Destroying Densest Subgraphs Is Hard

Cristina Bazgan
Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, Paris, France

André Nichterlein
Algorithmics and Computational Complexity, Technische Universität Berlin, Germany

Sofia Vazquez Alferez
Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, Paris, France

Abstract

We analyze the computational complexity of the following computational problems called Bounded-Density Edge Deletion and Bounded-Density Vertex Deletion: Given a graph G, a budget k and a target density τ_ρ, are there k edges (k vertices) whose removal from G results in a graph where the densest subgraph has density at most τ_ρ? Here, the density of a graph is the number of its edges divided by the number of its vertices. We prove that both problems are polynomial-time solvable on trees and cliques but are NP-complete on planar bipartite graphs and split graphs. From a parameterized point of view, we show that both problems are fixed-parameter tractable with respect to the vertex cover number but W[1]-hard with respect to the solution size. Furthermore, we prove that Bounded-Density Edge Deletion is W[1]-hard with respect to the feedback edge number, demonstrating that the problem remains hard on very sparse graphs.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Graph modification problems, NP-hardness, fixed-parameter tractability, W-hardness, special graph classes

Digital Object Identifier 10.4230/LIPIcs.SWAT.2024.6

Related Version *Full Version*: `http://arxiv.org/abs/2404.08599`

Acknowledgements We thank anonymous reviewers of SWAT 2024 for their detailed comments improving the presentation.

1 Introduction

Finding a densest subgraph is a central problem with applications ranging from social network analysis to bioinformatics to finance [21]. There is a rich literature on this topic with the first polynomial-time algorithms given more than 40 years ago [15, 29]. In this work, we study the robustness of densest subgraphs under perturbations of the input graph. More precisely, we study the (parameterized) complexity of the following two computational problems called Bounded-Density Edge Deletion and Bounded-Density Vertex Deletion: Given a graph G, a budget k and a target density τ_ρ, the questions are whether there are k edges, respectively, k vertices, whose removal from G results in a graph where the densest subgraph has density at most τ_ρ? Here, the density $\rho(G)$ of a graph G is defined as the ratio between its number m of edges and number n of vertices, that is, $\rho(G) = m/n$, which is equal to half the average degree of G. Thus, we contribute to the literature on graph modification problems with degree constraints [13, 23, 27]. More broadly, our work fits into parameterized algorithmics on graph modification problems – a line of research with a plethora of results. See Crespelle et al. [6] for a recent survey focusing on edge modification problems.

Denote with $\rho^*(G)$ the density of a densest subgraph of G. Note that cycles have density exactly one and forests a density below one. Thus, it is easy to see that a graph G is a forest if and only if $\rho^*(G) < 1$. Hence, our problems contain the NP-hard Feedback

Table 1 Our results for Bounded-Density Edge Deletion / Vertex Deletion.

	Edge Deletion	Vertex Deletion
Trees	$O(n^3)$ (Theorem 7)	$O(n)$ (Theorem 16)
Cliques	$O(n^2)$ (Theorem 9)	$O(n^2)$ (trivial)
Split	NP-complete (Theorem 13)	NP-complete (Theorem 20)
Planar Bipartite	NP-complete (Theorem 12)	NP-complete (Theorem 17)
Solution Size k	W[1]-hard (Theorem 15)	W[2]-hard (Theorem 23)
Vertex Cover Number	FPT (Theorem 14)	FPT (Theorem 21)
Feedback Edge Number	W[1]-hard (Theorem 15)	?

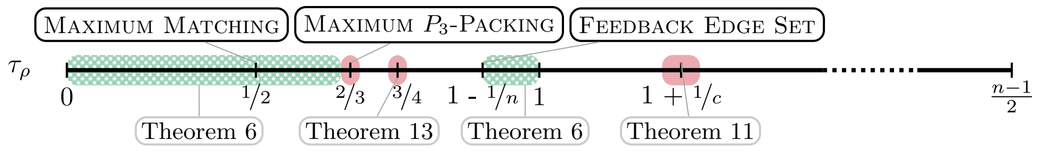

Figure 1 The computational complexity and special cases of Bounded-Density Edge Deletion for specific values of the target density τ_ρ, see Sections 3.1 and 3.2 for the details. Green (hatched) boxes indicate polynomial-time solvable cases while red (solid) boxes denote NP-hard cases. The c in $1 + 1/c$ can be any constant larger than 24. The complexity for larger values of τ_ρ remains open.

Vertex Set and the polynomial-time solvable Feedback Edge Set[1], respectively, as special cases. For target densities smaller than one not only are cycles to be destroyed, but also a bound on the size of the remaining connected components is implied. For example, for $\tau_\rho = 2/3$ ($= 1/2$ or $= 0$) each connected component in the resulting graph can have at most 2 edges (1 edge for $\tau_\rho = 1/2$ or 0 edges for $\tau_\rho = 0$). Consequently, Bounded-Density Vertex Deletion generalizes Dissociation Set ($\tau_\rho = 1/2$) and Vertex Cover ($\tau_\rho = 0$). Bounded-Density Edge Deletion generalizes Maximum Cardinality Matching ($\tau_\rho = 1/2$) and the NP-hard Maximum P_3-packing ($\tau_\rho = 2/3$) where the non-deleted edges form the matching and P_3-packing, respectively.

Our contributions. We refer to Table 1 for an overview of our results. Given the above connections to known computational problems, we start with the seemingly easier of the two problems: Bounded-Density Edge Deletion. We provide polynomial-time algorithms for specific target densities below one (see Figure 1) or when the input is a clique or tree (see Section 3.1). However, beyond these cases, the problem turns out to be surprisingly hard. There are target densities above or below one for which it is NP-hard, see Figure 1 for an overview. We show that Bounded-Density Edge Deletion remains NP-hard on claw-free cubic planar, planar bipartite, and split graphs (see Section 3.2). Moreover, we prove W[1]-hardness with respect to the combined parameter k and feedback edge number. This implies that the problem remains hard even on very sparse graphs as the feedback edge number in a connected graph is $m - n + 1$, despite being polynomial-time solvable on trees. Note that this also implies W[1]-hardness with respect to prominent parameters like treewidth. Moreover, our employed reduction shows W[1]-hardness for T_{h+1}-Free Edge

[1] Given a graph G and an integer k, Feedback Vertex Set (Feedback Edge Set) asks if there is a set of k vertices (k edges) whose removal makes G acyclic.

Deletion[2] with respect to the treewidth, thus answering an open question by Enright and Meeks [10]. On the positive side, using integer linear programming, we classify the problem as fixed-parameter tractable with respect to the vertex cover number (see Section 3.3).

Turning to Bounded-Density Vertex Deletion, we derive NP-hardness for all $\tau_\rho \in [0, n^{1-1/c}]$ for any constant c. Note that the density of a graph is between 0 and $(n-1)/2$ and the case $\tau_\rho = (n-1)/2$ is trivial. Moreover, we show NP-hardness on planar bipartite graphs of maximum degree four, line graphs of planar bipartite graphs, and split graphs (see Section 4.2) as well as a polynomial-time algorithm for trees (see Section 4.1). Furthermore, we prove W[2]-hardness with respect to k and fixed-parameter tractability with respect to the vertex cover number (see Section 4.3). Notably, the latter algorithm is easier than in the edge deletion setting; in particular it does not rely on integer linear programming.

Due to space restrictions the proofs of some statements (marked by $\star$) are omitted.

Further related work. The density as defined above is related to a variety of useful concepts. It belongs to a family of functions of the form $f(G, a, b, c) = a|E(G)|/(b|V(G)| - c)$ with $a, b, c \in \mathbb{Q}$. Depending on the values of a, b and c, the function has been used to study a variety of network properties [17]. For instance $\rho(G) = f(G, 1, 1, 0)$ is used in the study of random graphs [1], whilst $f(G, 1, 1, 1)$ comes up in the study of vulnerability of networks [7], and $f(G, 1, 3, 6)$ is used to study rigid frameworks [20]. A more general class of functions can be studied, where a, b, c are not rational numbers but functions. For instance, Hobbs [16] studies the vulnerability of a graph G by finding the subgraph H of G that maximizes $|E(H)|/(|V(H)| - \omega(H))$ with $\omega(H)$ being the number of connected components in H. The interpretation being that attacking such a subgraph would lead to the maximum number of connected components being created per unit of effort spent on the attack, where the effort is proportional to the number of edges being targeted by the attacker.

