O(e^k)$ times one can achieve a constant error probability that in one of the repetitions, called *trials*, the solution has indeed obtained a colorful coloring. If $T(k, G)$ is an upper bound on the time needed to evaluate ϕ, and χ, then this leads to an overall running time of $O((e^2(\Delta - 1))^{k-1} \cdot (\Delta + k) \cdot n) \cdot T(k, G)$ for the algorithm: For each trial the table $\mathcal{T}$ can be evaluated in $\mathrm{O}(3^k) \cdot T(k, G)$ time,[3] and thus the dominating part of the running time is the subgraph enumeration procedure.

In the following, we describe how this running time can be further improved by employing a known speed-up trick for color coding [17, 31]. The idea is to increase the number of colors, that is, to use ck colors, $c > 1$, when coloring the vertices. This modification has two effects. On the one hand, it increases the probability that the solution is colorful which reduces the number of necessary trials. On the other hand, it increases the running time needed for the dynamic programming, since the table now has $\Theta(2^{ck})$ entries. Hence, there is a running time trade-off between the two parts of the algorithm. In our application, we can observe that the dominating part of the running time for each trial is the subgraph enumeration which does not depend on the number of colors but only on k. Hence, it makes sense to increase the number of colors as long as the dynamic programming part is still faster than the subgraph enumeration part. As a result, we can achieve a sizable speed-up, the concrete analysis is as follows.

First, the probability P_c, $c > 1$, that an optimal solution $S \subseteq V$ is colorful when coloring V uniformly at random with ck colors is

$$P_c = \frac{\binom{ck}{k} k!}{(ck)^k} \geq \frac{1}{e^2}\sqrt{\frac{c}{2\pi(c-1)k}}\left(\frac{c}{c-1}\right)^{ck}(c-1)^k\sqrt{2\pi}k^{k+\frac{1}{2}}\frac{1}{(eck)^k}$$
$$= \frac{1}{e^2}\sqrt{\frac{c}{c-1}}\left(\left(\frac{c}{c-1}\right)^{c}\frac{c-1}{ec}\right)^k = \frac{1}{e^2}\sqrt{\frac{c}{c-1}}\left(\left(\frac{c}{c-1}\right)^{c-1}\frac{1}{e}\right)^k$$

where the inequality holds due to Stirling's approximation and Proposition 3. We make $\lceil 1/P_c \rceil$ coloring trials. The probability to have at least one trial in ℓ such that the optimal solution S is colorful is $1 - (1 - P_c)^\ell$. The probability of success is thus at least $1 - 1/e$ because $1 - (1 - P_c)^{1/P_c} \geq 1 - 1/e$. Then the enumeration of the connected subgraphs takes $\mathrm{O}(1/P_c \cdot (e(\Delta - 1))^{k-1} \cdot (\Delta + k) \cdot n \cdot T(k, G))$ time overall, whereas the dynamic programming part contributes $\mathrm{O}(1/P_c \cdot 3^{ck}) \cdot T(k, G)$ time. Hence, if we choose c in such a way that $3^{ck} \in \mathrm{O}((e(\Delta - 1))^{k-1})$, then the overall running time is dominated by the enumeration procedures. We claim that this is the case if we let c be smallest possible such that $c \geq (1 - 1/k) \cdot \log_3(e(\Delta - 1))$ and such that ck is an integer. First we note that it is possible to choose c in this way, that is, $(1 - 1/k) \cdot \log_3(e(\Delta - 1)) \geq 1$. Indeed, this is true if and only if $\log_3 e + \log_3(\Delta - 1) \geq k/(k - 1)$. Note that $\Delta, k \geq 3$ without loss of generality,[4] and thus $\log_3 e + \log_3(\Delta - 1) \geq 3/2$ and $3/2 \geq k/(k - 1)$. Hence we may choose c in this way. We see $3^{ck} \in \mathrm{O}((e(\Delta - 1))^{k-1})$ as follows. Because increasing c by $1/k$ increases the integer part of ck, we have $c \leq (1 - 1/k)\log_3(e(\Delta - 1)) + 1/k$ and thus $3^{ck} \leq 3^{(k-1)\log_3(e(\Delta-1))+1} = 3(e(\Delta - 1))^{k-1}$. Since the probability P_c is monotone

[3] We include the function T in this time bound since the values produced by ϕ and χ could be very large in the general case.

[4] If $\Delta \leq 2$ then the number of order k subgraphs is linear in the number of vertices and if $k \leq 2$ then the number of order-k subgraphs is $\mathrm{O}(\Delta n)$. In both cases enumeration is possible in linear time in the number of subgraphs, and, together with color coding and dynamic programming, it is not hard to obtain a running time of $\mathrm{O}(3^k \cdot \Delta \cdot n) \cdot T(k, G)$ for optimizing the objective function.

Δ	3	4	5	6	10
$\gamma/(\Delta-1)$	4.2	3.8	3.6	3.5	3.4
γ	8.4	11.3	14.4	17.5	29.8

Table 2: Approximate values of the base γ in the exponential running time factor in the improved color coding/dynamic programming routine.

ascending for increasing c, the overall running time is in the order of

$$\begin{aligned}
&\frac{1}{P_{\log_3(e(\Delta-1))}} \cdot (e(\Delta-1))^{k-1} \cdot (\Delta+k) \cdot n \cdot T(k,G) \\
\sim &\left(\left(\frac{\log_3(e(\Delta-1))-1}{\log_3(e(\Delta-1))}\right)^{\log_3(e(\Delta-1))-1} e^2(\Delta-1))\right)^{k-1} \cdot (\Delta+k) \cdot n \cdot T(k,G) \\
= &\left(\left(\log_{e(\Delta-1)} \frac{e(\Delta-1)}{3}\right)^{\log_3 \frac{e(\Delta-1)}{3}} e^2(\Delta-1)\right)^{k-1} \cdot (\Delta+k) \cdot n \cdot T(k,G).
\end{aligned}$$

It seems complicated to bring the base in the exponential term, let us call the base γ, into a more readable form. We give some numerical evaluations in Table 2.

Note that, choosing c as above, the base in the exponential factor of P_c tends to one for increasing Δ. Thus the overall running time of the algorithm tends to $\mathrm{O}((e(\Delta-1))^k \cdot (\Delta+k) \cdot n) \cdot T(k,G)$ as Δ increases. For example, for all $\Delta \geq 5$ we obtain the upper bound of $\mathrm{O}(((e+0.9)(\Delta-1))^k \cdot (\Delta+k) \cdot n) \cdot T(k,G)$. Concrete values for γ are given in Table 2. Concluding, we obtain the following theorem.

Theorem 4. *Let (G,ϕ,k) be an instance of* Fixed-Cardinality Optimization *such that*

1. *G has maximum degree $\Delta > 2$,*
2. *ϕ is component χ-linear,*
3. *and ϕ as well as χ can be evaluated in $T(k,G)$ time.*

Then, (G,ϕ,k) can be solved in $\mathrm{O}(\gamma^{k-1} \cdot (\Delta+k) \cdot n) \cdot T(k,G)$ time, reporting a yes-instance as a no-instance with probability at most $1/e$. Here $\gamma = (\log_\beta(\beta/3))^{\log_3(\beta/3)} e\beta$ and $\beta = e(\Delta-1)$. In particular, $\gamma \leq 4.2 \cdot (\Delta-1)$.

