On the Parameterized Complexity of Bounded-Density Vertex Deletion

Jakob Raupach
Technische Universität Berlin, Germany

Tom-Lukas Breitkopf
Algorithmics and Computational Complexity, Technische Universität Berlin, Germany

Anton Herrmann
Algorithmics and Computational Complexity, Technische Universität Berlin, Germany

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

Abstract

We explore the parameterized complexity of Bounded Density Vertex Deletion (BDVD): given a graph G, an integer budget k, and a target density τ_ρ, the task is to determine whether the density (i.e. number of edges divided by number of vertices) of the densest subgraph of G can be reduced to at most τ_ρ by deleting at most k vertices. Our primary focus is on structural graph parameters related to treewidth, as the parameterized complexity of BDVD with respect to treewidth was left as open question by Bazgan et al. [JCSS, 2025]. We resolve this question by showing W[1]-hardness with respect to various parameters, including treedepth and feedback vertex number. These results imply W[1]-hardness with respect to treewidth. We obtain positive results for parameters larger than treedepth and feedback vertex number, namely we show BDVD is in FPT parameterized by the max leaf number or vertex integrity. Under the assumption that the target density τ_ρ is a fixed constant the parameterized complexity landscape of BDVD changes drastically, allowing a fixed-parameter tractable algorithm even for parameters smaller than treewidth, namely cliquewidth. Altogether, our results provide a refined complexity landscape for Bounded Density Vertex Deletion, sharply distinguishing between tractable and intractable parameter regimes under structural parameterizations.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Problems, reductions and completeness

Keywords and phrases Graph Modification Problem, Integer Linear Programming, Dynamic Programming, Max Leaf Number, Vertex Integrity

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.67

Related Version *Full Version*: `https://arxiv.org/abs/2606.26012` [31]

1 Introduction

The problem of identifying a densest subgraph constitutes a central topic in graph theory, with numerous applications across domains such as social network analysis, bioinformatics, and financial modeling [27]. This problem has been extensively studied, with the first polynomial-time algorithms proposed more than four decades ago [30]. In this work, we examine the stability of densest subgraphs under vertex deletions in the input graph.

We investigate Bounded Density Vertex Deletion (BDVD), which asks whether it is possible to delete at most k vertices so that the density of the densest subgraph falls to at most a specified threshold τ_ρ (see Figure 1 for an example). Here, the density of a graph is defined as the ratio of the number of edges to the number of vertices. This problem was first introduced by Bazgan et al. [3] and observed to generalize several classical vertex deletion problems: namely, Vertex Cover for $\tau_\rho = 0$, Dissociation Set for $\tau_\rho = 1/2$,

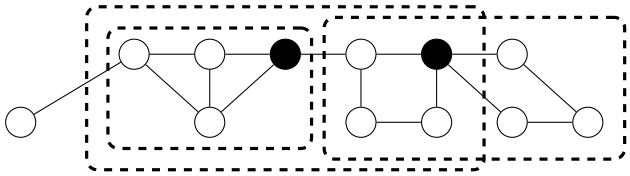

Figure 1 Example graph for BDVD with $k = 2$ and $\tau_\rho = 1$. The remaining graph may only consist of connected components, each containing at most one cycle. Several subgraphs whose density exceeds τ_ρ are indicated, together with a possible solution (shown as filled vertices). Note that at least one vertex of each indicated subgraph must be deleted in a solution.

and Feedback Vertex Set for $\tau_\rho = 1 - 1/n$ (where n is the number of vertices in the graph). Moreover, for integral target densities, BDVD actually coincides with d-Orientable Deletion [22]; this follows from a characterization of graph density by Charikar [8].

Densest subgraph problems have been studied extensively since the 1970s and have experienced renewed attention in recent years [27]. Early algorithmic approaches were based on flow techniques [27, 30, 21]. The aforementioned characterization for graph density by Charikar [8] is based on the dual of a linear programming approach for computing the density of a densest subgraph. This led to the notion of fractional orientations, which have since become a common tool in the study of dense subgraphs [4, 7].

An *orientation* of an undirected graph G assigns a direction to each edge. A graph is called *d-orientable*, for $d \in \mathbb{N}$, if there is an orientation resulting in a digraph with maximum indegree at most d[1]. The task in d-Orientable Deletion is to remove a minimum number of vertices so that the resulting graph becomes d-orientable. A *fractional orientation* assigns parts of an edge to each endpoint. More precisely, a fractional orientation is a function ϕ assigning two non-negative rationals $\phi(e)^u$ and $\phi(e)^v$ with $\phi(e)^u + \phi(e)^v = 1$ to each edge $e := \{u, v\} \in E(G)$. The characterization of Charikar [8] states that a densest subgraph has density τ_ρ if and only if the graph admits a fractional orientation with maximum indegree τ_ρ. Moreover, as observed by Bentert et al. [4] and Chandrasekaran et al. [7], if $\tau_\rho \in \mathbb{N}$, then there also exists an integral orientation[2] with maximum indegree τ_ρ (see Section 2.1 for more details). Hence, d-Orientable Deletion is equivalent to BDVD with $\tau_\rho = d$.

Computing an integral or fractional orientation minimizing the indegree can be done in polynomial-time via a simple reduction to flow [4]. Introduced by Asahiro et al. [1], d-Orientable Deletion turns out to be a computationally challenging problem: It is W[2]-hard with respect to the number of vertices to remove, W[1]-hard with respect to the cliquewidth of the input graph and not approximable within a better factor than $\ln n$ [22]. Algorithmic results include fixed-parameter tractability for treewidth and for the combined parameter cliquewidth plus d [5, 22]. Asahiro et al. [2] provided a $O(\log n)$-approximation for d-Orientable Deletion. Chandrasekaran et al. [7] gave the same approximation ratio for problems generalizing BDVD. All hardness results directly carry over to BDVD.

The classical computational complexity of BDVD was studied by Bazgan et al. [3]. They showed that the problem is solvable in polynomial time on trees, but is NP-hard on planar bipartite graphs of maximum degree 4. Unaware of the connection to d-Orientable Deletion, they reprove the W[2]-hardness of BDVD with respect to the number of vertices to be deleted. They also show fixed-parameter tractability when parameterized by the vertex

[1] In parts of the literature [1, 5], the bound d is for the maximum outdegree. To keep consistency with the characterization by Charikar [8], we bound the indegree, which is completely symmetrical [22].
[2] To avoid ambiguity between *orientation* and *fractional orientation*, we call the former *integral orientation*.

cover number. In the same work, Bazgan et al. considered the sister problem Bounded Density Edge Deletion (deleting edges instead of vertices; thereby generalizing the polynomial-time solvable Maximum Cardinality Matching for $\tau_\rho = 1/2$ and Feedback Edge Set for $\tau_\rho = 1 - 1/n$) and proved that it is W[1]-hard when parameterized by the solution size combined with the feedback edge number and pathwidth. Since the feedback edge number upper-bounds the treewidth, this result also implies W[1]-hardness with respect to the combined parameter treewidth and solution size. Whether an analogous hardness result holds for BDVD remained open.