The maximum average degree $\mathrm{mad}(G)$ of a graph G is the maximum of the average degrees of all subgraphs of G. Note that $\mathrm{mad}(G) = 2\rho^*(G)$. Recently, Nadara and Smulewicz [25] showed that for every graph G and positive integer k such that $\mathrm{mad}(G) > k$, there exists a polynomial-time algorithm to compute a subset of vertices $S \subseteq V(G)$ such that $\mathrm{mad}(G-S) \le \mathrm{mad}(G) - k$ and every subgraph of $G[S]$ has minimum degree at most $k-1$. Though no guarantees are given that S has minimum size for subsets S that achieve $\mathrm{mad}(G-S) \le \mathrm{mad}(G) - k$.

Modifying a graph to bound its maximum average degree can be of use because of a variety of results on the colorability of graphs with bounded mad. In general, mad can be used to give a bound on the chromatic number $\chi(G)$ of G, as $\chi(G) \le \lfloor \mathrm{mad}(G) \rfloor + 1$ [26]. It is well-known that for any planar graph G, the maximum average degree is related to the girth $g(G)$ of G in the following way: $(\mathrm{mad}(G) - 2)(g(G) - 2) < 4$ [25]. Several results are known for variations of coloring problems and mad [2, 3, 4, 19].

2 Preliminaries

Notation. For $n \in \mathbb{N}$ we set $[n] = \{1, 2, \ldots, n\}$. Let G be a simple, undirected, and unweighted graph. We denote the set of vertices of G by $V(G)$ and the set of edges of G by $E(G)$. We set $n_G = |V(G)|$ and $m_G = |E(G)|$. We denote the degree of a vertex $v \in V(G)$ by $\deg_G(v)$. If the graph is clear from context, then we drop the subscript. The minimum

[2] Given a graph and an integer k, the question is whether k edges can be removed so that no connected component has more than h vertices?

degree of G is denoted by $\delta(G)$, and the maximum degree of G is denoted by $\Delta(G)$. We denote with $H \subseteq G$ that H is a subgraph of G. The *density of* G is $\rho(G) = m/n$. We define the density of the empty graph as zero. We denote by $\rho^*(G)$ the density of the densest subgraph of G, that is, $\rho^*(G) = \max_{H \subseteq G} \rho(H)$. For a subset of vertices $W \subseteq V(G)$, we denote with $G[W]$ the subgraph *induced* by W. For two subsets of vertices $W, U \subseteq V(G)$ we set $E(W, U)$ to be the set of edges with one endpoint in W and another in U.

We denote by P_n the path on n vertices, by K_n the complete graph on n vertices (also called a clique of size n), and by $K_{a,b}$ the complete bipartite graph with a and b the size of its two vertex sets. A graph G is *r-regular* if $\deg(v) = r$ for every vertex $v \in G$. A *perfect P_3-packing of* G is a partition of $V(G)$ into sets $V_1, V_2, \ldots, V_{n/3}$ such that for all $i \in [n/3]$ the graph $G[V_i]$ is isomorphic to P_3.

A graph G is *balanced* if $\rho(G') \leq \rho(G)$ for every subgraph $G' \subseteq G$. Let $\rho'(G) = \frac{m}{n-1}$ for $1 \leq n - 1$ and define ρ' to be 0 for the empty graph and one-vertex graph. A graph G is *strongly balanced* if $\rho'(G') \leq \rho'(G)$ for every subgraph $G' \subseteq G$. Ruciński and Vince [31, page 252] point out that every strongly balanced graph is also balanced, though the converse is not true.

Problem Definitions. The problem definition for the edge deletion variant is as follows (the definition for vertex deletion is analogous):

BOUNDED-DENSITY EDGE DELETION
Input: A graph G, an integer $k \geq 0$ and a rational number $\tau_\rho \geq 0$.
Question: Is there a subset $F \subseteq E(G)$ with $|F| \leq k$ such that $\rho^*(G - F) \leq \tau_\rho$?

There are two natural optimization problems associated to BOUNDED-DENSITY EDGE DELETION which we call MIN DENSITY EDGE DELETION (given k minimize τ_ρ) and MIN EDGE DELETION BOUNDED-DENSITY (given τ_ρ minimize the number of edge deletions k).

We emphasize that all problems for vertex deletion are defined and named analogously.

Useful Observations. We often compare the ratio of vertices to edges in different induced subgraphs. To this end, the following basic result is useful.

▶ **Lemma 1** ($\star$). *$\frac{a}{b} \leq \frac{a+c}{b+d} \iff \frac{a}{b} \leq \frac{c}{d}$ and $\frac{a}{b} = \frac{a+c}{b+d} \iff \frac{a}{b} = \frac{c}{d}$.*

The following is a collection of easy observations that can be obtained with Lemma 1.

▶ **Lemma 2** ($\star$). *Let G^* be a densest subgraph of G with $\rho(G^*) = \rho^*(G)$. Then:*

1. *If G^* is not connected, then each connected component C of G^* has density $\rho^*(G)$.*
2. *If $\rho(G^*) = a/b$ for $a, b \in \mathbb{N}$ and $a < b$, then $a = b - 1$ and G^* is a tree on b vertices or a forest where each tree is on b vertices.*
3. *Any vertex $v \notin V(G^*)$ has at most $\lfloor \rho(G^*) \rfloor$ neighbors in $V(G^*)$.*
4. *The minimum degree in G^* is at least $\lceil \rho(G^*) \rceil$. This is tight for trees.*

Note that Lemma 2 (4.) implies that we can remove vertices with degree less than our desired target density τ_ρ.

▶ **Data Reduction Rule 3.** *Let v be a vertex with degree $\deg(v) < \tau_\rho$. Then delete v.*

The following two observations imply that there are only a polynomial number of "interesting" values for the target density τ_ρ. Thus, if we have an algorithm for the decision problem, then, using binary search, one can solve the optimization problems with little overhead in running time.

▶ **Observation 4.** *The density of a graph G on n vertices can have values between 0 and $(n-1)/2$ in intervals of $1/n$: $\rho(G) \in \{0, 1/n, 2/n, \dots, \binom{n}{2}/n = (n-1)/2\}$.*

▶ **Observation 5** ([15])**.** *The maximum density of a subgraph of G can take only a finite number of values: $\rho^*(G) \in \{m'/n' \mid 0 \leq m' \leq m, 1 \leq n' \leq n\}$. Moreover, the minimum distance between two different possible values of $\rho^*(G)$ is at least $1/(n(n-1))$.*

3 Bounded-Density Edge Deletion

In this section we provide our results for Bounded-Density Edge Deletion, starting with the polynomial-time algorithms, continuing with the NP-hard cases, and finishing with our parameterized results.

3.1 Polynomial-time solvable cases

Specific Density Intervals. We show that if the density τ_ρ falls within one of two intervals, then Min Edge Deletion Bounded-Density boils down to computing maximum matchings or spanning trees.

▶ **Theorem 6.** *Min Edge Deletion Bounded-Density can be solved in time $O(m\sqrt{n})$ if $0 \leq \tau_\rho < 2/3$ and in time $O(n+m)$ if $1 - 1/n \leq \tau_\rho \leq 1$.*

Proof. The proof is by case distinction on τ_ρ.

Note that if $\tau_\rho < 1/2$, then no edge can remain in the graph as a K_2 has density $1/2$. Similarly, if $1/2 \leq \tau_\rho < 2/3$, then no connected component can have more than one edge: Otherwise, the component would contain a P_3 which has density $2/3$. Hence, computing a maximum cardinality matching in time $O(m\sqrt{n})$ [24] and removing all edges not in the matching solves the given instance of Min Edge Deletion Bounded-Density.