3.3. Concrete Examples for Component Linear Functions

We conclude this section with concrete examples for Definition 1 and Theorem 4. First, the algorithm above applied to Densest k-Subgraph has a worst-case running time of $\mathrm{O}((4.2 \cdot (\Delta-1))^{k-1} \cdot (\Delta+k) \cdot k^2 \cdot n)$. Using random separation, Densest k-Subgraph can be solved in $2^{\mathrm{O}(\Delta k)} \cdot (\Delta+k) \cdot n$ time with one-sided error and constant error probability [14]; our algorithm thus improves on this running time.

Second, an objective function that is *not* component χ-linear for any function χ as in Definition 1 is $\phi(S)$ defined as “maximum size of an independent set in $G[S]$”. To see this, take any function χ that fulfills the two properties demanded by Definition 1. Then, $\chi(\phi(S),\phi(T)) \leq \max\{\phi(S),\phi(T)\}$: For any i and j, the complete bipartite graph with partite sets A and B of size i and j, respectively, fulfills $\phi(A \cup B) = \max\{|A|,|B|\} =$

$\max\{\phi(A), \phi(B)\}$. Since ϕ is super-χ we thus have $\chi(i, j) \leq \max\{i, j\}$. However, for vertex sets S and a connected component W in $G[S]$ we have

$$\phi(S) = \phi(W) + \phi(S \setminus W) \leq \chi(\phi(W), \phi(S \setminus W)) \leq \max\{\phi(W), \phi(S \setminus W)\}$$

because ϕ is component sub-χ. This is absurd because for nonempty W and $S \setminus W$ we clearly have $\phi(W) + \phi(S \setminus W) > \max\{\phi(W), \phi(S \setminus W)\}$.

Third, we demonstrate the merit of stating Definition 1 in the present general way. Suppose we are given a graph $G = (V, E)$ and are asked to decide whether among the densest k-vertex subgraphs of G, there is one with a connected component of size at least ℓ. We may define a corresponding objective function

$$\phi(S) := |E(G[S])| \cdot 2^{\lceil \log k \rceil} + \max\{|W| \mid W \text{ is a connected component of } G[S]\},$$

that is, the last $\lceil \log k \rceil$ bits of $\phi(S)$ are reserved for the size of a largest connected component of $G[S]$. Then we can define a function $\chi(a, b)$ that sums the first bits and takes the maximum of the last $\lceil \log k \rceil$ bits. This function is monotone ascending in both arguments and it is easy to check that ϕ is component sub-χ. To see that it is super-χ it suffices to observe that the number of edges in $G[S \cup T]$ is at least as large as the sum of the edges in $G[S]$ and $G[T]$ for disjoint S, T and similarly for the size of the largest component. Hence ϕ is component χ-linear and we may apply Theorem 4 to decide the above property of G.

Finally, we note that our framework is applicable to a subclass of Fixed-Cardinality Optimization, the so-called fixed-cardinality graph partitioning problems [9, 45]. In these problems a graph $G = (V, E)$ and an integer p is given and the task is to decide if there is a k-vertex subset $S \subseteq V$ such that $\phi(S) := a|E(S)| + b|\delta(S)|$ is at least p. Here, a, b, and k are fixed constants, $E(S)$ is the set of edges in $G[S]$, and $\delta(S)$ is the set of edges with exactly one endpoint in S. It is not hard to show that, whenever $a \geq 2b$, then ϕ is component $+$-linear (see Appendix C). Thus Theorem 4 is also applicable to any fixed-cardinality graph partitioning problem with $a \geq 2b$. Combining Theorem 4 with an $\mathrm{O}(\Delta^k \cdot \mathrm{poly}(n))$-time algorithm of Bonnet et al. [9] for the case $a \leq 2b$ yields an algorithm for general fixed-cardinality graph partitioning problems with running time $\mathrm{O}(\gamma^{k-1} \cdot \mathrm{poly}(n))$, where γ is as defined in Theorem 4. For all $\Delta \geq 4$ the resulting algorithm improves on the running time upper bound $\mathrm{O}(4^{k+\mathrm{o}(k)} \cdot \Delta^k \cdot \mathrm{poly}(n))$ achieved recently by Shachnai and Zehavi [45].

4. Application to μ-Clique

We now describe how to use the algorithms presented for Fixed-Cardinality Optimization in order to obtain fixed-parameter algorithms for μ-Clique. More precisely, this will lead to fixed-parameter algorithms for the parameters maximum degree Δ of G and the h-index of G, and for the combined parameter that comprises $n - k$ and degeneracy d of G. Before presenting these algorithms, we observe relationships between the order of μ-cliques and the sparsity parameters under consideration.

4.1. Upper-Bounding the Solution Size

The relation between the order of a μ-clique and its maximum degree, h-index and degeneracy is as follows.

Lemma 4. *A μ-clique with*

(i) maximum degree Δ has order at most $\Delta/\mu + 1$.

(ii) h-index h has order at most $\frac{h\cdot(h-1)+2\cdot(n-h)\cdot h}{\mu\cdot(n-1)} < \frac{2\cdot h}{\mu}$.

(iii) degeneracy d has order less than $(4 \cdot d + \mu)/2 \cdot \mu$.

Proof. Let G be a μ-clique of order n.

(i): If G has maximum degree Δ it has at most $n \cdot \Delta/2$ edges. Since its density is at least μ it has at least $\mu \cdot \binom{n}{2}$ edges. Combining the two statements we get

$$\mu \cdot \frac{n\cdot(n-1)}{2} \leq \frac{n \cdot \Delta}{2}$$

and thus $n \leq \Delta/\mu + 1$.

(ii): If G has h-index h it has at most h vertices of degree more than h. Hence, at least $n - h$ vertices have degree at most h. Let H denote the set of at most h vertices that have degree more than h, and let E_H denote the set of edges with both endpoints in H. Clearly, $|E_H| \leq \binom{h}{2}$. Since all vertices in $V \setminus H$ have degree at most h, there can be at most $(n-h)\cdot h$ edges incident with these vertices. Hence, the total number of edges in G is at most $\binom{h}{2} + (n-h)\cdot h$. Combining this with the lower bound $\mu \cdot \binom{n}{2}$ for the edge number of G we get

$$\mu \cdot \frac{n\cdot(n-1)}{2} \leq \frac{h\cdot(h-1)}{2} + (n-h)\cdot h$$

and thus

$$n \leq \frac{h\cdot(h-1) + 2\cdot(n-h)\cdot h}{\mu\cdot(n-1)} \leq \frac{2\cdot h}{\mu}.$$

(iii): If G is d-degenerate, it has at most $d \cdot (n - \frac{d+1}{2})$ edges. Thus,

$$\mu \cdot \frac{n\cdot(n-1)}{2} \leq d \cdot (n - \frac{d+1}{2})$$

which implies

$$n \leq \frac{2\cdot d + \mu + \sqrt{4d^2 - 4d^2\mu + \mu^2}}{2\cdot\mu} < \frac{4\cdot d+\mu}{2\cdot\mu}.$$

□

The upper bound $\frac{h\cdot(h-1)+2\cdot(n-h)\cdot h}{\mu\cdot(n-1)}$ on the order of μ-cliques is tight as a graph consisting of a clique of order h and of $n-h$ further vertices that are an independent set but adjacent to all vertices of the clique has density exactly μ if n is equal to the upper bound. It is not hard to see that also the upper bound for the order with respect to the maximum degree is tight. The bound with respect to the degeneracy can be improved slightly, but this is not the main focus of this article.