Heavily relying on fractional orientations, Bentert et al. [4] showed that Bounded Density Edge Deletion is polynomial-time solvable whenever the target density is a multiple of $1/2$ or strictly less than $2/3$, while it is NP-hard for all other values. They also proved that Bounded Density Edge Deletion is fixed-parameter tractable for the parameter treewidth when the target density is fixed via a reduction to General Factors.

Our results. All the listed special cases of BDVD mentioned above are fixed-parameter tractable with respect to the treewidth; whether this is also true for BDVD in general was posed as open question by Bazgan et al. [3]. As can already be seen at the special case d-Orientable Deletion, BDVD has some aspects of a number problem: After deleting vertices, the edges have to be distributed in a balanced way to their endpoints. If d becomes large, then this number problem aspect becomes quite challenging: The fixed-parameter tractability of d-Orientable Deletion with respect to cliquewidth and d is based on a dynamic program which stores partial solutions for each maximum indegree $d' \leq d$ [22]. The W[1]-hardness with respect to cliquewidth alone [22] shows that this approach of dealing with the inherent number problem can presumably not be improved. In Section 2.1, we extend this dynamic program to BDVD for target density $\tau_\rho = p/q$ with $p, q \in \mathbb{N}$, resulting in a running time of $2^{O(p^7 \log q + \mathrm{cw} \cdot p \log p)} n^{O(1)}$, where cw is the cliquewidth of the input graph.

While structural observations (any graph of treewidth tw is easily tw-orientable) allows to deduce fixed-parameter tractability for d-Orientable Deletion, this turns out to be insufficient for BDVD: We provide a dynamic program running in time $(\mathrm{tw} \cdot q)^{O(\mathrm{tw}^2)} \cdot n^{O(1)}$, where tw is the treewidth of the input graph. In Section 3, we establish that the dependency on q cannot be avoided: we show that BDVD is W[1]-hard when parameterized by treewidth, and larger parameters like treedepth and feedback vertex set. For unrestricted target densities, we also prove in Section 2 fixed-parameter tractability with respect to the even larger parameters vertex integrity and maximum leaf number, respectively. An overview of our results is shown in Figure 2.

Graph theory & problem definition. For a graph G, we write $V(G)$ and $E(G)$ to denote the set of vertices and edges of G, respectively. For an edge $\{u, v\} \in E(G)$ we sometimes use uv as a shorthand. We say H is a *subgraph* of G, denoted $H \subseteq G$, if H is a graph with $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$. Further for any set of vertices $V' \subseteq V(G)$ we denote by $G[V']$ the *induced* subgraph of G on V'. For any set of vertices $V' \subseteq V(G)$ we denote by $G - V' \coloneqq G[V(G) \setminus V']$ the graph obtained from G by deleting the vertices V' and all incident edges. If V' consists only of one vertex v, then we also write $G - v$ instead of $G - \{v\}$. We denote the degree of a vertex $v \in V(G)$ by $\deg_G(v) = |N_G(v)|$. If the graph is clear from the context, then we omit G. A path P in G is a sequence $(v_1, \ldots, v_n)$ of pairwise different vertices such that $v_i v_{i+1} \in E(G)$ for all $i \in [n-1] \coloneqq \{1, \ldots, n-1\}$. The vertices v_1 and v_n are called the endpoints of P. The complete graph (or *clique*) on $t \in \mathbb{N}$ vertices is denoted by K_t, and the complete bipartite graph (or *biclique*) with parts of sizes $s, t \in \mathbb{N}$ is denoted

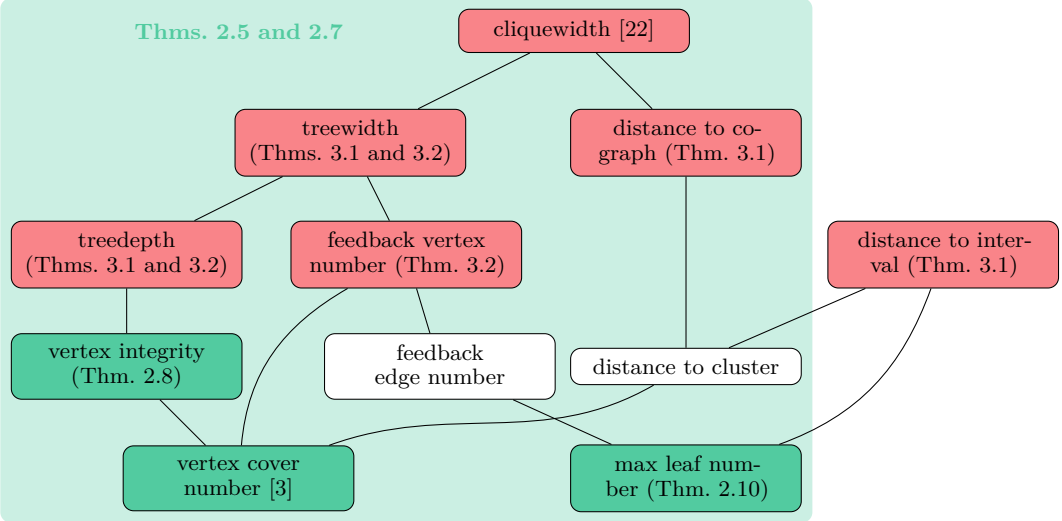

Figure 2 Hasse diagram of parameterized complexity of BDVD with parameters growing from top to bottom. Green indicates fixed-parameter tractability while red indicates W[1]-hardness. Hardness even holds for the combined parameters treedepth + feedback vertex number and treedepth + distance to cograph + distance to interval graph. Indicated with a light green background are all the parameters for which BDVD becomes fixed-parameter tractable if the target density τ_ρ is constant.

by $K_{s,t}$. We define the *density* $\rho(G)$ of a non-empty graph G with $\rho(G) \coloneqq |E(G)|/|V(G)|$ and the density of the empty graph as 0. We also define the *density of the densest subgraph* of a graph G as $\rho^*(G) \coloneqq \max_{H \subseteq G} \rho(H)$. A subgraph $H \subseteq G$ attaining this maximum is called a *densest subgraph* of G.

The problem definition of BDVD as introduced by Bazgan et al. [3] is as follows:

Bounded Density Vertex Deletion
Input: A graph G, an integer $k \in \mathbb{N}$ and a rational number $\tau_\rho \geq 0$.
Question: Is there a subset $S \subseteq V(G)$ with $|S| \leq k$ such that $\rho^*(G - S) \leq \tau_\rho$?

Parameterized complexity. Parameterized complexity provides a multivariate approach for measuring the time complexity of a computational problem [12, 13]. An instance (x, k) of a parameterized problem consists of a classical instance x together with a number k, called the parameter. A parameterized problem is *fixed-parameter tractable* (FPT) if it can be decided in time $f(k) \cdot |x|^{O(1)}$ for some computable function f. If for some $t \geq 1$ a parameterized problem is W[t]-hard with respect to parameter k, then it is considered unlikely to be fixed-parameter tractable when parameterized by k.