The second interval is similar. If $1 - 1/n \leq \tau_\rho < 1$, then the resulting graph cannot have any cycle as a cycle has density 1. Moreover, any tree on at most n vertices has density at most $1 - 1/n$. Thus, in this case Min Edge Deletion Bounded-Density is equivalent to computing a minimum feedback edge set, which can be done in time $O(n+m)$ by e. g. deleting all edges not in a spanning tree.

Lastly, if $\tau_\rho = 1$, then each connected component can have at most one cycle, that is, the resulting graph must be a pseudoforest: Consider a connected component C with ℓ vertices and at least two cycles. Any spanning tree of C contains $\ell - 1$ edges and misses at least one edge per cycle. Hence, C contains at least $\ell + 1$ edges and has, thus, density larger than one. Thus, each connected component in the remaining graph can have at most as many edges as vertices. Hence, a solution to the Min Edge Deletion Bounded-Density instance is to do the following for each connected component: delete all edges not in a spanning tree and reinsert an arbitrary edge. This can be done in $O(n+m)$ time. ◀

Trees. If the input is a tree, then any target threshold $\tau_\rho \geq 1$ makes the problem trivial. Hence, the case $\tau_\rho < 1$ is left. Thus, each tree in the remaining graph can have at most $h = \lfloor 1/(1-\tau_\rho) \rfloor$ many vertices: a tree with $h' > h$ vertices has density $(h'-1)/h' = 1 - 1/h' > 1 - 1/h \geq 1 - (1-\tau_\rho) = \tau_\rho$. Hence, the task is to remove as few edges as possible so that each connected component in the remaining graph is of order at most h. This problem is known as T_{h+1}-Free Edge Deletion and can be solved in $O((wh)^{2w}n)$ time [10], where w is the treewidth. As trees have treewidth one and $h \leq n$ (otherwise the problem is trivial), we get the following.

▶ **Theorem 7.** *On the trees,* Min Edge Deletion Bounded-Density *can be solved in time $O(n^3)$.*

Cliques. The problem is not completely trivial on cliques: While a target threshold τ_ρ indicates an upper bound on the remaining edges (as $\rho(G-F) \leq \tau_\rho$ must hold), the question is whether for all m and n there is a balanced graph G with m edges and n vertices; recall that a graph is balanced if the whole graph is a densest subgraph. Ruciński and Vince [31] showed a slightly stronger statement about strongly balanced graphs. Recall that every strongly balanced graph is also balanced; refer to Section 2 for formal definitions.

▶ **Theorem 8** ([31, Theorem 1])**.** *Let n and m be two integers. If $1 \leq n-1 \leq m \leq \binom{n}{2}$, then there exists a strongly balanced graph with n vertices and m edges.*

The proof of Ruciński and Vince [31] is constructive: A strongly balanced graph (that is also a balanced graph) with m edges and n vertices can be constructed in $O(m)$ time.

▶ **Theorem 9.** *On the complete graph K_n,* Min Edge Deletion Bounded-Density *and* Min Density Edge Deletion *can be solved in time $O(n^2)$.*

Proof. We provide the proof for Min Edge Deletion Bounded-Density. The proof for Min Density Edge Deletion is analogous.

Consider an instance $(G = K_n, \tau_\rho)$ of Min Edge Deletion Bounded-Density. Let $F \subseteq E(G)$ be a solution that our algorithm wants to find. Throughout the proof we assume $\tau_\rho \leq (n-1)/2$ and $n \geq 1$, as otherwise $F = \emptyset$. We consider two cases: $\tau_\rho < 1$ and $\tau_\rho \geq 1$.

Case 1. ($\tau_\rho < 1$): Let $t \leq n$ be the largest integer satisfying $(t-1)/t \leq \tau_\rho$. By Lemma 2 (2.), the resulting graph $K_n - F$ must be a collection of trees on at most t vertices each. Thus, partition the vertices in $\lceil n/t \rceil$ parts of size at most t. For each part keep an arbitrary spanning tree. Then F consists of all non-kept edges, i. e., all edges between the parts and all edges not in the selected spanning trees. Clearly, this can be done in $O(n^2)$ time.

Case 2. ($\tau_\rho \geq 1$): We will construct a strongly balanced graph G' on n vertices with density as close to τ_ρ as possible, as allowed by Observation 4. To this end, let t be the largest number in $\{0, 1/n, \ldots, (n-1)/2\}$ so that $t \leq \tau_\rho$, thus $t = \ell/n$ for some integer $\ell \in \{n, n+1, \ldots, \binom{n}{2}\}$ (as $\tau_\rho \geq 1$).

Now we use Theorem 8 to construct a balanced graph G' with ℓ edges and n vertices, thus $\rho^*(G') \leq \tau_\rho$. We will select $F \subseteq E(G)$ so that $G' = G - F$, that is, F contains all edges not in G' and $|F| = \binom{n}{2} - \ell$. By choice of ℓ we know that $(\ell+1)/n > \tau_\rho$. Hence, removing less than $\binom{n}{2} - \ell$ edges from G means the whole graph has density more than τ_ρ. As building the graph G' takes time $O(\ell)$ the overall running time is $O(n^2)$.

To construct the analogous proof for Min Density Edge Deletion use two cases: $k > \binom{n}{2} - n$ (equivalent of Case 1 above) and $k \leq \binom{n}{2} - n$ (equivalent of Case 2). ◀

3.2 NP-Hardness for special graph classes

Claw-free cubic planar graphs. For our first hardness proof of Bounded-Density Edge Deletion we provide a reduction from Perfect P_3-packing which was proven NP-complete even in claw-free cubic planar graphs [33]. Denote with P_k a path on k vertices. A *perfect P_3-packing* of a given graph G is a partition of G into subgraphs in which each subgraph is isomorphic to P_3. The Perfect P_3-packing problem is defined as follows:

Perfect P_3-packing
Input: A graph G.
Question: Is there a perfect P_3-packing of G?

▶ **Theorem 10.** *Bounded-Density Edge Deletion is* NP*-complete for $\tau_\rho = 2/3$ even on claw-free cubic planar graphs.*

Proof. Given an instance G of perfect P_3-packing where G is a claw-free cubic planar graph, let $\mathcal{I} = (G, k, \tau_\rho)$ be an instance of Bounded-Density Edge Deletion where $k = m - 2n/3$ and $\tau_\rho = 2/3$. We claim that G has a perfect P_3-packing if and only if there is a set $F \subseteq E(G)$ with $\rho^*(G - F) = 2/3$ and $|F| = m - 2n/3$.

"$\Rightarrow$:" Given a perfect P_3-packing of G, we set F to be the set of all edges in G that are not in the P_3-packing. Clearly $\rho^*(G - F) = 2/3$. Additionally, $|F| = m - 2n/3$ since there are $n/3$ paths in a perfect P_3-packing, and each path has two edges.

"$\Leftarrow$:" Consider $F \subseteq E(G)$ of size at most $m - 2n/3$ such that $\rho^*(G - F) \leq 2/3$. Then $|E(G) \setminus F| \geq 2n/3$. Note that $\tau_\rho = 2/3$ implies, by Lemma 2 (2.), that all connected components of $G - F$ must be trees of size at most 3. In other words, all connected components in $G - F$ are singletons, P_2's, or P_3's. Denote by t the number of connected components that are P_3's in $G - F$. Then there are $3t$ vertices and $2t$ edges of $G - F$ which belong to a P_3. Consequently, there are $n - 3t$ vertices and $|E(G) \setminus F| - 2t \geq 2(n - 3t)/3$ edges of $G - F$ which are either singletons or belong to a P_2. This is possible only if $n - 3t = 0$, that is $G - F$ is a perfect P_3-packing of G. ◀

Planar graphs with $\Delta = 3$, target density above 1. We next show that Bounded-Density Edge Deletion remains NP-complete even for $\tau_\rho > 1$. To prove this, we reduce from Vertex Cover on cubic planar graphs, which is known to be NP-complete [14]. Vertex Cover is defined as follows:

Vertex Cover

Input:	A graph G and a positive integer k.
Question:	Is there a vertex cover $C \subseteq V(G)$ of size at most k, that is, for each $\{u, v\} \in E$ at least one of u or v is in C?