4.2. Parameterization by Maximum Degree

The fixed-parameter algorithms for this parameterization can be obtained by a straightforward application of the generic algorithms described in Section 3 with the size bounds given by Lemma 4.

In most application settings for μ-Clique, one would add the further constraint that the solution has to induce a connected subgraph. By Lemma 4, we have $k \le \Delta/\mu + 1$. Furthermore, for each vertex set $S \subseteq V$ of order at most k, we can compute the number of edges in $G[S]$ in $\mathrm{O}(k \cdot \Delta)$ time: for each vertex v of S find all neighbors of v in S by traversing its adjacency list once. Hence, $T(k, G)$, the time needed to compute the objective function, is $\mathrm{O}(k \cdot \Delta)$ in this case. Plugging both bounds into the time bound given by Theorem 3, we obtain the following.

Proposition 2. *For any fixed μ, $0 < \mu < 1$, we can determine in $\mathrm{O}((e(\Delta-1))^{\Delta/\mu} \cdot \Delta^3/\mu^2 \cdot n)$ time whether G contains a connected μ-clique of order k. Herein, Δ is the maximum degree of G.*

For the general case, in which the solution S may be disconnected, we may use the running time given in Theorem 4 instead, since the objective function “number of edges in a graph” is component linear.

Theorem 5. *For any fixed μ, $0 < \mu < 1$, μ-Clique can be solved in time $\mathrm{O}((4.2 \cdot (\Delta - 1))^{\Delta/\mu} \cdot \Delta^3/\mu^2 \cdot n)$, reporting a yes-instance as a no-instance with probability at most $1/e$. Herein, Δ is the maximum degree in the input graph.*

4.3. Parameterization by h-index.

We now describe how to extend our fixed-parameter results to also hold for the parameter h-index of the input graph. In many practical applications the h-index is much smaller than the maximum degree. For example, social and biological networks have few so-called hubs, that is, vertices of very high degree, and many low-degree vertices. Hence, the h-index is a better parameter than the maximum degree Δ for these graphs.

The main idea of the algorithm is as follows. Let H be the set of the h vertices with degree at least h, and assume that S is a vertex set of size k such that $G[S]$ is a μ-clique. First, by trying all 2^h subsets of H, guess the set H_S of vertices that are in $S \cap H$. Fix one such set H_S. It remains to determine which vertices of $V \setminus H$ belong to S. The number of edges in S depends on the number of edges between $S \setminus H_S$ and H_S and of the number of edges between vertices of H_S. Hence, our goal in the following is simply to find a subgraph of $V \setminus H$ that maximizes this number.

Accordingly, we compute for every vertex $v \in V \setminus H$ the number $\deg_{H_s}(v)$ of neighbors of v in H_S. Define the value $\phi(S')$, $S' \subseteq V \setminus H$, of a subgraph $G[S']$ of the graph $G[V \setminus H]$ to be

$$\phi(S') := |E(G[S'])| + \sum_{v \in S'} \deg_{H_s}(v).$$

The task is to find a vertex set $S' \subseteq V \setminus H$ of order $k - |H_S|$ that maximizes $\phi(S')$. The overall maximum number of edges for any subgraph of order k containing H_S is then this value plus the number of edges in $G[H_S]$. The overall optimum solution is simply the maximum among all possible choices of H_S.

The running time of this algorithm can be bounded as follows. We try 2^h different possibilities for H_S. For each possibility, we first compute $\deg_{H_s}(v)$ for each vertex in $V \setminus H$ which can be performed in $\mathrm{O}(h \cdot n)$ time since all vertices in $V \setminus H$ have degree at most h. Then, we solve Fixed-Cardinality Optimization with ϕ as defined above. Clearly, ϕ is component $+$-linear. Furthermore, by Lemma 4 we have $k <$

$2h/\mu$ and thus also $k - |H_S| < 2h/\mu$. Since the maximum degree in $G[V \setminus H]$ is h, solving each instance of Fixed-Cardinality Optimization can thus be performed in $\mathrm{O}((4.2 \cdot (h-1))^{2h/\mu} \cdot h^2/\mu^2 \cdot n)$ time.

Altogether, we obtain the following.

Theorem 6. *μ-Clique can be solved in time $\mathrm{O}(2^h \cdot (4.2 \cdot (h-1))^{2h/\mu} \cdot h^2/\mu^2 \cdot n)$, reporting a yes-instance as no-instance with probability at most $1/e$ where h is the h-index of the input graph.*

4.4. Parameterization by Degeneracy and Dual Parameter

Our final application of the generic algorithm will lead to a fixed-parameter tractability result for Densest k-Subgraph parameterized by the combined parameter degeneracy d and $\ell := n - k$. As we will show in the next section, it is not possible to achieve fixed-parameter tractability for either d or ℓ alone. Hence, it is interesting to study their combination. Recall that in μ-Clique we fix some constant minimum density μ of the sought graph. This is necessary to bound the maximum value of k and, ultimately, obtain feasible running time bounds. For the combined parameter (d, ℓ) this constraint can be dropped leading to an algorithm for Densest k-Subgraph. The algorithm is mainly based on the following observation.

Lemma 5. *Let $G = (V, E)$ be a graph and let $S \subseteq V$ such that $|S| = k$ and $G[S]$ has maximum density among all subgraphs of order k. Then, there is no vertex in $V \setminus S$ that has degree at least $\ell + d$, where $\ell = n - k$.*

Proof. Assume that there is a vertex v of degree at least $\ell + d$ in $V \setminus S$. Since v has at most $\ell - 1$ neighbors in $V \setminus S$, it has at least $d + 1$ neighbors in S. However, because G is d-degenerate, there is a vertex u of degree at most d in $G[S]$. Thus, $G[(S \setminus \{u\}) \cup \{v\}]$ is a graph with at least one edge more than $G[S]$. This contradicts the fact that $G[S]$ is densest possible. □

Note that we can regard Densest k-Subgraph as the problem of deleting a set of exactly ℓ vertices while removing the least possible number of edges. In other words, we aim to solve a minimization variant of Fixed-Cardinality Optimization where ϕ is defined as

$$\phi(S) := \left(\sum_{v \in S} \deg(v)\right) - |E(G[S])|.$$

We can translate this easily into a maximization variant by changing the sign of the function and adding a normalizing term of n to the contribution of each vertex. Formally, we aim to solve Fixed-Cardinality Optimization with

$$\phi(S) := \left(\sum_{v \in S} n - \deg(v)\right) + |E(G[S])|.$$

Note that this objective function is component +-linear: The first part of the sum is independent of the edges in the subgraph and the second part is simply a sum of the edges. Hence, if a set S consists of two connected components, the value of $\phi(S)$ is the sum of the objective values of the two, that is, ϕ is component sub-+. Further, if S and T

are disjoint, then $\phi(S \cup T)$ is the sum of the two objective values plus the number of edges going between S and T in G meaning that ϕ is super-+. Consequently, we can apply Theorem 4 to the corresponding problem.