2 Algorithmic Results

We first consider the case in which $p, q \in \mathbb{N}$ are part of the parameter indicating the "encoding size" of $\tau_\rho = p/q$. We show fixed-parameter tractability with respect to the combined parameter cliquewidth + $p + q$ by means of dynamic programming. As indicated in Figure 2 this immediately implies fixed-parameter tractability for most other parameters considered in this work when combined with p and q. As the larger parameter treewidth bounds the cliquewidth exponentially, however, the running time of the dynamic program becomes double-exponential in the parameter. We therefore provide another dynamic program for the parameter treewidth combined with p and q with a single-exponential running time.

As will be established in Section 3, BDVD becomes much harder if $p + q$ is not part of the parameter. For this setting we therefore turn to larger parameters. Specifically, we show fixed-parameter tractability for the parameters vertex integrity, thereby strengthening a previous result for vertex cover number [3], and max leaf number. As can be seen in Figure 2, this draws a somewhat close line to the border of tractability established later. Both algorithmic approaches use some initial branching to simplify the instance and then use Integer Linear Programming to tackle the number problems that remain.

2.1 Cliquewidth and Target Density

Here, we show that Bounded Density Vertex Deletion is fixed-parameter tractable parameterized by cliquewidth, p, and q where $p, q \in \mathbb{N}$ with $\tau_\rho = {}^{p}/_{q}$. Recall that the cliquewidth cw of a graph G is the smallest number of labels such that G can inductively be obtained as a labeled graph from the following four operations [10].

- **i)** Introduce $\circ_i$: The graph consisting of a single vertex with label i.
- **ii)** Union $G_1 \oplus G_2$: Creates the disjoint union of the labeled graphs G_1 and G_2.
- **iii)** Relabel $\rho(i_1, i_2)$: Relabels all vertices of label i_1 with the label i_2.
- **iv)** Join $\eta(i_1, i_2)$: All edges between the vertices of the labels i_1 and i_2 are added.

An *ℓ-expression* T is a rooted binary tree in which every node is associated with one of the four operations, ℓ labels are used and the following properties hold.

- Each leaf is associated with an introduce operation.
- Each node with two children is associated with an union operation.
- Each node with exactly one child is associated with a relabel or join operation.

If G is obtained as a labeled graph at the root, then we call T an *ℓ-expression* of G. The subgraph of G constructed up to node t of T is denoted by G_t. Given a graph G of cliquewidth cw, one can compute a $(2^{\mathrm{cw}+1} - 1)$-expression of G in FPT time with respect to cw [26].

In the following it is helpful to think about fractional orientations instead of subgraph densities. The question then is whether we can delete k vertices from G such that there exists a fractional orientation with maximum indegree at most ${}^{p}/_{q}$? Formally a fractional orientation is a function ϕ assigning two non-negative rationals $\phi(e)^u$ and $\phi(e)^v$ with $\phi(e)^u + \phi(e)^v = 1$ to each edge $e := \{u, v\} \in E(G)$ of graph G. We denote with

$$\deg^-_\phi(v) := \sum_{u \in N(v)} \phi(\{u, v\})^v \qquad \text{and} \qquad \Delta^-_\phi := \max_{v \in V(G)} \deg^-_\phi(v)$$

the *indegree* of a vertex $v \in V(G)$ and the *maximum indegree* of a graph G with respect to ϕ. The connection between fractional orientations and the density of the densest subgraph is formalized by the following lemma.

▶ **Lemma 2.1** (Charikar [8]). *For every graph G, there exists a fractional orientation ϕ such that $\rho^*(G) = \Delta^-_\phi(G)$.*

In this context, a fixed target density allows us to restrict ourselves to the existence of *q-quantized fractional orientations* which we define next.

▶ **Definition 2.2.** *Let G be a graph and $q \in \mathbb{N}$. A fractional orientation ϕ is called* q-quantized *if for every edge $e = \{u, v\} \in E(G)$ the following holds: $\phi(e)^u = {}^{p_u}/_{q}$ and $\phi(e)^v = {}^{p_v}/_{q}$ for $p_u, p_v \in \mathbb{N}$.*

▶ **Observation 2.3.** *Let G be a graph and ϕ a q-quantized fractional orientation. Then, for all $v \in V(G)$ we have $\deg^-_\phi(v) = {}^{p_v}/_{q}$ for some $p_v \in \mathbb{N}$.*

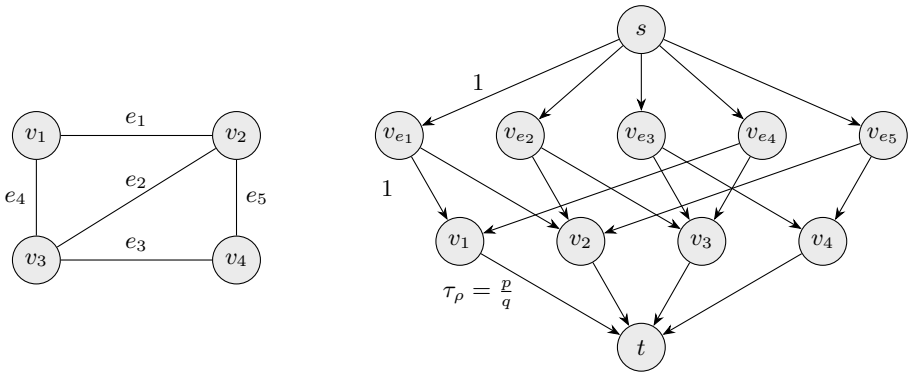

Figure 3 A sketch of the flow network which is constructed and scaled in Lemma 2.4. We remark that the figure is taken from Bentert et al. [4].

A consequence of the next lemma is that it is sufficient to consider q-quantized fractional orientations for the remainder of the section. In essence the proof takes the flow construction of Bentert et al. [4] which is depicted in Figure 3 and uses it to compute a fractional orientation. By scaling the network and using known integral properties for maximum flows the claim follows.

▶ **Lemma 2.4.** *Let G be a graph. For every $\tau_\rho = p/q \geq \rho^*(G)$, there exists a q-quantized fractional orientation ϕ with $\Delta^-_\phi(G) \leq \tau_\rho$.*

Proof. We prove the statement by scaling the flow network N by Bentert et al. [4] and then using known integral properties for maximum flows. The vertex set of the flow network N contains a source s, a sink t, all vertices from G and for every edge $e \in E(G)$ one vertex v_e. Formally this means $V(N) := \{s, t\} \cup V(G) \cup \{v_e \mid e \in E(G)\}$. For every $e \in E(G)$ add the arc (s, v_e) with capacity one and for every $u \in V(G)$ add the arc (u, t) with capacity $\tau_\rho = p/q$ to N. Moreover, for every edge $e = uw \in E(G)$ introduce two arcs (v_e, u) and (v_e, w) with capacity one. A sketch of the constructed network can be seen in Figure 3.