▶ **Theorem 11** ($\star$). *Bounded-Density Edge Deletion is* NP*-complete for $\tau_\rho > 1$ even on planar graphs with maximum degree* 3.

Bipartite graphs and split graphs. Finally, we show that Bounded-Density Edge Deletion is NP-hard even on planar bipartite graphs and on split graphs.

▶ **Theorem 12** ($\star$). *Bounded-Density Edge Deletion remains* NP*-complete on planar bipartite graphs.*

▶ **Theorem 13** ($\star$). *Bounded-Density Edge Deletion is* NP*-complete on split graphs with $\tau_\rho = 3/4$.*

3.3 Parameterized Complexity Results

FPT wrt. Vertex Cover Number. The vertex cover number of a graph denotes the size of a smallest vertex cover, i. e., the size of a set of vertices whose removal results in an edge-less graph. Although this parameter is relatively large, we still need to rely on integer linear programming in our next algorithm.

▶ **Theorem 14.** *Bounded-Density Edge Deletion can be solved in $2^{O(\ell 2^{2\ell})} + O(m + n)$ time where ℓ is the vertex cover number.*

Proof. Consider an instance (G, k, τ_ρ) of Bounded-Density Edge Deletion.

We provide an algorithm that first computes a minimum vertex cover $C \subseteq V(G)$, $|C| = \ell$, in $O(2^\ell + n + m)$ time [8]. Then it computes edge deletions within $G[C]$ and incident to $S = V(G) \setminus C$ with an ILP, i. e., computes F. To this end, we divide the vertices in S into at most 2^ℓ classes $I_1, \ldots, I_{2^\ell}$, where two vertices are in the same class if and only if they have the same neighbors. Hence, we can define for a class I_i the neighborhood $N(I_i) = N(v)$ for $v \in I_i$. We denote with $|I_i|$ the number of vertices in the class I_i. The usefulness of these classes hinges on the fact that at least one densest subgraph G' of G is such that it either contains all the vertices of a class I_i or none. This is a consequence of Lemma 1: if removing (adding) a vertex of I_i from a subgraph of G increases its density, then removing (adding) any other vertices from I_i must do the same, as they are twins and non-adjacent.

We say a class I_j is *obtainable* from a class I_i if $N(I_j) \subseteq N(I_i)$, that is, by deleting some edges a vertex from I_i can get into class I_j. Note that I_j is obtainable from I_j. Denote with $\mathrm{ob}(I_i)$ all classes obtainable from I_i, formally, $\mathrm{ob}(I_i) = \{I_j \mid N(I_j) \subseteq N(I_i)\}$. Similarly, we denote with $\mathrm{ob}^{-1}(I_j)$ all classes I_i so that I_j is obtainable from I_i, formally, $\mathrm{ob}^{-1}(I_j) = \{I_i \mid N(I_j) \subseteq N(I_i)\}$. If I_j is obtainable from I_i, then we set $\mathrm{cost}(i \to j) = |N(I_i) \setminus N(I_j)|$ to be the number of edges that need to be deleted from a vertex $v \in I_i$ to make it a vertex in I_j.

We now give our ILP. To handle edges with one endpoint in the independent set S we do the following: For each pair of classes I_i, I_j, we add a variable $x_{i\to j}$ whose purpose is to denote how many vertices from class I_i will end up in class I_j by deleting edges. For convenience, we further have a variable $y_j = \sum_{I_i \in \mathrm{ob}^{-1}(I_j)} x_{i\to j}$ denoting the total number of vertices that end up in a class I_j after deleting edges. To ensure correctness we require that no vertex from class I_i gets transformed into a vertex from a class not obtainable from I_i which we represent with the constraint $\sum_{I_j \in \mathrm{ob}(I_i)} x_{i\to j} = |I_i|$ for all $i \in [2^\ell]$. Moreover, both $x_{i\to j}$ and y_j must be integers for all $i, j \in [2^\ell]$.

To handle edges within C we do the following: For each edge $e \in E(C)$ we create a binary variable z_e, where $z_e = 1$ if e survives and 0 if it gets deleted. Again for convenience, for a subset $C' \subseteq C$ we denote with $m_{C'} = \sum_{e \in E(C')} z_e$ the number of edges within C' that survive. The ILP is as follows:

$$\text{Minimize} \quad \sum_{j=1}^{2^\ell} \sum_{I_i \in \mathrm{ob}(I_j)} \mathrm{cost}(i \to j) \cdot x_{i\to j} + \sum_{e \in E(C)} (1 - z_e) \tag{1}$$

$$\text{such that} \quad \sum_{I_j \in \mathrm{ob}(I_i)} x_{i\to j} = |I_i| \qquad \forall i \in [2^\ell] \tag{2}$$

$$\sum_{I_i \in \mathrm{ob}^{-1}(I_j)} x_{i\to j} = y_j \qquad \forall j \in [2^\ell] \tag{3}$$

$$\sum_{e \in E(C')} z_e = m_{C'} \qquad \forall C' \subseteq C \tag{4}$$

$$m_{C'} + \sum_{i \in \mathcal{I}} y_i |N(I_i) \cap C'| \le \tau_\rho(|C'| + \sum_{i \in \mathcal{I}} y_i) \qquad \forall C' \subseteq C, \forall \mathcal{I} \subseteq [2^\ell] \tag{5}$$

$$x_{i\to j}, y_i \in \{0, 1, 2, \ldots\} \qquad \forall i, j \in [2^\ell] \tag{6}$$

$$z_e \in \{0, 1\} \qquad \forall e \in E(C) \tag{7}$$

To prove correctness it remains to show that the ILP admits a solution such that the objective value is at most k if and only if there exist an edge deletion set F of size at most k such that $\rho^*(G - F) \le \tau_\rho$.

Firstly, suppose there exists a feasible solution to the ILP that achieves an objective value of at most k. Because the vertices in I_i are indistinguishable, the set of variables $\{x_{i\to j} \mid j \in [2^\ell]\}$ uniquely determines the set of edges incident to I_i that need to go into F for each class $I_i, i \in [2^\ell]$. The other edges in F can be directly read off the solution $E(C) \cap F = \{z_e \mid z_e = 0\}$. A densest subgraph G' of G contains some vertices from C and from S. If G' does not contain all vertices from some class I_i, then either adding or removing all vertices from I_i will give a graph G'' that is as least as dense as G' (cf. Lemma 1). Inequality (5) ensures that all such possible subgraphs G'' have density at most τ_ρ. Therefore, this constraint guarantees that all subgraphs of $G - F$ have density at most τ_ρ. In the objective function, $\sum_{e\in E(C)}(1 - z_e)$ counts the number of edges that are deleted from C, whilst $\sum_{j=1}^{2^\ell} \sum_{I_i \in \text{ob}(I_j)} \text{cost}(i \to j) \cdot x_{i\to j}$ counts the number of edges that are deleted from $E(C, S)$. Since the feasible solution achieves an objective value of at most k, we have that $|F| \le k$. Thus, F as built above is the desired edge deletion set of size at most k.

Secondly, consider an edge subset $F \subseteq E(G)$ of size at most k such that $\rho^*(G - F) \le \tau_\rho$. We can assign values to $x_{i\to j}$ by reading the number of edges in F that are incident to each vertex v in class I_i. The first and second constraint will be satisfied by construction. The third constraint will be satisfied because $\rho^*(G - F) \le \tau_\rho$. And since $|F| \le k$ the minimum of the objective function will have value at most k.