By Lemma 5 we can focus on the subgraph G' of G that contains only the vertices of degree at most $\ell + d$. Computing G and computing the value of $n - \deg(v)$ for each vertex of G' can be performed in $O(m)$ time. Further, the solution size is constrained to be exactly ℓ. Finally, evaluating $\phi(S)$ for a vertex set in S can be performed in $\mathrm{O}((\ell+d)\cdot|S|)$ time. Altogether, this results in the following.

Theorem 7. Densest k-Subgraph *can be solved in* $\mathrm{O}((4.2\cdot(\ell+d-1))^{\ell}\cdot\ell^2\cdot n+m)$ *time, reporting a yes-instance as no-instance with probability at most* $1/e$*, where* $\ell := n-k$ *and* d *is the degeneracy of the input graph.*

5. Hardness Results

In this section, we present two reductions that show the limits of the approach presented above. That is, we show that we cannot replace the h-index by the smaller parameter in Theorem 6, retaining fixed-parameter tractability. Similarly, it is not possible to drop the degeneracy or ℓ from the parameterization in Theorem 7. It is noteworthy that the second result stands in contrast to Clique which does admit a fixed-parameter algorithm with respect to ℓ.

5.1. W[1]-hardness for Parameterization by Dual

First, we show that considering only the dual parameter ℓ leads to W[1]-hardness also in the case of μ-Clique.

Theorem 8. *For any fixed* μ*,* $0 < \mu < 1$*,* μ-Clique *is W[1]-hard with respect to the parameter* $\ell = n - k$*.*

To present the corresponding reduction we need to construct a gadget graph of a given density and some further properties as follows.

Lemma 6. *Given four positive integers* a*,* b*,* c*, and* d*, where* $a < b$ *and* $d \leq c(c-1)/2$*, we can construct in* $\mathrm{poly}(a,b,c,d)$ *time a graph* G *such that*
– G *is* $2(a-1)$*-connected, has maximum degree at most* $2a$*, and*
– adding c *vertices and* d *edges to* G *results in a graph that has density exactly* a/b *and has average degree more than* a*.*

The proof is similar to the proof in [26, Lemma 2]. We make a small tweak to obtain bounded maximum degree.

Proof of Lemma 6. Without loss of generality, assume that $b - a > 1$ and that $a > 1$. Otherwise we can show the claim using $2a$ instead of a and $2b$ instead of b. We set the number of vertices of G to $n := (2b-1)c$ and the number of edges to $m := ac(2bc-1) - d$. First, since

$$\begin{aligned} 2m < 2ac(2bc-1) &\leq 2(b-2)c(2bc-1) \\ &= (2bc-4c)(2bc-1) \\ &< (n-3c)(n+c) \\ &< n(n-1) \end{aligned}$$

there is indeed a simple graph with the claimed number of edges. Moreover, since

$$\frac{m}{n} = \frac{ac(2bc-1)-d}{(2bc-1)c} = a - \frac{d}{(2bc-1)c},$$

it holds that $a \geq m/n > a-1$. Thus, we can first add $(a-1)n$ edges to G in polynomial time such that G is $2(a-1)$-connected and has maximum degree at most $2(a-1)$ [27]. Then, we add the remaining at most n edges to G such that they allow for a partition into two matchings, increasing the maximum degree by at most two.

The density of the graph G' that results from adding c vertices and d edges to G is

$$\frac{2(m+d)}{(n+c)(n+c-1)} = \frac{2(ac(2bc-1)-d+d)}{(2bc-c+c)(2bc-c-1+c)} = \frac{2ac(2bc-1)}{(2bc)(2bc-1)} = \frac{a}{b}.$$

The average degree of G' follows directly. □

The above construction is also used in Section 5.2. We are now ready to prove Theorem 8.

Proof of Theorem 8. We reduce from the W[1]-hard Clique problem [18, 19]. Let $(G = (V, E), s)$ be an instance of Clique, that is, G is an undirected graph, and we ask whether G contains a clique of order s. Assume without loss of generality that $\mu = a/b$, and that $a = \delta|V|$ for some integer constant $\delta > 2$. In the following, we describe how to construct a graph G^* such that deleting $n-k$ vertices from G^* yields a μ-clique if and only if G has a clique of order s.

The idea of the construction can be roughly described as follows. We add to G a large and dense graph $H = (W, F)$ that has minimum degree much larger than $|V|$. Then, we add edges between H and G such that in the resulting graph G^*, all vertices from V have degree $|V|$. Then we show that G^* is not a μ-clique and that, because of the way H is constructed, only by deleting exactly s vertices from V that induce a clique in G we can obtain a subgraph of G^* that has density μ and at least $|W| + |V| - s$ vertices. The crucial observation that helps proving that we must delete a clique is that

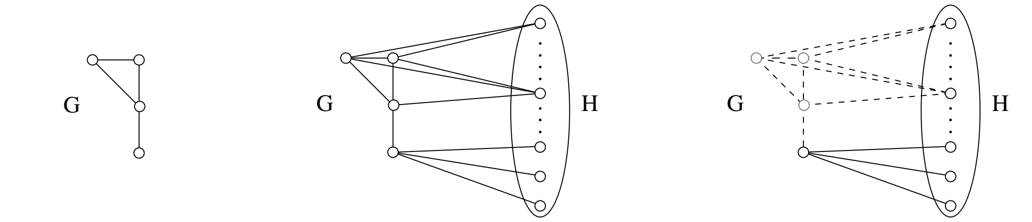
1. by deleting a clique, we remove $|V|+(|V|-1)+\ldots+(|V|-s+1) = |V|\cdot s-(s-1)\cdot s/2$ edges from G^*, and
2. by deleting a set of vertices that is not a clique, we remove more edges from G^*.

Next, we describe the details of the construction. We construct $H = (W, F)$ such that it fulfills the following condition: A graph that contains H as induced subgraph and

- $|V| - s$ additional vertices and
- $|E| + \left(\sum_{v \in V}(|V| - \deg_G(v))\right) - (|V| \cdot s - (s-1) \cdot s/2)$ additional edges

has density exactly $\mu = a/b$. By Lemma 6 we can construct H in $\text{poly}(a, b, |V|)$ time such that it fulfills this condition and such that

- H is $2(a-1)$-connected, and
- a graph that contains H and the above described additional number of vertices and edges has average degree more than a.



(a) The input graph G of the Clique Instance.

(b) The connections to the gadget graph H that is added to G. For each vertex v of G we add $4 - \deg(v)$ edges between v and H.

(c) Deleting in G^* a clique of order three from G deletes $3 \cdot 4 - 3 = 9$ edges from G^*. If a set of three deleted vertices is not a clique in G, more edges are deleted.

Figure 1: An illustration of the reduction presented in the proof of Theorem 8.

Initially, let $G^* := (V \cup W, E \cup F)$ be the disjoint union of the graphs G and H. Then we add for each vertex $v \in V$ exactly $|V| - \deg_G(v)$ edges to G^* between v and H (the neighbors of v in H can be chosen arbitrarily). After adding these edges, every vertex $v \in V$ has degree exactly $|V|$ in G^*. A schematic illustration is presented in Figure 1.