By Lemma 2.1 we know that there exists a fractional orientation ϕ_1 in the graph G with $\Delta^-_{\phi_1} = \rho^*(G) \leq \tau_\rho$. Consequently, the following flow f_1 is feasible in N:

$$f_1(a) := \begin{cases} 1 & \text{if } a = (s, v_e) \text{ for some } e \in E(G) \\ \phi_1(e)^u & \text{if } a = (v_e, u) \\ \deg^-_{\phi_1}(u) & \text{if } a = (u, t) \text{ for some } u \in V(G). \end{cases}$$

Note that f_1 is a maximum flow for N since all outgoing arcs from the source have a maximal throughput of 1. Now consider the network qN, which is obtained from N by multiplying all capacities with q. It is clear that qf_1, that is the flow f_1 scaled by the factor q, is a maximum flow in qN. Since all capacities in qN are integral, we know that there exists an integral maximum flow. Let qf_2 be this integral maximum flow. Then f_2 is a maximum flow in N in which the flow on every arc is a multiple of $1/q$ since qf_2 is integral. This allows us to define the q-quantized fractional orientation ϕ_2 by setting $\phi_2(e)^u := f_2((v_e, u))$. Recall that the arc (v, t) has capacity τ_ρ in N for every $v \in V(G)$ and therefore by construction $\Delta^-_{\phi_2} \leq \tau_\rho$. ◀

The idea of our algorithm is to extend the dynamic programming algorithm from Hanaka et al. [22] which is an extension of an algorithm from Bodlaender et al. [5]. The main idea is bottom-up dynamic programming over a given cliquewidth expression T to keep track

of all possible indegree signatures. For a node t of T, a pair (k, A), where k is a natural number and $A = (A_{i,j})_{j\in[1,\mathrm{cw}],j\in[0,p]}$ is a $\mathrm{cw} \times (p+1)$ matrix, is an *indegree signature* if there exists $K_t \subseteq V(G_t)$ with $|K_t| \leq k$ and a fractional orientation ϕ_t of $G_t - K_t$ such that the number of label i vertices with indegree j/q in $G_t - K_t$ is exactly $A_{i,j}$. For a label i, we refer to $j \in [0, p]$ as an *indegree class* of i. The improvement from [22] over [5] is the combinatorial observation that the exact size of an indegree class does not matter if it is large enough ($> 3p^3$ for us) as in this case all newly created edges can be oriented towards the vertices from the large indegree class. To further extend the algorithm from Hanaka et al. [22], we use the fact that we can restrict ourselves to q-quantized fractional orientations when the maximum permitted indegree is p/q.

▶ **Theorem 2.5.** *Given a* cw*-expression of the graph,* Bounded Density Vertex Deletion *can be solved in* $q^{O(p^7)}p^{O(\mathrm{cw}\cdot p)} \cdot n^{O(1)}$.

Proof. Let G be the input graph and T a given cw-expression of G. In the following we state how to compute all indegree signatures (k, A) for the nodes of T in a bottom-up manner. The smallest number k for which there is an indegree signature (k, A) of the root r equals the minimum number of vertices which have to be deleted such that there exists a fractional orientation with maximum indegree at most p/q. Without explicitly mentioning it again, we always assume that we set an entry of the matrix A of an indegree signature (k, A) to "large" if the respective value would be larger than $3p^3$. We need to distinguish between four cases for a node t of T, where the first three cases are essentially the same as in [5, 22].

Introduce Node $\circ_i$. We have two indegree signatures. If $k = 0$, then $A_{i,0} = 1$ and all other entries of A are zero. If $k = 1$, then A is the zero matrix.

Union Node $G_{t'} \oplus G_{t''}$. Let t' and t'' be the children of t. Two indegree signatures (k', A') and (k'', A'') of the nodes t' and t'' respectively are merged to $(k := k' + k'', A)$ where we set $A_{i,j} := A'_{i,j} + A''_{i,j}$ for all j.

Relabel Node $\rho(i_1, i_2)$. Let t' be the child of t. For every indegree signature (k', A') of t' we obtain an indegree signature $(k := k', A)$ of t by setting $A_{i_1,j} := 0, A_{i_2,j} := A'_{i_1,j} + A'_{i_2,j}$ and $A_{i,j} := A'_{i,j}$ fo all $i \notin \{i_1, i_2\}$ and all j.

Join Node $\eta(i_1, i_2)$. Let t' be the child of t. Join nodes are the only nodes where edges are created. As we have multiple options how to orient the new edges, we potentially compute a lot of indegree signatures (k, A) of t from one indegree signature (k', A') of t'. If both rows A'_{i_1} and A'_{i_2} contain an entry with the label "large", then the indegree signature (k', A') of t' can be discarded since $\rho(K_{2p+1,2p}) > p/q$. Now suppose the row A'_{i_2} contains no label "large" and let A'_{i_1,j_1} be an entry of the row A'_{i_1}. There are two cases.

- A'_{i_1,j_1} contains the label "large". In this case we can orient all newly created edges towards the vertices from A'_{i_1,j_1} and move them to the larger indegree class $j_1 + sq$ where $s := \sum_{j=0}^{p} A'_{i_2,j}$. The following claim (adjusted from [22] to our setting) shows that this does not change the smallest k for which there is an indegree signature of the root r.

  ▷ Claim 2.6. If there is an indegree signature (k, A_1) of the root r when not all of the created edges are fully oriented towards A'_{i_1,j_1}, then there is also an indegree signature (k, A_2) of the root r when all of them are fully oriented towards A'_{i_1,j_1}.

  Proof. We show that there exists at least one vertex from the indegree class j_1 that receives the full orientation of every edge introduced in ascendants of the node t' in T without exceeding the indegree p/q. This implies that all vertices from the same indegree

class j_1 of label i_1 can receive the full orientation of these edges without exceeding the indegree p/q. Since $A'_{i_1,j_1} > 3p^3$ and $\rho(K_{2p+1,2p}) > p/q$ it is clear that $\sum_{j=0}^{p} A'_{i_2,j} < 2p$. As we are only considering q-quantized fractional orientations, we know that every vertex with label i_2 can receive orientation from at most p edges. Thus, more than $3p^3 - 2p^2$ of the vertices from the indegree-class j_1 receive the full orientation from all new edges to which they are incident and thus are moved to a higher indegree-class $j' > j_1$. There are only $p+1$ indegree-classes, so a vertex can move at most p times to a higher indegree-class. Consequently, more than $3p^3 - 2p^3$ vertices of j_1 are assigned all newly created edges in the ascendants of t'. ◁

- A'_{i_1,j_1} does not contain the label "large". In this case, the algorithm needs to consider all possible q-quantized fractional orientations of the new edges that are incident to the vertices from A'_{i_1,j_1}. Since $A'_{i_1,j_1} \leq 3p^3$ and each of the $p+1$ indegree classes of the label i_2 also has size at most $3p^3$, there are $O(p^7)$ new edges between the indegree class j_1 of the label i_1 and vertices of the label i_2. Each edge has $q+1$ possible orientations, and therefore the algorithm has to consider $(q+1)^{O(p^7)}$ possible q-quantized fractional orientations for A'_{i_1,j_1}. If one vertex would get indegree larger than p/q, then the respective fractional orientation is ignored. Otherwise, the vertices are moved to the corresponding indegree classes in A according to the fractional orientation.

For each of the $p+1$ indegree classes of the label i_1, we create at most $(q+1)^{O(p^7)}$ indegree signatures $(k := k', A)$ of t for a given indegree signature (k', A') of t'.