The ILP has at most $2^{2\ell}$ variables of type $x_{i\to j}$ and at most $\binom{\ell}{2}$ of type z_e. The linear program can be expressed without variables of types y_j and $m_{C'}$ simply by substituting their expressions into the other constraints. This yields a total of $O(2^{2\ell+1})$ variables. In addition, we have $O(2^\ell \cdot 2^{2^\ell})$ constraints. Since an ILP instance $\mathcal{I}$ on p variables can be solved in time $O(p^{2.5p+o(p)} \cdot |\mathcal{I}|)$ [22, 18, 12] we obtain a total (FPT) running time of $2^{O(\ell 2^{2\ell})} + O(m + n)$ for our algorithm. We note that one can achieve a running time of $\ell^{O(2^{2\ell})} + O(n + m)$ by using a randomized algorithm that solves ILPs on p variables in time $\log(2p)^{O(p)}$[30]. ◀

W[1]-Hardness wrt. feedback edge number and solution size. We next prove that Bounded-Density Edge Deletion is W[1]-hard with respect to the combined parameter solution size and feedback edge number. To this end, we use the construction of Enciso et al. [9], who reduced Multicolored Clique to Equitable Connected Partition. These problems are defined as follows.

Multicolored Clique
Input: An undirected graph G properly colored with ℓ colors.
Question: Does G contain a clique on ℓ vertices?

Equitable Connected Partition
Input: An undirected graph G and a positive integer c.
Question: Is there a partition of G into c equally sized connected components, that is, a partition of $V(G)$ into $V_1, \dots, V_c$ so that each $G[V_i]$ is connected and $||V_i| - |V_j|| \le 1$ for each $i, j \in [c]$?

Note that since G is properly colored any clique in G is multicolored, that is, contains at most one vertex per color.

As a byproduct, we also obtain W[1]-hardness for T_{h+1}-Free Edge Deletion parameterized by the combined parameter feedback edge number and solution size. This confirms a conjecture by Enright and Meeks [10]: T_{h+1}-Free Edge Deletion is indeed W[1]-hard with respect to treewidth.

Connection between problems. Let us briefly discuss the connections between the three problems to see why the construction of Enciso et al. [9] works for all three: Consider a cycle on $3n$ vertices. Requiring $c = 3$ for Equitable Connected Partition guarantees one solution (up to symmetry): delete 3 edges so that 3 paths on n vertices remain. Similarly, requiring $k = 3$ and $h = n$ for T_{h+1}-Free Edge Deletion guarantees the same solution. The same applies for Bounded-Density Edge Deletion with $\tau_\rho = (n-1)/n$ and $k = 3$.

The construction of Enciso et al. [9] will enforce the following: Any connected component in a solution will be a tree. Hence, we can control its maximum size via the density τ_ρ or directly via h. Combining the budget k with the size constraint ensures that the minimum number of vertices per component is equal to the maximum number of vertices per component. Thus, for the constructed graph all three problems will ask essentially for the same solution.

▶ **Theorem 15.** *Bounded-Density Edge Deletion and T_{h+1}-Free Edge Deletion are W[1]-hard with respect to the combined parameter solution size and feedback edge number.*

Proof. We use the same reduction as Enciso et al. [9]. Let (G, ℓ) be an instance of Multicolored Clique, which is W[1]-hard with respect to ℓ [11]. Assume without loss of generality that G contains exactly n/ℓ vertices of each color $i \in [\ell]$. (We can add isolated vertices of the appropriate color to G to meet this demand.) Denote with $\lambda\colon E(G) \to [m]$ a bijection that assigns each edge in G a unique integer from $[m]$. Further, denote with $v_1^i, \ldots, v_{n/\ell}^i$ the vertices of color i in G.

Construct instances (H, k, τ_ρ) of Bounded-Density Edge Deletion and (H, k, h) of T_{h+1}-Free Edge Deletion as follows. Let α and β be the smallest integers satisfying $\beta > 2m + 10$ and $\alpha > \beta \cdot n/\ell + 2m + 10$. Then set $h = \alpha + \beta \cdot n/\ell + m + 1$ which will be the maximum number of vertices any subtree in the resulting graph $H - F$ is allowed to have. To ensure this, set $\tau_\rho = 1 - 1/h$.

The basic building block for H is the so-called *anchor*. For $q \geq 1$, a q-anchor is a vertex adjacent to $q - 1$ many vertices of degree one. The idea is that we cannot afford to cut off single vertices, thus all anchors in the constructed graph stay intact. Hence, a q-anchor acts as a single vertex with weight q (the vertex contributes q to the size of its connected component). We use anchors in the *choice* gadget (see Figure 2 for an illustration): Let $A = \{a_1, \ldots, a_q\}$ be a set of integers so that $1 \leq a_i < a_j$ for all $1 \leq i < j$. An A-*choice* is a path on p vertices $u_1, \ldots, u_p$ where vertex u_i, $i \in [p]$, is the center of an $(a_i - a_{i-1})$-anchor (u_1 is the center of an a_1-anchor). Note that an A-choice has exactly a_p many vertices (a_p is the maximum of A), and that removing the edge $\{u_i, u_{i+1}\}$ splits the A-choice gadget into two connected components with a_i and $a_q - a_i$ vertices respectively. We say we *cut* the A-choice at a_i, $i \in [|A| - 1]$, to indicate the removal of the edge $\{u_i, u_{i+1}\}$. To *connect* two vertices u and v by an A-choice means to merge the first and last anchor of the A-choice gadget with u and v, respectively. Merging a vertex v with an q-anchor means to remove the anchor, add q degree-one vertices adjacent only to v and make v adjacent to all vertices the center of the anchor was adjacent to. In other words, identify v with the center of the anchor and add one vertex only adjacent to v (to ensure the overall number of vertices stays the same). Note that connecting two vertices u and v by an A-choice still leaves $|A| - 1$ many possible cuts of the A-choice. We only connect center vertices of α-anchors by choice gadgets, thus enforcing a cut in each choice gadget.

For each of the ℓ colors in G add $2(\ell - 1)$ many α-anchors to the initially empty graph H. Note that α is sufficiently large to ensure that no two anchors can be in the same connected component, that is, $2\alpha > h$. Denote with N_j^i, P_j^i, $j \in [\ell] \setminus \{i\}$, the center vertices of the

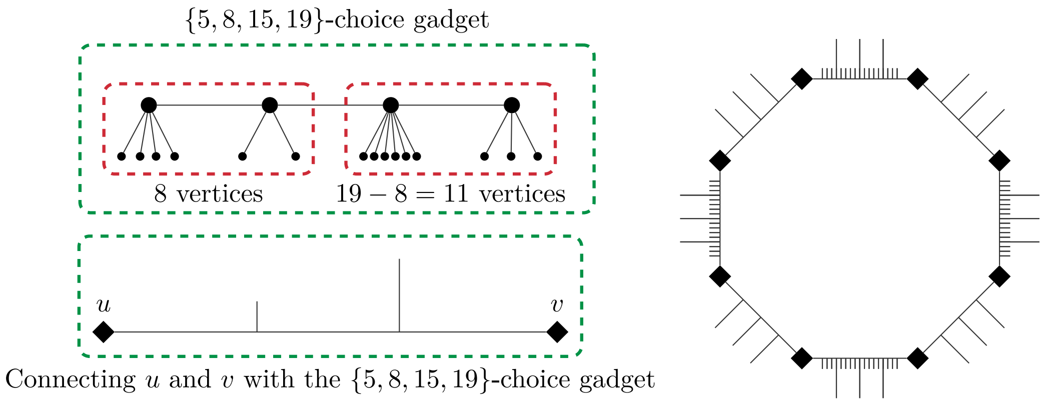
{5, 8, 15, 19}-choice gadget
8 vertices
19 − 8 = 11 vertices
u
v
Connecting u and v with the {5, 8, 15, 19}-choice gadget