We complete the construction of the μ-Clique instance by setting $k := n - s$ (recall that n denotes the number of vertices in the μ-Clique instance, that is, $n = |V| + |W|$). Clearly, the described construction can be performed in polynomial time. Note that the parameter of the Clique instance is s and the parameter of the μ-Clique instance is $n - k = s$, that is, the reduction is parameter-preserving. To prove the theorem, it remains to show that.

$$(G, s) \text{ is a yes-instance of Clique} \Leftrightarrow (G^*, k) \text{ is a yes-instance of } \mu\text{-Clique}$$

$\Rightarrow$: Let $S := \{v_1, \ldots, v_s\}$ be an order-s clique in G. We show that $G^* - S$ is a μ-clique with k vertices. Clearly, the overall number of vertices in $G^* - S$ is $n - s = k$. The overall number of edges in $G^* - S$ is $|F| + |E| + \big(\sum_{v \in V}(|V| - \deg_G(v))\big) - (|V| \cdot s - (s-1) \cdot s/2)$ which can be seen as follows. First, note that, by construction, G^* has $|F| + |E| + \big(\sum_{v \in V}(|V| - \deg_G(v))\big)$ edges. Hence, it remains to show that the number of edges that have at least one endpoint in S is $|V| \cdot s - (s-1) \cdot s/2$. Since S is a size-s clique and since each vertex $v \in S$ has degree exactly $|V|$ in G^*, each vertex $v \in S$ has exactly $|V| - (s-1)$ neighbors in $(W \cup V) \setminus S$. Furthermore, there are exactly $\binom{s}{2} = (s-1) \cdot s/2$ edges in $G[S]$. Hence, the overall number of edges with at least one endpoint in S is

$$s \cdot (|V| - (s-1)) + (s-1) \cdot s/2) = |V| \cdot s - (s-1) \cdot s/2.$$

The claimed overall number of edges in $G^* - S$ follows. Consequently, $G^* - S$ is a μ-clique.

$\Leftarrow$: Let $S \subseteq W \cup V$ be a vertex set such that $G^* - S$ is a μ-clique of order at least k. Clearly, $|S| \leq s$. We show that G contains an order-s clique by showing the following claims. First, we show that it can be assumed that $S \subseteq V$. Second, we show that $|S| = s$. Finally, we show that $G[S]$ is a clique.

First, we show that it can be assumed that $S \subseteq V$. Suppose that S contains some vertex $w \in W$. Note that w has in G^* at least $4|V| - s > |V|$ neighbors in $W \setminus S$

since H (which is a subgraph of G^*) is $4|V|$-connected (because we have chosen a such that $2(a-1) \geq 4|V|$). Let v be some arbitrary vertex in $V \setminus S$. Clearly, v has degree at most $|V|$ in $G^* - S$. Therefore, deleting the set $S' := \{v\} \cup S \setminus \{w\}$ from G^* yields a graph that has more edges than $G^* - S$ and, obviously, the same number of vertices. Since $G^* - S$ is a μ-clique, so is $G^* - S'$. This "replacement procedure" can be applied as long as S contains a vertex from W. Hence, we can assume without loss of generality that $S \subseteq V$.

Second, we show that $|S| = s$. More precisely, we show that for all vertex sets $S := \{v_1, \ldots, v_i\}$ of size $i < s$ it holds that $G^* - S$ has density less than μ. Let $S' := \{v_1, \ldots, v_i, \ldots, v_s\}$ be an arbitrary size-s superset of S. Note that $G^* - S'$ has at most

$$|F| + |E| + \left(\sum_{v \in V} (|V| - \deg_G(v))\right) - (|V| \cdot s - (s-1) \cdot s/2)$$

edges which can be seen as follows. The number of edges in G^* is $m := |F| + |E| + \left(\sum_{v \in V} (|V| - \deg_G(v))\right)$. The graph $G_1^* := G^* - v_1$ has at most $m - |V|$ edges, since the minimum degree in G^* is $|V|$. Consequently, the graph $G_2^* := G_1^* - v_2$ has at most $m - |V| - (|V| - 1)$ edges, the graph $G_3^* := G_2^* - v_3$ has at most $m - |V| - (|V| - 1) - (|V| - 2)$ edges, and so on. Hence, the number of edges in $G^* - S'$ is at most

$$\begin{aligned} & m - |V| - (|V| - 1) - \ldots - (|V| - s + 1) \\ = {} & |F| + |E| + \left(\sum_{v \in V} (|V| - \deg_G(v))\right) - (|V| \cdot s - (s-1) \cdot s/2). \end{aligned}$$

This means, by construction of H, that $G^* - S'$ has density *at most* μ. For illustrative purposes, suppose that $G^* - S$ is obtained from $G^* - S'$ by adding, one by one, the vertices from $S' \setminus S$. Furthermore, note that $G^* - S'$ has, again by construction of H, average degree more than $a > 2|V|$. Finally, note that the vertices that are added to $G^* - S'$ have degree at most $|V|$ in the final graph $G^* - S$. This means that $G^* - S$ must have density *less than* μ since $G^* - S'$ has average degree at least $2|V|$ and adding a vertex to a graph whose degree is (after it has been added to the graph) less than the average degree of the graph before the vertex has been added reduces the density (note that by adding at most $s < |V|$ vertices to $G^* - S'$ we always produce a graph with average degree more than $|V|$). Summarizing, the graph $G^* - S$ has density less than μ if $|S| < s$.

Finally, we show that $G[S]$ is a clique. Note that, as argued above, $G^* - S$ has at most $F| + |E| + \left(\sum_{v \in V} (|V| - \deg_G(v))\right) - (|V| \cdot s - (s-1) \cdot s/2)$ edges. Hence, by construction of H and by the fact that $G^* - S$ is a μ-clique, it also holds that $G^* - S$ has *exactly* this many edges. Let m_S denote the number of edges in $G[S]$. The number of edges that are removed from G^* by the deletion of S is $\sum_{v \in S} |V| - m_S$, since every vertex has degree exactly $|V|$ in G^*. It follows that $m_S = (s-1) \cdot s/2)$ and, therefore, that $G[S]$ is a clique. □

Somehow counter-intuitively, the reduction used to prove Theorem 8 suggests that in order to obtain a graph with density μ it might be of advantage to delete a clique from the input graph. Hence, one cannot expect that the set of removed vertices induces a sparse graph. From the above reduction, we also obtain a lower bound on the running time of algorithms for μ-Clique. This bound is based on the exponential-time hypothesis (ETH) which assumes that 3SAT cannot be solved in $2^{o(n)}$ time [32, 39].

Theorem 9. *For any fixed μ, $0 < \mu \leq 1$, μ-Clique cannot be solved in time $2^{o(\Delta)} \cdot \mathrm{poly}(n)$ unless the exponential time hypothesis (ETH) fails. Herein, Δ is the maximum degree of the input graph.*

Proof. Unless the ETH fails, Clique does not have algorithms with running time $2^{o(n)}$ [32]. We observe that the reduction used in Theorem 8 produces instances with maximum degree at most cn, where n is the number of vertices in the Clique instance and c a constant. Thus, since the h-index is upper bounded by the maximum degree, if there is an algorithm with the claimed running time for μ-Clique, there is also a subexponential algorithm for Clique and the ETH fails.