For an indegree signature (k, A) every entry of A contains either the label "large" or a number from the interval $[0, 3p^3]$. Hence, for every node t of the cliquewidth expression there exist at most $n \cdot (3p^3+2)^{\mathrm{cw}\cdot(p+1)}$ indegree signatures. Combined with the running time of the most expensive operation (join), this yields a running time of $q^{O(p^7)}p^{O(\mathrm{cw}\cdot p)} \cdot n^{O(1)}$. ◀

2.2 Treewidth and Target Density

Theorem 2.5 implies that BDVD is in FPT parameterized by treewidth, p and q where $p, q \in \mathbb{N}$ with $\tau_\rho = p/q$ as the treewidth of a graph bounds in the cliquewidth [11, 9]. As the cliquewidth can be exponential in the treewidth, however, this gives a double-exponential running time. Using dynamic programming over nice tree decompositions we obtain a single-exponential running time for the parameter treewidth and q. The algorithm follows the standard bottom-up dynamic programming approach on nice tree decompositions. To keep the number of states manageable, we again use quantized fractional orientations, which restrict the number of possible orientations that need to be considered for each bag of the decomposition. For every bag, the algorithm considers all subsets of the bag and evaluates all feasible *weighted* fractional orientations for them, where the weights are used to track the indegree contributions of already forgotten vertices adjacent to vertices contained in the subset, while simultaneously storing the minimum number of vertices that must be deleted so that the remaining graph admits a fractional orientation consistent with the considered weighted fractional orientation and with maximum indegree at most τ_ρ. We can further show that $p \leq \mathrm{tw}\cdot q$ (otherwise no vertex needs to be deleted), which leads to the next theorem. The $\star$ indicates that the proof is deferred to the full version [31].

▶ **Theorem 2.7 ($\star$).** *Given an n-vertex graph G together with a nice tree decomposition of width at most* tw *and a target density $\tau_\rho = p/q$, then* Bounded Density Vertex Deletion *can be solved in time $(q\cdot p)^{\mathcal{O}(\mathrm{tw}^2)} \cdot n^{\mathcal{O}(1)}$ and in time $(\mathrm{tw}\cdot q)^{\mathcal{O}(\mathrm{tw}^2)} \cdot n^{\mathcal{O}(1)}$.*

2.3 Vertex Integrity

In the previous subsections we established fixed-parameter tractability of BDVD for restricted target density. As we will see in Section 3 the problem becomes much harder in case we allow arbitrary target densities. For this setting we thus turn to the larger parameter vertex integrity vi of the input graph G and show that BDVD is in FPT parameterized by vi. The *vertex integrity* of a graph G is the smallest number vi of vertices such that we can remove vi vertices and every component in the obtained graph has size at most vi. Note that such a witness set of at most vi vertices can be computed in FPT time [14]. The parameter vi lies between the vertex cover number and the treedepth [20]. In this sense our results improves a result from Bazgan et al. [3], who showed that BDVD is in FPT for the vertex cover number, and provides a sharp bound for the tractability in the current landscape of structural parameters.

▶ **Theorem 2.8.** *Given a witness for the vertex integrity,* Bounded Density Vertex Deletion *can be solved in* $2^{\mathrm{vi}\cdot 2^{O(\mathrm{vi}^2)}} n^{O(1)}$.

Proof. Let $(G, k, \tau_\rho = p/q)$ be a BDVD-instance and $|X| \leq \mathrm{vi}$ a set of vertices such that every component of $G - X$ has size at most vi. By branching over all possible subsets of X, we can guess which vertices of X we do and do not choose into the solution. For the rest of the proof we therefore assume no vertex from the set X is deleted, by considering only vertices not taken into the solution by the branching. The idea of the algorithm is to formulate an ILP which decides how many components with the same "type" are transformed in the same way by deleting the vertices of a solution.

We say two components C_1 and C_2 of $G - X$ have the same *type* t if there exists a bijection $\phi : V(C_1) \to V(C_2)$ such that for all $u \in X, v, v' \in V(C_1)$

- $uv \in E(G)$ if and only if $u\phi(v) \in E(G)$ and
- $vv' \in E(G)$ if and only if $\phi(v)\phi(v') \in E(G)$.

The number of different types is bounded by $\mathrm{vi}\cdot 2^{2\,\mathrm{vi}^2}$.

For every component type t of $G - X$ and for every component type s which can be obtained by deleting vertices from a component with type t, we introduce one variable x_t^s. Intuitively, the value of the variable corresponds to the number of components with type t which are turned into components with type s when deleting the vertices of a solution. If type s can be obtained from t, then we write $s \subseteq t$. Further, we denote by k_t^s the number of vertices which need to be deleted in this process and by n_t the number of components with type t in $G - X$. This allows us to formulate the following objective function and the first constraints:

$$\min \quad \sum_{t} \sum_{s, s\subseteq t} k_t^s x_t^s$$

$$\text{s.t.} \quad \sum_{s, s\subseteq t} x_t^s = n_t \qquad \forall \text{ types } t \text{ of } G - X. \tag{1}$$

$$0 \leq x_t^s \qquad \forall \text{ types } t, s \text{ of } G - X. \tag{2}$$

Let G' be the graph obtained from a solution satisfying the above constraints, meaning G' is the subgraph of G which contains X and for any type s, the graph $G' - X$ has $\sum_{t, s\subseteq t} x_t^s$ components of type s. It remains to add constraints which ensure G' contains no subgraph with density larger than $\tau_\rho = p/q$.

▷ Claim 2.9. Suppose H is a densest subgraph in G' and let C and C' be two components of the same type in $G' - X$. We can then assume that H restricted to C and C' yields again two components with the same type.

Proof. Since C and C' are of the same type in $G' - X$, there exists a bijection $\phi : V(C) \to V(C')$. Without loss of generality assume $\rho(H[V(C) \cup X]) \geq \rho(H[V(C') \cup X])$. Then updating H such that for any $v \in V(C)$ it holds that $\phi(v) \in V(H) \iff v \in V(H)$, cannot decrease $\rho(H)$. ◁

As a consequence we only need to care about the density of subgraphs which behave the same on all components with the same type. This allows us to formulate density constraints that ensure that in each remaining subgraph H we will have $|E(H)|/|V(H)| \leq \tau_\rho$, or equivalently $q \cdot |E(H)| \leq p \cdot |V(H)|$. A subtype assignment w assigns each type s one type $w(s)$ such that $w(s) \subseteq s$; the idea is that $w(s)$ indicates the part of s that are in the subgraph H. Let $W \subseteq G[X]$. We denote the number of vertices in a component of type $w(s)$ by $n_{w(s)}$ and the number of edges in the component and between the component and W by $m^W_{w(s)}$. Finally, for each subgraph $W \subseteq G[X]$ and subtype assignment w we introduce the following constraint that ensures $q \cdot |E(H)| \leq p \cdot |V(H)|$ for the corresponding subgraph:

$$q \cdot \big(|E(W)| + \sum_s \sum_{s \subseteq t} m^W_{w(s)} x^s_t\big) \leq p \cdot \big(|V(W)| + \sum_s \sum_{s \subseteq t} n_{w(s)} x^s_t\big). \tag{3}$$