Figure 2 *Left above:* A $\{5, 8, 15, 19\}$-choice gadget consisting of four anchors. The two red dashed boxed indicate a possible split into 8 and 11 vertices. *Left below:* A more compact representation (used in further figures) of the same gadget used to connect u and v. The length of the vertical lines correspond to the number of degree-one neighbors of the corresponding anchor; we omit the line for the first and last anchor. The diamond shaped vertices indicate α-anchors. Each choice-gadget in the construction connects two α-anchors as visualized above. *Right:* A gadget constructed for each color. It consists of $2(\ell - 1)$ many α-anchors (so $\ell = 5$ in the example) that are connected in a cycle via choice-gadgets (in the example there are 4 vertices per color).

anchors for color $i \in [\ell]$. Set $A_V = \{1\} \cup \{p\beta \mid p \in [n/\ell]\}$ containing $1 + n/\ell$ elements, thus accommodating one cut for each vertex of color i. Let j' be the "successor index" of j, formally:

$$j' = \begin{cases} 1, & \text{if } (j = \ell \wedge i \neq 1) \vee (j + 1 = i = \ell) \\ 2, & \text{if } j = \ell \wedge 1 = i \\ j + 1, & \text{if } j + 1 \neq i \wedge j + 1 \leq \ell \\ j + 2, & \text{if } j + 1 = i \wedge j + 2 \leq \ell \end{cases}$$

For each $j \in [\ell] \setminus \{i\}$, connect N^i_j and $P^i_{j'}$ by an A_V-choice gadget. Set $A^i_E = \{(p-1)\beta + \lambda(\{u, v^i_p\}) \mid p \in [n/\ell] \wedge u \in V(G) \wedge \{u, v^i_p\} \in E(G)\} \cup \{\beta n/\ell + m + 1\}$ containing one element for each edge incident to each vertex of color i and a large final element to make sure each A^i_E-choice has exactly $\beta n/\ell + m + 1$ many vertices. Next, connect P^i_j and N^i_j by an A^i_E-choice gadget. For $i, j \in [\ell], i \neq j$ connect P^i_j and N^j_i by an $[m + 1]$-choice gadget (leaving one m cut possibilities), see Figure 3 for an illustration and some intuition. Finally, set $k = 2(\ell - 1)\ell + 2\binom{\ell}{2} = 3(\ell - 1)\ell$.

Observe that removing in H the degree-one vertices, we are left with ℓ cycles that are pairwise connected by two paths. Thus, the feedback edge number of the constructed graph H is $\ell + 2\binom{\ell}{2} - (\ell - 1) = 2\binom{\ell}{2} + 1$.

We show that any solution for the instance (H, k, τ_ρ) creates connected components of exactly the same size. Let F be a solution to the constructed Bounded-Density Edge Deletion instance (H, k, τ_ρ), that is, $|F| \leq k$ and the densest subgraph in $G - F$ has density at most τ_ρ. Observe that each "large" α-anchor is in a different connected component in $H - F$ as $1/(1 - \tau_\rho) = h < 2\alpha$. Thus, F contains at least one edge of each choice gadget as each choice gadget connects two α-anchors. Since there are k choice gadgets, it follows that F contains exactly one edge from each choice gadget. Thus, $H - F$ contains exactly $2(\ell - 1)\ell$ many connected components. Note that H consists of $2(\ell - 1)\ell$ many α-anchors, $(\ell - 1)$ many

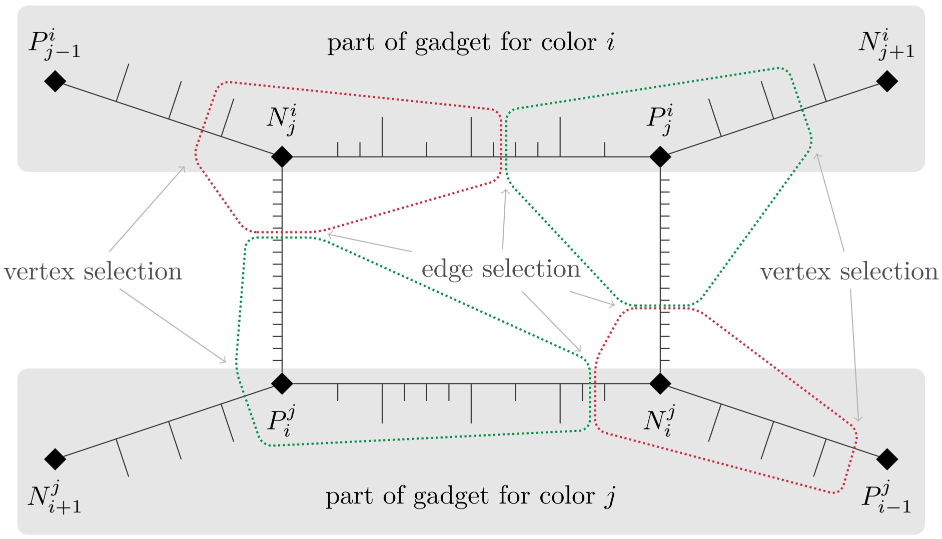

Figure 3 Illustration of the representation of edges of the original graph in the constructed graph. The top part (highlighted in gray) indicates part of the gadget (cycle) created for color i; correspondingly on the bottom for color j. The intuition for the reduction is as follows. Consider a solution for H, that is, a set of edges F so that $\rho^*(H - F) \le \tau_\rho$. Some connected components in $H - F$ are indicated by dashed lines enclosing parts of the vertices. The size constraint for each connected component (enforced by the density threshold $\tau_\rho < 1$) ensures that each connected component of $H - F$ contains at most one α-anchor (indicated by diamond-shaped vertices), at most n/ℓ many β-anchors (indicated by the longer vertical lines in the choice-gadgets, here $n/\ell = 4$), and an additional $m + 1$ vertices. In the vertex-selection part (the A_V-choice gadgets that only contains β-anchors) there are n/ℓ many choices to cut; each corresponding to selecting one of the n/ℓ vertices of the respective color. The bound of at most n/ℓ many β-anchors per connected component enforces the same choice throughout the color gadget (see right side of Figure 2). The edge selection follows the same idea as the vertex selection for each color (the number are just smaller): The $[m+1]$-choice gadgets between N^i_j and P^j_i as well as between P^i_j and N^j_i imply that $2m+2$ vertices need to be distributed to the connected components around $N^i_j, P^j_i, P^i_j, N^j_i$. Moreover, the vertex pairs $\{N^i_j, P^i_j\}$ and $\{N^j_i, P^j_i\}$ are connected via an A^i_E- resp. A^j_E-choice gadget. Both of these gadgets contain $n/\ell + m + 1$ vertices. Thus, $4m + 4$ vertices needs to be distributed to the connected components around $N^i_j, P^j_i, P^i_j, N^j_i$. By construction, this requires a cut at an anchor representing the same edge in all four connections. This implies that the selected vertices in the color gadgets are adjacent in the original graph. As one vertex needs to be selected per color a clique needs to be selected.

A_V-choice and A^i_E-choice gadgets for each $i \in [\ell]$, and $2\binom{\ell}{2}$ many $[m+1]$-choice gadgets. Recall that a A-choice contains exactly $\max_{a \in A}\{a\}$ vertices. Thus, the number of vertices in H is

$$2(\ell-1)\ell\cdot\alpha+(\ell-1)\ell\cdot(2\beta n/\ell+m+1)+(\ell-1)\ell\cdot(m+1) = 2(\ell-1)\ell\cdot(\alpha+\beta n/\ell+m+1) = 2(\ell-1)\ell\cdot h.$$

Hence, by pigeon principle, each connected component in $H - F$ contains exactly h vertices. Thus, F certifies also a solution to the Equitable Connected Partition instance $(H, 2(\ell-1)\ell)$. The correctness of the reduction follows from the arguments of Enciso et al. [9], as they use the same reduction for Equitable Connected Partition. ◀

We remark that the above theorem also implies W[1]-hardness with respect to the treewidth for Bounded-Density Edge Deletion and T_{h+1}-Free Edge Deletion as the treewidth of is upper bounded by twice the feedback edge number. This resolves an open question of Enright and Meeks [10].

4 Bounded-Density Vertex Deletion

In this section we show that Bounded-Density Vertex Deletion is NP-complete on several graph classes. We also prove the W[2]-hardness with respect to the parameter k, the number of vertices to remove.