To observe the bound on the maximum degree, consider the graph G^* in the construction. Every vertex in G^* that stems from G has degree exactly $n = |V|$. Every vertex in H has degree at most $2a = 2\delta n$ for some constant δ by Lemma 6 and the way we have chosen a in the construction. In the course of the construction, every vertex in H gets at most n additional neighbors that stem from G, and, thus, the maximum degree in G^* is at most $(2\delta + 1)n$. □

Clearly, Theorem 9 also excludes algorithms with running time $2^{o(h)}$ where h is the h-index of the input graph. We remark that the number of vertices in the produced instance can only be bounded by a quadratic polynomial in the number of vertices of the clique instance. Hence, the exclusion of subexponential algorithms for μ-Clique with respect to the number of vertices and $0 < \mu < 1$ does not follow from Theorem 8.

5.2. W[1]-hardness for Parameterization by Degeneracy and Solution Size

Next, we show that the parameter h-index cannot be replaced by the smaller parameter degeneracy.

Theorem 10. *For any fixed μ, $0 < \mu < 1$, μ-Clique is W[1]-hard parameterized by (d, k), where d denotes the degeneracy of the input graph.*

Proof. We reduce from Clique. Let $(G = (V, E), s)$ be an instance of Clique. An equivalent instance of μ-Clique, $0 < \mu < 1$, is constructed from G as follows. First, replace every edge $\{u, v\} \in E$ by a length-two path, that is, remove $\{u, v\}$ from the graph, insert a new vertex $s_{\{u,v\}}$, and make it adjacent to u and v. In the following, let $S := \{s_e \colon e \in E\}$ denote the set of added vertices, and let G_1 denote the graph constructed in this way. Next, we make a useful observation and then continue the description of the construction.

The graph G_1 has the following property.

> An induced subgraph $G_1[K]$ of G_1 of order $s + \binom{s}{2}$ has at most $2 \cdot \binom{s}{2}$ edges.
> In case of equality, $K \cap V$ induces a clique of order s in G.

This can be shown as follows. Let $K^1 := K \cap V$ denote the vertices of K that are also vertices of G, and let $K^2 := K \setminus K^1$ denote the other vertices of K. By construction, every vertex $v \in K^2$ has in $G_1[K]$ degree at most two. Furthermore, every vertex of K^1 has in $G_1[K]$ only neighbors that are in K^2 and vice versa. Consequently, the number of edges in $G_1[K]$ is at most $2 \cdot |K^2|$. Assume that $G_1[K]$ has more than $2 \cdot \binom{s}{2}$ edges. Then, $|K^2| > \binom{s}{2}$ and thus $|K^1| < s$. In the following, let $x := s - |K^1| = |K^2| - \binom{s}{2}$ denote the number of "excess" vertices from K^2. Clearly, there are at most $\binom{s-x}{2}$ vertices in K^2 that have two neighbors in K^1. Hence, the total number of edges in $G_1[K]$ is at most $2 \cdot \binom{s-x}{2} + (x + \binom{s}{2} - \binom{s-x}{2}) = \binom{s-x}{2} + x + \binom{s}{2}$. Simple calculus shows that for all x with $0 \leq x < s - 2$ this number is decreasing with increasing x. In case $x \geq s - 2$, the number of edges in G is clearly at most $\binom{s}{2} + s - 1$. In summary, this shows that no subgraph of G_1 of order $s + \binom{s}{2}$ can have more than $2 \cdot \binom{s}{2}$ edges. It also follows that $2 \cdot \binom{s}{2}$ edges can only be achieved in case $|K^1| \geq s$, and since the number of edges is at most $2 \cdot |K^2|$ this implies $|K^1| = s$. Finally, this means that each pair of vertices in K^1 has a common neighbor in K^2. By construction, K^1 thus is a clique in G.

To conclude our construction, we add a gadget graph as described in Lemma 6: Let $\mu = a/b$ and without loss of generality, let $a \geq 3$. Then, we add the gadget graph H such that it is $2(a-1) \geq 4$-connected, and adding $s + \binom{s}{2}$ vertices and $2\binom{s}{2}$ edges makes H have density exactly μ. Furthermore, H can be constructed in time $\text{poly}(a, b, s + \binom{s}{2}, 2\binom{s}{2})$. Let G^* be the disjoint union of G_1 and H. Set the instance of μ-Clique to $(G^*, s + \binom{s}{2} + |V(H)|)$.

Since G_1 has degeneracy at most two, and the gadget graph H can be constructed in time $\text{poly}(s)$ (observe that we can assume a and b to be constants), the degeneracy of the constructed graph is upper bounded by some polynomial in s. For the correctness of our construction, first observe the following. If G has a clique C on s vertices, then H together with $C \cup S_C$ form a μ-clique in G^*, where $S_C = \{s_{\{u,v\}} : u, v \in C\}$. This follows by the definition of H. For the reverse direction, we prove that we can assume that every μ-clique M in G^* of order at least $s + \binom{s}{2} + |V(H)|$ contains every vertex of H. If this is true, then, it follows that $|M \cap V(G_1)| = s + \binom{s}{2}$, and, hence, $M \cap V(G)$ induces a clique on s vertices in G because of the property of G_1 we have shown above. Assume that M does not contain some i vertices of H. Then, consider the vertices in $S \cap M$. Each of these vertices has degree at most two and the removal of them makes $M \cap V(G_1)$ an independent set. Thus, we may remove the vertices in $S \cap M$ from M, if $|S \cap M| < i$, remove some further vertices in $M \cap V(G_1)$ from M, and add all the missing vertices of H to M. The removal implies losing at most $2i$ edges, but in adding the missing vertices of H we gain at least $2i$ edges, since, by construction, H is 4-connected. Thus, the correctness of the construction follows. □

We can use the reduction behind Theorem 10 to also exclude polynomial-size problem kernels for the parameters maximum degree and h-index.

Theorem 11. *For any fixed μ, $0 < \mu < 1$, μ-Clique does not admit a polynomial-size problem kernel with respect to either maximum degree or h-index unless NP ⊆ coNP/poly.*

Proof. It suffices to prove the statement for the larger maximum degree parameter. For this, we observe that the reduction used in the proof of Theorem 10 implies a cross-composition [7] from Clique into μ-Clique parameterized by maximum degree. A cross-composition from a problem $L \subseteq \Sigma^*$ into a parameterized problem P is an algorithm that, given t strings $x_1, x_2, \ldots, x_t \in \Sigma^*$, computes an instance x^* of P with parameter value k such that its running time is bounded by a polynomial in $\sum_{i=1}^{t} |x_i|$, k is bounded by a polynomial in $\max_{i=1}^{t} |x_i|$, and $x^* \in P$ if and only if $x_i \in L$ for some $1 \leq i \leq t$.[5] If

[5] For readability, we simplified the more general definition of cross-composition here.

a parameterized problem that has a cross-composition from an NP-hard problem also admits a polynomial-size problem kernel, then NP $\subseteq$ coNP/poly [7].