The ILP has $2^{O(\mathrm{vi}^2)}$ many variables and $2^{O(\mathrm{vi}^2)}$ many constraints. Thus, the ILP can be solved in $2^{\mathrm{vi} \cdot 2^{O(\mathrm{vi}^2)}} n^{O(1)}$ time [17, 25, 24]. ◀

2.4 Max Leaf Number

The *max leaf number* ml of a graph G is the maximum number of leaves in any spanning tree of G. It is a rather large parameter since on graphs with bounded max leaf number the number of high-degree vertices is also bounded. To be more precise, let $V_{\geq 3}$ be the set of vertices with degree at least three in G, then the size of $V_{\geq 3}$ is bounded by $O(\mathrm{ml}^2)$ [16, 19]. The next theorem uses this fact to show that Bounded Density Vertex Deletion is in FPT with respect to the max leaf number. In case the target density is greater than one, it is sufficient to branch on all possible sets of vertices in $V_{\geq 3}$ to take into a solution. In case the target density is below one, after guessing which vertices form $V_{\geq 3}$ to take into a solution and how to cut the paths between them we require an ILP to calculate whether all resulting connected components can be made small enough by well-placed cuts.

▶ **Theorem 2.10** (⋆). *Bounded Density Vertex Deletion can be solved in* $\mathrm{ml}^{O(\mathrm{ml}^2)} \cdot n^{O(1)}$ *running time.*

We remark that the case of $\tau_\rho > 1$ from the algorithm above actually shows fixed-parameter tractability for the smaller parameter feedback edge number. After removing all degree-one vertices, the number of vertices with degree at least three is bounded by a function of the feedback edge number. In contrast to this, the case of $\tau_\rho < 1$ is unclear regarding the feedback edge number. It seems plausible that Bounded Density Vertex Deletion is W[1]-hard for this parameter since the same is true for the edge deletion variant [3].

3 Hardness Results

Here, we contrast our algorithmic findings with hardness results. In particular, we provide two reductions that show W[1]-hardness of BDVD with respect to treedepth combined with further parameters. This implies hardness for smaller parameters such as treewidth and cliquewidth for which we previously established fixed-parameter tractability when combined with p and q, the "encoding size" of $\tau_\rho = p/q$.

3.1 Treedepth, Distance to Cograph, and Distance to Interval Graph

In the previous section we established that BDVD is fixed-parameter tractable with respect to vertex integrity. We contrast this result by showing that BDVD is W[1]-hard parameterized by the treedepth. The reduction even gives the stronger result that BDVD is W[1]-hard for the combined parameter treedepth + distance to cograph + distance to interval graph. The *treedepth* of a graph G is the minimum height of a rooted forest F such that $G \subseteq \text{clos}(F)$, where $\text{clos}(F)$ denotes the graph with vertex set $V(F)$ and edge set $\big\{\{u,v\} \mid u \text{ is an ancestor of } v \text{ in } F \text{ and } u \neq v\big\}$ [29]. A graph G is a *cograph* if it does not contain an induced P_4 [6]. A graph is called chordal if every induced subgraph that is a cycle has exactly three vertices. A graph G is called an *interval graph* if it is chordal and, for every triple of distinct vertices $v_1, v_2, v_3 \in V(G)$ such that no two of them are adjacent, at least one vertex v_i is adjacent to every path connecting the other two vertices [28].

To establish W[1]-hardness for the parameters mentioned above we provide a reduction from Restricted Unary Bin Packing, which is defined as follows [23].

Restricted Unary Bin Packing

Input: A set $S = \{s_1, \ldots, s_n\}$ of integers in unary encoding, $k \in \mathbb{Z}^+$, as well as a function $f : S \to \binom{[k]}{2} \coloneqq \{\{x,y\} \mid x,y \in \{1,\ldots,k\}, x \neq y\}$, that maps every $s_i \in S$ to a set of exactly two integers from $[k]$.

Question: Determine whether we can partition S into k subsets $S_1, \ldots, S_k$ such that for all $i \in [k]$ it holds that

- **(i)** $\sum_{s \in S_i} s = \frac{1}{k} \sum_{s \in S} s$
- **(ii)** For all $s \in S_i$, $i \in f(s)$.

▶ **Theorem 3.1.** *Bounded Density Vertex Deletion is* W[1]*-hard with respect to the combined parameter treedepth + distance to cograph + distance to interval graph.*

Proof. We provide a reduction from Restricted Unary Bin Packing which is known to be W[1]-hard parameterized by the number of bins k [23]. Let (S, k, f) be an instance of Restricted Unary Bin Packing. We construct an instance (G, k', τ_ρ) of BDVD as follows.

1. For every $p \in [k]$, create a corresponding vertex v_p in G.
2. For each $s_i \in S$ with $f(s_i) = \{p, q\}$, introduce two vertices x_p^i and x_q^i in G, making x_p^i adjacent to v_q and x_q^i adjacent to v_p. We also make x_p^i adjacent to x_q^i. Then add a set $V_i = \{y_1^i, \ldots, y_{3s_i-1}^i\}$ of $3s_i - 1$ vertices to G, and make each of x_p^i and x_q^i adjacent to every vertex in V_i. We call the induced subgraph $G[\{x_p^i\} \cup \{x_q^i\} \cup V_i]$ a choice gadget for s_i. An example can be seen in Figure 4.
3. Set $k' = |S|$ and $\tau_\rho = 1 - 1/3r+1$, where $r = \sum_{s_i \in S} s_i/k$.

Clearly, (G, k', τ_ρ) can be constructed in polynomial time in the size of (S, k, f) when all integers in S are encoded in unary. Moreover, we show that all parameters mentioned in the theorem are bounded by a function of k.

For the treedepth we construct a rooted forest F as follows. First, we add the path $(v_1, \ldots, v_k)$ to F and root F at v_1. Then, for each $s_i \in S$ with $f(s_i) = \{p, q\}$, we connect x_p^i to x_q^i and make x_q^i adjacent to every vertex $y^i \in V_i$. Finally, we connect x_p^i to v_k. The resulting rooted forest has height $k + 3$ and satisfies $G \subseteq \text{clos}(F)$. Hence, G has treedepth at most $k + 3$.

By construction it is clear that no choice gadget contains an induced P_4 and therefore is a cograph. Moreover, any induced cycle in a choice gadget consists of exactly three vertices, namely one vertex from V_i together with x_u^i and x_v^i. Hence, each choice gadget is chordal.

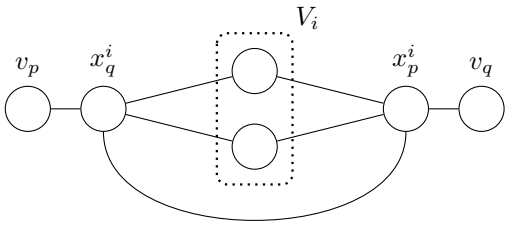

Figure 4 Choice gadget for item $s_1 = 1$ with $f(s_i) = \{p, q\}$ together with vertices v_p and v_q representing bins p and q in the reduction from Theorem 3.1.