4.1 Polynomial-time algorithm for trees

In this part we discuss the case of trees and show that the problem is polynomial-time solvable. As in the edge deletion variant, we only need to consider the case that $\tau_\rho < 1$ (see the discussion before Theorem 7). Thus, we need to delete vertices such that each connected component in the resulting graph has at most $h = 1/(1 - \tau_\rho)$ many vertices.

Shen and Smith [32] consider the problem of deleting k vertices to minimize the size of the largest connected component. They established an algorithm in $O(n^3 \log n)$ time for trees. We improve on this as follows.

▶ **Theorem 16.** *Bounded-Density Vertex Deletion can be solved in $O(n)$ time on trees.*

Proof. Let T be the input tree. We propose a simple greedy algorithm that works bottom up from the leaves. In each vertex we visit we store the size (number of vertices) of the current subtree. For a vertex v this can be easily computed by summing up the values in its children and adding one. Whenever a vertex reaches subtree size above h, we delete the vertex. Naturally, when computing the subtree size, then deleted children will be ignored. This algorithm runs in $O(n)$ time.

Clearly, the algorithm provides a graph where each connected component contains at most h vertices. It remains to show that there is no smaller solution. To this end, consider a subtree T_v rooted at vertex v. If T_v contains more than h vertices, then any solution must delete at least one vertex in T_v. If T_v has size less than h, then we claim there is an optimal solution not deleting any vertex in T_v: Assume otherwise and let $S \subseteq V(T)$ be an optimal solution deleting a vertex u in T_v. Then removing u from S and adding the parent from v to S results in another solution of the same cost – a contradiction. Hence, our algorithm produces an optimal solution. ◀

4.2 NP-hardness results

We prove in the following the NP-hardness of Bounded-Density Vertex Deletion by reduction from Feedback Vertex Set, which remains NP-hard on Hamiltonian planar 4-regular graphs [5]. Given an instance of Feedback Vertex Set, that is a graph G, and an integer k, the problem consists in deciding the existence of a subset $V' \subseteq V(G)$ with $|V'| \leq k$ such that the $G - V'$ is cycle-free.

Bounded-Density Vertex Deletion is closely related to Feedback Vertex Set. Given a graph G and a subset $V' \subseteq V(G)$, we have that $G - V'$ is cycle-free if and only if $\rho^*(G - V') < 1$. Thus Bounded-Density Vertex Deletion is NP-complete on all classes of graphs where Feedback Vertex Set is NP-complete, for example on planar and maximum degree 4 graphs. We can even prove a stronger result as follows:

Let G be an undirected graph. The bipartite incidence graph of G (also called subdivision of G) is the bipartite graph H whose vertex set is $V(G) \cup E(G)$ and there is an edge in H between $v \in V(G)$ and $e \in E(G)$ if and only if e is incident to v in G.

▶ **Theorem 17.** *Bounded-Density Vertex Deletion is* NP*-complete for $\tau_\rho < 1$ even for planar bipartite graphs with maximum degree 4.*

Proof. We prove the NP-hardness by reduction from Feedback Vertex Set. Considering a graph G on n vertices and an integer k, instance of Feedback Vertex Set, let H be the bipartite incidence graph of G. Remark that if G is planar of maximum degree 4 then H is still planar with maximum degree 4. We show in the following that G contains a subset $V' \subseteq V(G)$ with $|V'| \leq k$ such that $G - V'$ is cycle-free if and only if H contains a subset $F \subseteq V(G) \cup E(G)$ with $|F| \leq k$ such that the $\rho^*(H - F) \leq 1 - 1/n^2$.

Note that to any cycle $v_1, v_2, \ldots, v_i, v_1$ in G it corresponds a cycle $v_1, v_1v_2, v_2, \ldots, v_i, v_iv_1, v_1$ in H. Moreover, to any cycle from H where the vertices alternate between vertices from $V(G)$ and $E(G)$, there corresponds a cycle in G. Furthermore, a subgraph $G[V']$ is cycle-free if and only if $\rho(G[V']) < 1$.

If G contains a subset $F \subseteq V$ with $|F| \leq k$ such that the $G - F$ is cycle-free, then $H - F$ contains no cycle and thus $\rho^*(H - F) \leq 1 - 1/n^2$.

Consider, now, that H contains a subset $F \subseteq V(G) \cup E(G)$ with $|F| \leq k$ such that $\rho^*(H - F) \leq 1 - 1/n^2$. Any vertex $e \in F \cap E(G)$ from H has two neighbors. We can replace it by one of its neighbors in F. This exchange makes the degree of e less than or equal to 1 in $H - F$, and therefore guarantees that e is not in any cycle. Thus, exchanging e in F for one of its neighbors cannot create any cycles, and moreover $|F| \leq k$. At the end of these exchanges, F contains only vertices from $V(G)$ and moreover $H - F$ contains no cycle. Thus F is a solution for the instance (G, k) of Feedback Vertex Set. ◀

In the next reduction we use the following problem:

Dissociation Set

Input:	A graph G and an integer k
Question:	Is there a subset of vertices $S \subseteq V(G)$ of size at least k such that $G[S]$ has maximum degree at most 1?

Dissociation Set is NP-hard even in line graphs of planar bipartite graphs [28].

▶ **Theorem 18.** *Bounded-Density Vertex Deletion is* NP*-complete for $\tau_\rho = 1/2$ even for line graphs of planar bipartite graphs.*

Proof. Given an instance (G, k') of Dissociation Set on planar line graphs of planar bipartite graphs we construct the instance $(G, k = n - k', \tau_\rho = 1/2)$ of Bounded-Density Vertex Deletion. We can easily see that G has a solution S of size at least k' if and only if $\rho^*(G[S]) \leq 1/2$ as $G[S]$ has density at most 1/2 and $V(G) \backslash S$ has size at most k. ◀

The two previous results show hardness for small values of τ_ρ. Next we show that Bounded-Density Vertex Deletion remains NP-hard for most values of τ_ρ.

▶ **Theorem 19** ($\star$). *Bounded-Density Vertex Deletion is* NP*-complete for any rational τ_ρ such that $0 \leq \tau_\rho \leq n^{1-1/c}$, where c is any constant.*

The last result in this subsection concerns split graphs. The following hardness result holds for large (non-constant) values of τ_ρ.

▶ **Theorem 20** ($\star$). *Bounded-Density Vertex Deletion remains NP-hard on split graphs.*

4.3 Parameterized complexity results

We show that Bounded-Density Vertex Deletion is FPT with respect to vertex cover number and W[2]-hard with respect to k, the number of vertices to remove.

▶ **Theorem 21.** *Bounded-Density Vertex Deletion can be solved in time $2^{O(\ell^2)}n^{O(1)}$ where ℓ is the vertex cover number.*

Proof. Let (G, k, τ_ρ) be an instance of Bounded-Density Vertex Deletion where G has vertex cover number ℓ. One can find a minimum vertex cover of size ℓ in time $O(2^\ell + n + m)$ [8]. Denote by C the set of vertices that belongs to the minimum vertex cover, and by $S = V(G) - C$ the set of vertices in the independent set. We divide the vertices in S into at most 2^ℓ classes $I_1, \dots, I_{2^\ell}$, where two vertices $v_1, v_2 \in S$ are in the same class I_i if they have the same neighbors in C.

Notice that for $k \geq \ell$ we always have a yes-instance, as deleting the ℓ vertices in the vertex cover yields a graph with density 0. Thus, we are interested in the case where we delete at most $\ell - 1$ vertices, that is $k \leq \ell - 1$.