Let several instances of Clique be given and without loss of generality, assume that each instance asks for a clique of order k'.[6] Merge the instances into one instance of Clique by taking the disjoint union of the graphs. It is clear that this graph contains a clique of given order if and only if one of its connected components does. Then, apply the reduction used in Theorem 10 to the resulting graph. To obtain that this procedure is a cross-composition, it remains to show that the maximum degree in the created instance is bounded by a polynomial in the maximum size of the input instances. This follows since the reduction used for Theorem 10 does not merge any connected components and the introduced gadget graph has size polynomial in k'. Thus, there is cross-composition from Clique into μ-Clique parameterized by the maximum degree. □

6. Outlook

There are many possibilities for future research on the computational problems considered in this work. Obviously, it would be interesting to improve the presented algorithms. For the case of Fixed-Cardinality Optimization, a running time of $\Delta^{o(k)} \cdot \text{poly}(n)$ is unlikely: Clique is a special case of this problem and such a running time would imply $n^{o(k)}$-time algorithms for Clique which contradict the ETH [16]. Hence, one could focus on improving the constants in the base of the exponential function here. For μ-Clique, it would be interesting to obtain a running time of $2^{O(\Delta/\mu)} \cdot \text{poly}(n)$ or to show that such a running time is unlikely under the usual complexity-theoretic assumptions. Furthermore, it would be interesting to obtain polynomial-size Turing kernels [6] for μ-Clique and any of the considered parameters. Also, is there a better polynomial-time algorithm for μ-Clique on planar graphs than the trivial brute-force XP-algorithm for degeneracy? Finally, a further restriction that can be made in the area of community detection is to bound the size of the neighborhood of the μ-cliques. Efficient algorithms exploiting such bounds would be interesting and also practically relevant.

Acknowledgment. We thank Mikko Koivisto for pointing out Theorem 1 by Bollobás [8].

References

[1] N. Alon, R. Yuster, and U. Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995.

[2] O. Amini, I. Sau, and S. Saurabh. Parameterized complexity of the smallest degree-constrained subgraph problem. In *Proceedings of the 3rd International Workshop on Parameterized and Exact Computation (IWPEC '08)*, volume 5018 of *Lecture Notes in Computer Science*, pages 13–29. Springer, 2008.

[3] B. Balasundaram and F. M. Pajouh. Graph theoretic clique relaxations and applications. In *Handbook of Combinatorial Optimization*, pages 1559–1598. Springer, 2013.

[4] B. Balasundaram, S. Butenko, and S. Trukhanov. Novel approaches for analyzing biological networks. *Journal of Combinatorial Optimization*, 10(1):23–39, 2005.

[5] B. Balasundaram, S. Butenko, and I. V. Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59(1):133–142, 2011.

[6] If an instance asks for a smaller clique, simply add a new vertex and connect it to all other vertices of this instance.

[6] D. Binkele-Raible, H. Fernau, F. V. Fomin, D. Lokshtanov, S. Saurabh, and Y. Villanger. Kernel(s) for problems with no kernel: On out-trees with many leaves. *ACM Transactions on Algorithms*, 8 (4):38, 2012.

[7] H. L. Bodlaender, B. M. P. Jansen, and S. Kratsch. Kernelization lower bounds by cross-composition. *SIAM Journal on Discrete Mathematics*, 28(1):277–305, 2014.

[8] B. Bollobás. *The Art of Mathematics.* Cambridge University Press, 1st edition, 2006.

[9] E. Bonnet, B. Escoffier, V. T. Paschos, and E. Tourniaire. Multi-parameter analysis for local graph partitioning problems: Using greediness for parameterization. *Algorithmica*, 71(3):566–580, 2015.

[10] N. Bourgeois, A. Giannakos, G. Lucarelli, I. Milis, and V. T. Paschos. Exact and approximation algorithms for densest k-subgraph. In *Proceedings of the 7th International Workshop on Algorithms and Computation (WALCOM '13)*, volume 7748 of *Lecture Notes in Computer Science*, pages 114–125. Springer, 2013.

[11] P. Briggs and L. Torczon. An efficient representation for sparse sets. *ACM Letters on Programming Languages and Systems*, 2(1-4):59–69, 1993.

[12] M. Bruglieri, M. Ehrgott, H. W. Hamacher, and F. Maffioli. An annotated bibliography of combinatorial optimization problems with fixed cardinality constraints. *Discrete Applied Mathematics*, 154 (9):1344–1357, 2006.

[13] L. Cai. Parameterized complexity of cardinality constrained optimization problems. *The Computer Journal*, 51(1):102–121, 2008.

[14] 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)*, volume 4169 of *Lecture Notes in Computer Science*, pages 239–250. Springer, 2006.

[15] M.-S. Chang, L.-H. Chen, L.-J. Hung, P. Rossmanith, and G.-H. Wu. Exact algorithms for problems related to the densest k-set problem. *Information Processing Letters*, 114(9):510–513, 2014.

[16] J. Chen, B. Chor, M. Fellows, X. Huang, D. W. Juedes, I. A. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005.

[17] P. Deshpande, R. Barzilay, and D. R. Karger. Randomized decoding for selection-and-ordering problems. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL '07)*, pages 444–451, 2007.

[18] R. G. Downey and M. R. Fellows. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theoretical Computer Science*, 141(1&2):109–131, 1995.

[19] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Springer, 2012.

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

[21] D. Eppstein, M. Löffler, and D. Strash. Listing all maximal cliques in large sparse real-world graphs. *ACM Journal of Experimental Algorithmics*, 18, 2013.

[22] U. Feige and M. Seltser. On the densest k-subgraph problem. Technical report, The Weizmann Institute, Department of Applied Math and Computer Science, 1997.

[23] U. Feige, D. Peleg, and G. Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

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

[25] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.

[26] J. Guo, I. A. Kanj, C. Komusiewicz, and J. Uhlmann. Editing graphs into disjoint unions of dense clusters. *Algorithmica*, 61(4):949–970, 2011.

[27] F. Harary. The maximum connectivity of a graph. *Proceedings of the National Academy of Science of the United States of America*, 48(7):1142–1146, 1962.

[28] D. Hermelin, C.-C. Huang, S. Kratsch, and M. Wahlström. Parameterized two-player nash equilibrium. *Algorithmica*, 65(4):802–816, 2013.

[29] P. Hilton and J. Pedersen. Catalan numbers, their generalization, and their uses. *The Mathematical Intelligencer*, 13(2):64–75, 1991.

[30] K. Holzapfel, S. Kosub, M. G. Maaß, and H. Täubig. The complexity of detecting fixed-density clusters. *Discrete Applied Mathematics*, 154(11):1547–1562, 2006.

[31] F. Hüffner, S. Wernicke, and T. Zichner. Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica*, 52(2):114–132, 2008.

[32] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and Systems Sciences*, 63(4):512–530, 2001.

[33] J. Katrenic and I. Schiermeyer. Improved approximation bounds for the minimum rainbow subgraph

problem. *Information Processing Letters*, 111(3):110–114, 2011.

[34] S. Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS '04)*, pages 136–145. IEEE Computer Society, 2004.

[35] S. Khuller and B. Saha. On finding dense subgraphs. In *Proceedings of 36th International Colloquium on Automata, Languages and Programming (ICALP '09)*, volume 5555 of *Lecture Notes in Computer Science*, pages 597–608. Springer, 2009.