Since both x_u^i and x_v^i are adjacent to all other vertices of the gadget, and every edge is incident to either x_u^i or x_v^i, it follows that each choice gadget is an interval graph. Since the removal of the k vertices corresponding to the bins leaves a graph where each component is a choice gadget, it follows that the distance to cograph/interval graph is at most k.

Next, we prove the correctness of the reduction, that is, we show that (S, k, f) is a yes-instance of Restricted Unary Bin Packing if and only if (G, k', τ_ρ) is a yes-instance of Bounded Density Vertex Deletion.

"⇒". Suppose there exists a solution $S_1, \ldots, S_k$ of the Restricted Unary Bin Packing-instance (S, k, f). We define a solution for the BDVD-instance by

$$V' = \bigcup_{p \in [k]} \bigcup_{s_i \in S_p} \{x_p^i\}.$$

Clearly, $|V'| = |S| = k'$ as V' contains one vertex for each item in S. It suffices to show that $G - V'$ contains no subgraph with density exceeding τ_ρ. Since $G - V'$ is a forest, it is enough to verify that every connected component of $G - V'$ has density at most τ_ρ.

Observe that each component of $G - V'$ contains exactly one vertex from $\{v_1, \ldots, v_k\}$, each corresponding to a bin. Let v_p be such a vertex. For any $s_i \in S$ with $f(s_i) = \{p, q\}$ and $s_i \in S_q$ with $p \neq q$, the vertex x_q^i adjacent to v_p has been deleted. Hence, we only need to consider the $s_i \in S_p$. Let $G_p \subseteq G - V'$ be the connected component containing v_p. Its vertex set is $V(G_p) = \{v_p\} \cup \bigcup_{s_i \in S_p} (\{x_q^i\} \cup V_i)$ and therefore $|V(G_p)| = 1 + \sum_{s_i \in S_p} 3s_i = 1 + 3r$. Since G_p is a tree, we have $|E(G_p)| = |V(G_p)| - 1 = 3r$, and thus

$$\rho(G_p) = \frac{|E(G_p)|}{|V(G_p)|} = \frac{3r}{3r+1} = \tau_\rho.$$

This completes the forward direction.

"⇐". Let V' be a solution to (G, k', τ_ρ). We first claim that V' must contain exactly one vertex from each choice gadget in G.

Indeed, each choice gadget contains a cycle, so its density is at least 1. Since $\tau_\rho < 1$, at least one vertex from every choice gadget must be deleted. There are k' pairwise disjoint choice gadgets and therefore exactly one vertex is deleted from each gadget, and no vertices are deleted from the remaining graph.

Next, we claim that the deleted vertex in any gadget must be either x_p^i or x_q^i. Since $s_i \geq 1$, we have $|V_i| \geq 2$. Moreover, any vertex in V_i forms a cycle together with x_p^i and x_q^i. Hence, deleting a single vertex from V_i does not destroy all cycles in the gadget, so the deleted vertex must be one of x_p^i or x_q^i.

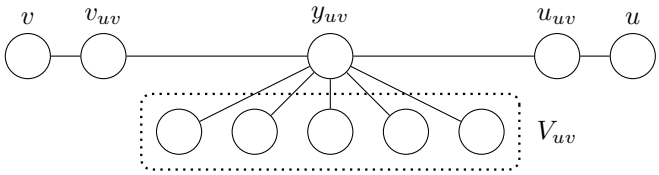

Figure 5 The corresponding gadget from G' for a single edge $\{u, v\}$ in G (and $d = 6$) from the proof of Theorem 3.2.

This allows us to define a partition $S_1, \ldots, S_k$: for each s_i, we add s_i to S_p if x_p^i is contained in V'.

Now observe that $G - V'$ has exactly k components and since V' is a solution we know that each component has size at most $3r + 1$. By construction $G - V'$ has exactly $k(3r + 1)$ vertices and subsequently we know that each component in $G - V'$ has size exactly $3r + 1$.

Let G_p be such a component containing the vertex v_p. Then $|V(G_p)| = 3r + 1$ which implies $\sum_{s \in S_p}(|V_s| + 1) = 3r$ and therefore $\sum_{s \in S_p} s = r$.

Hence, $S_1, \ldots, S_k$ satisfies requirement (i), and (ii) is enforced by construction, completing the backward direction. ◀

3.2 Treedepth and Feedback Vertex Number

The feedback vertex number of a graph G is the minimum cardinality of any set $X \subseteq V(G)$ such that $G - X$ is acyclic [12]. It is a smaller parameter than the vertex cover number and the max leaf number, for which BDVD is in FPT ([3] and Theorem 2.10). Note that in the proof of Theorem 3.1 we actually constructed a feedback vertex set, but of size not bounded by the parameter. Here, we establish that BDVD is W[1]-hard for the combined parameter feedback vertex number + treedepth.

▶ **Theorem 3.2.** *Bounded Density Vertex Deletion is* W[1]*-hard with respect to the combined parameter treedepth + feedback vertex number.*

Proof. We construct a similar reduction to Hanaka et al. [23] and reduce from Bounded Degree Vertex Deletion (known to be W[1]-hard parameterized by treedepth + feedback vertex number [18]) to Bounded Density Vertex Deletion.

Bounded Degree Vertex Deletion

Input: An undirected Graph G, a degree bound d and a solution size k.

Question: Is there a subset $S \subseteq V(G)$ with $|S| \leq k$ such that $G - S$ contains no vertex with a degree of more than d?

Let (G, k, d) be an instance of Bounded Degree Vertex Deletion. We construct an instance (G', k', τ_ρ) of Bounded Density Vertex Deletion as follows.

i. Add a copy of each vertex $v \in V(G)$ to G'.
ii. For every edge $\{u, v\} \in E(G)$ we add three vertices u_{uv}, v_{uv} and y_{uv} to G' and connect y_{uv} to u_{uv} and v_{uv}. We also connect u to u_{uv} and v to v_{uv}.
   Additionally we add a set V_{uv} of $d - 1$ vertices to G' and connect each vertex to y_{uv}.
   An example can be seen in Figure 5.
iii. At last we set $k' = k + |E(G)|$ and $\tau_\rho = {}^{d}/_{d+1}$.

Hanaka et al. [23] use the same construction of G', reducing Bounded Degree Vertex Deletion to Component Order Connectivity. As they show, for every edge gadget, at least one of the vertices u_{uv}, v_{uv}, or y_{uv} must be included in any solution S, as otherwise the

induced subgraph $G[\{u_{uv}, v_{uv}, y_{uv}\} \cup V_{uv}]$ would be too large with respect to the Component Order Connectivity constraint. Consequently, for any feasible solution S, the resulting graph $G' - S$ must necessarily be a forest.