The vertices in each I_i (with $i = 1, \dots, 2^\ell$) are all indistinguishable from each other, and we need to delete between 0 and $\ell - 1$ vertices S. Thus we need to check at most $(\ell 2^\ell)^\ell$ sets of vertices from S as candidates for deletion. For vertices in C we check all 2^ℓ subsets of vertices from C as candidates for deletion. In total, we check at most $2^\ell(\ell 2^\ell)^\ell$ subsets as candidates for deletion, yielding a running time of $2^{\ell^2+\ell\log\ell+\ell}n^{O(1)}$. ◀

▶ Remark 22. Note that if $\tau_\rho \geq \ell$, then nothing needs to be deleted: On the one hand, any vertex from the independent set has degree at most $\ell \leq \tau_\rho$. By Lemma 2 (4.) no vertex of the independent set belongs to a subgraph of density greater than τ_ρ. On the other hand, the vertex cover has density at most $(\ell - 1)/2 < \tau_\rho$.

The following hardness result holds for large (non-constant) values of τ_ρ.

▶ **Theorem 23** ($\star$). *Bounded-Density Vertex Deletion is W[2]-hard with respect to the number k of vertices to remove.*

5 Conclusion

Our work provides a first (parameterized) analysis of Bounded-Density Edge Deletion and Bounded-Density Vertex Deletion. Both problems turn out to be NP-hard even in restricted cases but polynomial-time solvable on trees and cliques. While the W[1]-hardness for Bounded-Density Edge Deletion with respect to the feedback edge number rules out fixed-parameter tractability with respect to many other parameterizations, the respective reduction relies on a very specific target density. In fact, Figure 1 seems to indicate that the computational complexity of Bounded-Density Edge Deletion might change when altering the target density τ_ρ a bit. Does this alternating pattern between tractable and intractable target density extend beyond target density 2? For example the polynomial-time solvable f-Factor problem might be helpful in designing polynomial-time algorithms for Bounded-Density Edge Deletion with $\tau_\rho \in \mathbb{N}$ (any $2\tau_\rho$-regular subgraph would be a valid resulting graph $G - F$). Could such behavior be exploited in approximation algorithms?

We leave also several open questions concerning the parameterized complexity of these problems. Is Bounded-Density Vertex Deletion fixed-parameter tractable with respect to the treewidth (plus the solution size)? What is the parameterized complexity with respect to parameters that are smaller than the vertex cover number, e. g., vertex integrity or twin cover number?

References

1 Béla Bollobás. *Random Graphs*, pages 215–252. Springer New York, New York, NY, 1998. doi:10.1007/978-1-4612-0619-4_7.

2 Marthe Bonamy, Benjamin Lévêque, and Alexandre Pinlou. Graphs with maximum degree $\Delta \geq 17$ and maximum average degree less than 3 are list 2-distance $(\Delta + 2)$-colorable. *Discrete Mathematics*, 317:19–32, 2014.

3 Oleg V Borodin, Alexandr Kostochka, and Matthew Yancey. On 1-improper 2-coloring of sparse graphs. *Discrete Mathematics*, 313(22):2638–2649, 2013.

4 Oleg V Borodin and Alexandr V Kostochka. Vertex decompositions of sparse graphs into an independent vertex set and a subgraph of maximum degree at most 1. *Siberian mathematical journal*, 52(5):796–801, 2011.

5 Dario Cavallaro and Till Fluschnik. Feedback vertex set on hamiltonian graphs. In *47th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2021)*, volume 12911 of *Lecture Notes in Computer Science*, pages 207–218. Springer, 2021. doi:10.1007/978-3-030-86838-3_16.

6 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *Comput. Sci. Rev.*, 48:100556, 2023. doi:10.1016/J.COSREV.2023.100556.

7 William H. Cunningham. Optimal attack and reinforcement of a network. *J. Assoc. Comput. Mach.*, 32(3):549–561, 1985.

8 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

9 Rosa Enciso, Michael R. Fellows, Jiong Guo, Iyad A. Kanj, Frances A. Rosamond, and Ondrej Suchý. What makes equitable connected partition easy. In *In Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC 2009)*, volume 5917 of *LNCS*, pages 122–133. Springer, 2009.

10 Jessica Enright and Kitty Meeks. Deleting edges to restrict the size of an epidemic: a new application for treewidth. *Algorithmica*, 80:1857–1889, 2018.

11 Michael R. Fellows, Danny Hermelin, Frances A. Rosamond, and Stéphane Vialette. On the parameterized complexity of multiple-interval graph problems. *Theor. Comput. Sci.*, 410(1):53–61, 2009. doi:10.1016/J.TCS.2008.09.065.

12 András Frank and Éva Tardos. An application of simultaneous Diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987. doi:10.1007/BF02579200.

13 Vincent Froese, André Nichterlein, and Rolf Niedermeier. Win-win kernelization for degree sequence completion problems. *J. Comput. Syst. Sci.*, 82(6):1100–1111, 2016. doi:10.1016/J.JCSS.2016.03.009.

14 M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete problems. In *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing*, STOC '74, pages 47–63, New York, NY, USA, 1974. Association for Computing Machinery.

15 A. V. Goldberg. Finding a maximum density subgraph. Technical report, University of California at Berkeley, USA, 1984.

16 Arthur M. Hobbs. Network survivability. In *Applications of discrete mathematics*, pages 332–353. McGraw-Hill, New York, 1991.

17 Lavanya Kannan, Arthur Hobbs, Hong-Jian Lai, and Hongyuan Lai. Transforming a graph into a 1-balanced graph. *Discrete Appl. Math.*, 157(2):300–308, 2009.

18 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Math. Oper. Res.*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415.

19 Michael Kopreski and Gexin Yu. Maximum average degree and relaxed coloring. *Discrete Mathematics*, 340(10):2528–2530, 2017.

20 G. Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970.

21 Tommaso Lanciano, Atsushi Miyauchi, Adriano Fazzone, and Francesco Bonchi. A survey on the densest subgraph problem and its variants. *CoRR*, abs/2303.14467, 2023. doi:10.48550/arXiv.2303.14467.

22 H. W. Lenstra. Integer Programming with a Fixed Number of Variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. URL: http://www.jstor.org/stable/3689168.

23 Luke Mathieson and Stefan Szeider. Editing graphs to satisfy degree constraints: A parameterized approach. *J. Comput. Syst. Sci.*, 78(1):179–191, 2012. doi:10.1016/J.JCSS.2011.02.001.

24 Silvio Micali and Vijay V. Vazirani. An o(sqrt(|v|)|e|) algoithm for finding maximum matching in general graphs. In *21st Annual Symposium on Foundations of Computer Science (sfcs 1980)*, pages 17–27, 1980. doi:10.1109/SFCS.1980.12.

25 Wojciech Nadara and Marcin Smulewicz. Decreasing the Maximum Average Degree by Deleting an Independent Set or a d-Degenerate Subgraph. *Electron. J. Comb.*, 29(1), 2022.

26 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Prolegomena*, pages 21–60. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

27 André Nichterlein. *Degree-constrained editing of small-degree graphs*. PhD thesis, Berlin Institute of Technology, 2015. URL: https://opus4.kobv.de/opus4-tuberlin/frontdoor/index/index/docId/6520.

28 Yury Orlovich, Alexandre Dolgui, Gerd Finke, Valery Gordon, and Frank Werner. The complexity of dissociation set problems in graphs. *Discrete Appl. Math.*, 159(13):1352–1366, 2011.

29 Jean-Claude Picard and Maurice Queyranne. A network flow solution to some nonlinear 0-1 programming problems, with applications to graph theory. *Networks*, 12(2):141–159, 1982. doi:10.1002/NET.3230120206.

30 Victor Reis and Thomas Rothvoss. The subspace flatness conjecture and faster integer programming. In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 974–988, 2023. doi:10.1109/FOCS57990.2023.00060.

31 Andrzej Ruciński and Andrew Vince. Strongly balanced graphs and random graphs. *Journal of graph theory*, 10(2):251–264, 1986.

32 Siqian Shen and J Cole Smith. Polynomial-time algorithms for solving a class of critical node problems on trees and series-parallel graphs. *Networks*, 60(2):103–119, 2012.

33 Wenying Xi and Wensong Lin. On maximum p3-packing in claw-free subcubic graphs. *J. Comb. Optim.*, 41(3):694–709, April 2021.