[36] C. Komusiewicz and M. Sorge. Finding dense subgraphs of sparse graphs. In *Proceedings of the 7th International Symposium on Parameterized and Exact Computation (IPEC '12)*, volume 7535 of *Lecture Notes in Computer Science*, pages 242–251. Springer, 2012.

[37] C. Komusiewicz, M. Sorge, and K. Stahl. Finding connected subgraphs of fixed minimum density: Implementation and experiments. In *Proceedings of the 14th International Symposium on Experimental Algorithms (SEA '15)*, 2015. Accepted for publication.

[38] S. Kosub. Local density. In *Network Analysis*, volume 3418 of *Lecture Notes in Computer Science*, pages 112–142. Springer, 2004.

[39] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[40] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Number 31 in Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.

[41] J. Pattillo, A. Veremyev, S. Butenko, and V. Boginski. On the maximum quasi-clique problem. *Discrete Applied Mathematics*, 161(1-2):244–257, 2013.

[42] H. Robbins. A remark on stirling's formula. *American Mathematical Monthly*, pages 26–29, 1955.

[43] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang. Dense subgraphs with restrictions and applications to gene annotation graphs. In *Proceedings of the 14th Annual International Conference on Research in Computational Molecular Biology (RECOMB '10)*, volume 6044 of *LCNS*, pages 456–472. Springer, 2010.

[44] A. Schäfer, C. Komusiewicz, H. Moser, and R. Niedermeier. Parameterized computational complexity of finding small-diameter subgraphs. *Optimization Letters*, 6(5):883–891, 2012.

[45] H. Shachnai and M. Zehavi. Parameterized algorithms for graph partitioning problems. In *Proceedings of the 40th International Workshop on Graph-Theoretic Concepts in Computer Science (WG '14)*, volume 8747 of *Lecture Notes in Computer Science*, pages 384–395. Springer, 2014.

Appendix A. A Lower Bound on the Number of Connected Subgraphs

For the lower bound, we use the fact that the number of ℓ-ary ordered trees with k inner vertices is exactly $\frac{1}{(\ell-1)k+1}\binom{\ell k}{k}$ (see Hilton and Pedersen [29], for example). In ordered trees the order of the children of a vertex matters. For example, adding two children to the "left" leaf of a vertex with two leaf children yields a different binary ordered tree than adding two children to the "right" leaf. More formally, an ℓ-ary ordered tree is uniquely described by a vector in $\mathbb{N}^\ell$ for each vertex u, such that the number of vertices induced by the subtree of the ith child of u is exactly the ith entry in u's vector and each vertex has either ℓ or zero children.

Lemma 7. *For every pair of integers k and Δ there is a graph G with maximum degree Δ and a vertex v such that there are at least $\frac{1}{(\Delta-2)k+1}\binom{(\Delta-1)k}{k}$ connected subgraphs of order at most k that contain v.*

Proof. Take G to be a tree with root v where every vertex has either zero or $\Delta - 1$ children and the path of each leaf to v is of length at least k. Give the children of every vertex in G an arbitrary order and consider an arbitrary *ordered* rooted tree T with k vertices such that every vertex has at most $\Delta - 1$ children. Observe that T induces a subtree of G by identifying the roots and then embedding the remaining vertices in the natural way. In fact, in this way we obtain a bijection between the subtrees of G of order k and the corresponding ordered trees of order k. Furthermore, by taking an arbitrary ordered tree of order k such that each vertex has at most $\Delta - 1$ children and adding leaves to each vertex which has less than $\Delta - 1$ children we obtain a $(\Delta - 1)$-ary ordered tree with k inner vertices. Vice versa, removing all leaves from a $(\Delta - 1)$-ary ordered tree with k inner vertices we obtain a corresponding tree of order k. Hence, the number of k-vertex subtrees of G is lower-bounded by the number $\frac{1}{(\Delta-2)k+1}\binom{(\Delta-1)k}{k}$ of $(\Delta - 1)$-ary ordered trees with k inner vertices. □

Appendix B. Bounds on a Special Binomial Coefficient

Proposition 3.

$$\frac{1}{e^2}\sqrt{\frac{\ell}{2\pi(\ell-1)k}}\left(\frac{\ell}{\ell-1}\right)^{\ell k}(\ell-1)^k \le \binom{\ell k}{k} \le e\sqrt{\frac{\ell}{2\pi(\ell-1)k}}\left(\frac{\ell}{\ell-1}\right)^{\ell k}(\ell-1)^k.$$

Proof. Both bounds employ Stirling's approximation, we use it in the form

$$\sqrt{2\pi}\ell^{\ell+1/2}e^{-\ell} \le \ell! \le \sqrt{2\pi}\ell^{\ell+1/2}e^{-\ell+1},$$

see Robbins [42]. We get

$$\begin{aligned}\binom{\ell k}{k} &= \frac{(\ell k)!}{((\ell-1)k)!k!} \le \frac{(\ell k)^{\ell k+\frac{1}{2}}e^{-\ell k+1}}{\sqrt{2\pi}((\ell-1)k)^{(\ell-1)k+\frac{1}{2}}e^{-(\ell-1)k}k^{k+\frac{1}{2}}e^{-k}}\\ &= \frac{(\ell k)^{\ell k+\frac{1}{2}}e}{\sqrt{2\pi}((\ell-1)k)^{(\ell-1)k+\frac{1}{2}}k^{k+\frac{1}{2}}} = e\sqrt{\frac{\ell k}{2\pi(\ell-1)k^2}}\cdot\frac{(\ell k)^{\ell k}}{((\ell-1)k)^{(\ell-1)k}k^k},\end{aligned}$$

where the last fraction is equal to

$$\left(\frac{\ell k}{(\ell-1)k}\right)^{\ell k}\cdot\left(\frac{(\ell-1)k}{k}\right)^{k}.$$

Thus

$$\binom{\ell k}{k} \le e\sqrt{\frac{\ell}{2\pi(\ell-1)k}}\left(\frac{\ell}{\ell-1}\right)^{\ell k}(\ell-1)^k.$$

The lower bound is obtained in a similar way, we omit the details. □

Appendix C. Component +-linearity of the Objective Function in Fixed-Cardinality Graph Partitioning Problems

Recall that, in fixed-cardinality graph partitioning problems, $\phi(S) := a|E(S)| + b|\delta(S)|$ where a and b are problem-specific constants and $G(V, E)$ is the input graph. Recall the definitions of $E(S)$ and $\delta(S)$. We extend the definition of δ to $\delta(S, T)$, which denotes the set of edges with one endpoint in S and one in T for disjoint vertex sets S, T. Clearly, for any two disjoint vertex sets S, T, we have

$$\begin{aligned}\phi(S \cup T) &= a|E(S \cup T)| + b|\delta(S \cup T)|\\ &= a(|E(S)| + |E(T)| + |\delta(S,T)|) + b(|\delta(S)| + |\delta(T)| - 2|\delta(S,T)|)\\ &= \phi(S) + \phi(T) + (a - 2b)\cdot|\delta(S,T)|.\end{aligned}$$

Hence, if T is a connected component of $S \cup T$, then $\phi(S \cup T) = \phi(S) + \phi(T)$ and thus ϕ is component sub-+. Further, if $a - 2b \geq 0$, that is, $a \geq 2b$, then ϕ is super-+, showing that ϕ is component +-linear.