Since Bounded Density Vertex Deletion with $\tau_\rho < 1$ merely restricts the component size of the remaining forest, the correctness of the reduction follows directly from the arguments of Hanaka et al. [23]. ◀

4 Conclusion

We filled several gaps in the parameterized complexity landscape of Bounded Density Vertex Deletion, including the resolution of an open question of Bazgan et al. [3] about the fixed-parameter tractability with respect to treewidth, see Figure 2 for an overview. A common theme in our results is that restricting the target density τ_ρ makes a significant difference in the complexity of BDVD. Without restricting the target density, when considering parameters related to (very) sparse graph such as feedback vertex number, feedback edge number, or max leaf number, the challenging case seems to be $\tau_\rho < 1$. Then BDVD behaves like a balanced partitioning problem, which is hard even in very sparse graphs [3, 15].

We leave open the question of the parameterized complexity of BDVD for parameters between the (W[1]-hard) distance to cograph and distance to interval graph and the (fixed-parameter tractable) vertex cover number, e.g. distance to cluster and twin-cover number. Another open question posed by Bazgan et al. [3] remains whether BDVD is hard for the combined parameter treewidth and solution size. Since the sister problem Bounded Density Edge Deletion is known to be W[1]-hard when parameterized by treewidth and solution size, it seems plausible that BDVD exhibits similar hardness. Similarly the parameterized complexity of BDVD for the feedback edge number (for which Bounded Density Edge Deletion is W[1]-hard) is an open question.

References

1 Yuichi Asahiro, Jesper Jansson, Eiji Miyano, and Hirotaka Ono. Graph orientations optimizing the number of light or heavy vertices. *Journal of Graph Algorithms and Applications*, 19(1):441–465, 2015. doi:10.7155/jgaa.00371. 2

2 Yuichi Asahiro, Jesper Jansson, Eiji Miyano, and Hirotaka Ono. Degree-constrained graph orientation: Maximum satisfaction and minimum violation. *Theory of Computing Systems*, 58(1):60–93, 2016. doi:10.1007/s00224-014-9565-5. 2

3 Cristina Bazgan, André Nichterlein, and Sofia Vazquez Alferez. Destroying densest subgraphs is hard. *Journal of Computer and System Sciences*, 151, 2025. doi:10.1016/j.jcss.2025.103635. 1, 2, 3, 4, 5, 9, 10, 13, 14

4 Matthias Bentert, Tom-Lukas Breitkopf, Vincent Froese, Anton Herrmann, and André Nichterlein. Density matters: A complexity dichotomy of deleting edges to bound subgraph density. In *Proceedings of the 43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026)*, LIPIcs. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. doi:10.4230/LIPIcs.STACS.2026.12. 2, 3, 6

5 Hans L. Bodlaender, Hirotaka Ono, and Yota Otachi. Degree-constrained orientation of maximum satisfaction: Graph classes and parameterized complexity. *Algorithmica*, 80(7):2160–2180, 2018. doi:10.1007/S00453-017-0399-9. 2, 6, 7

6 Andreas Brandstädt, Van Bang Le, and Jeremy P. Spinrad. *Graph Classes: a Survey*, volume 3 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 1999. 11

7 Karthekeyan Chandrasekaran, Chandra Chekuri, and Shubhang Kulkarni. On Deleting Vertices to Reduce Density in Graphs and Supermodular Functions. In *Proceedings of the 52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*, Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025. doi:10.4230/LIPIcs.ICALP.2025.43. 2

8 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization (APPROX 2000)*, LNCS. Springer, 2000. doi:10.1007/3-540-44436-X_10. 2, 5

9 Derek G. Corneil and Udi Rotics. On the relationship between clique-width and treewidth. *SIAM Journal on Computing*, 34(4):825–847, 2005. doi:10.1137/S0097539701385351. 8

10 Bruno Courcelle, Johann A. Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009. 5

11 Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discret. Appl. Math.*, 101(1-3):77–114, 2000. doi:10.1016/S0166-218X(99)00184-5. 8

12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 1st edition, 2015. 4, 13

13 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. doi:10.1007/978-1-4471-5559-1. 4

14 Pål Grønås Drange, Markus S. Dregi, and Pim van 't Hof. On the computational complexity of vertex integrity and component order connectivity. *Algorithmica*, 76(4):1181–1202, 2016. doi:10.1007/s00453-016-0127-x. 9

15 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)*, LNCS, pages 122–133. Springer, 2009. doi:10.1007/978-3-642-11269-0_10. 14

16 David Eppstein. Metric dimension parameterized by max leaf number. *Journal of Graph Algorithms and Applications*, 19(1):313–323, 2015. doi:10.7155/jgaa.00360. 10

17 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Comb.*, 7(1):49–65, 1987. doi:10.1007/BF02579200. 10

18 Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On structural parameterizations of the bounded-degree vertex deletion problem. *Algorithmica*, 83(1), 2021. doi:10.1007/s00453-020-00758-8. 13

19 Jaroslav Garvardt and Christian Komusiewicz. Modularity clustering parameterized by max leaf number. In Édouard Bonnet and Pawel Rzazewski, editors, *Proceedings of the 19th International Symposium on Parameterized and Exact Computation (IPEC 2024)*, LIPIcs, pages 16:1–16:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.IPEC.2024.16. 10

20 Tatsuya Gima, Tesshu Hanaka, Masashi Kiyomi, Yasuaki Kobayashi, and Yota Otachi. Exploring the gap between treedepth and vertex cover through vertex integrity. *Theoretical Computer Science*, 918:60–76, 2022. doi:10.1016/j.tcs.2022.03.021. 9

21 Andrew V. Goldberg. Finding a maximum density subgraph. Technical report, University of California at Berkeley, 1984. 2

22 Tesshu Hanaka, Ioannis Katsikarelis, Michael Lampis, Yota Otachi, and Florian Sikora. Parameterized orientable deletion. *Algorithmica*, 82(7):1909–1938, 2020. doi:10.1007/s00453-020-00679-6. 2, 3, 6, 7

23 Tesshu Hanaka, Michael Lampis, Manolis Vasilakis, and Kanae Yoshiwatari. Parameterized Vertex Integrity Revisited. In *Proceedings of the 49th International Symposium on Mathematical Foundations of Computer Science (MFCS 2024)*, Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.MFCS.2024.58. 11, 13, 14

24 Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538. 10

25 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415. 10

26 Tuukka Korhonen and Marek Sokolowski. Almost-linear time parameterized algorithm for rankwidth via dynamic rankwidth. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, (STOC 2024)*, pages 1538–1549. ACM, 2024. doi:10.1145/3618260.3649732. 5

27 Tommaso Lanciano, Atsushi Miyauchi, Adriano Fazzone, and Francesco Bonchi. A survey on the densest subgraph problem and its variants. *ACM Computing Surveys*, 56(8), April 2024. doi:10.1145/3653298. 1, 2

28 Cornelis Lekkeikerker and Johan Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51(1), 1962. URL: http://eudml.org/doc/213681. 11

29 Jaroslav Nesetril and Patrice Ossona De Mendez. *Sparsity: graphs, structures, and algorithms*. Springer, 2014. doi:10.1007/978-3-642-27875-4. 11

30 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. 1, 2

31 Jakob Raupach, Tom-Lukas Breitkopf, Anton Herrmann, and André Nichterlein. On the parameterized complexity of bounded-density vertex deletion, 2026. arXiv:2606.26012. 1, 8