Bandwidth Parameterized by Cluster Vertex Deletion Number

Tatsuya Gima
JSPS Research Fellow, Nagoya University, Japan

Eun Jung Kim
Université Paris-Dauphine, PSL University, CNRS UMR7243, LAMSADE, Paris, France

Noleen Köhler
Université Paris-Dauphine, PSL University, CNRS UMR7243, LAMSADE, Paris, France

Nikolaos Melissinos
Department of Theoretical Computer Science, Faculty of Information Technology, Czech Technical University in Prague, Czech Republic

Manolis Vasilakis
Université Paris-Dauphine, PSL University, CNRS UMR7243, LAMSADE, Paris, France

Abstract

Given a graph G and an integer b, Bandwidth asks whether there exists a bijection π from $V(G)$ to $\{1, \ldots, |V(G)|\}$ such that $\max_{\{u,v\} \in E(G)} |\pi(u) - \pi(v)| \leq b$. This is a classical NP-complete problem, known to remain NP-complete even on very restricted classes of graphs, such as trees of maximum degree 3 and caterpillars of hair length 3. In the realm of parameterized complexity, these results imply that the problem remains NP-hard on graphs of bounded pathwidth, while it is additionally known to be W[1]-hard when parameterized by the treedepth of the input graph. In contrast, the problem does become FPT when parameterized by the vertex cover number of the input graph. In this paper, we make progress towards the parameterized (in)tractability of Bandwidth. We first show that it is FPT when parameterized by the cluster vertex deletion number cvd plus the clique number ω of the input graph, thus generalizing the previously mentioned result for vertex cover. On the other hand, we show that Bandwidth is W[1]-hard when parameterized only by cvd. Our results generalize some of the previous results and narrow some of the complexity gaps.

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

Keywords and phrases Bandwidth, Clique number, Cluster vertex deletion number, Parameterized complexity

Digital Object Identifier 10.4230/LIPIcs.IPEC.2023.21

Related Version *Full Version*: https://arxiv.org/abs/2309.17204

Funding Our research visit to Nagoya University, Japan was funded by PRC CNRS JSPS project PARAGA (Parameterized Approximation Graph Algorithms).
Tatsuya Gima: Partially supported by JSPS KAKENHI Grant Number JP23KJ1066.
Eun Jung Kim: Supported by ANR project ANR-18-CE40-0025-01 (ASSK).
Noleen Köhler: Supported by ANR project ANR-18-CE40-0025-01 (ASSK).
Nikolaos Melissinos: Supported by the CTU Global postdoc fellowship program.
Manolis Vasilakis: Partially supported by ANR project ANR-21-CE48-0022 (S-EX-AP-PE-AL).

Acknowledgements We would like to thank Virginia Ardévol Martínez and Yota Otachi for interesting discussions at the preliminary stages of this work.

1 Introduction

Given an undirected graph G and an integer b, Bandwidth asks whether there exists a bijection $\pi : V(G) \to \{1, \ldots, |V(G)|\}$ of the vertices of G (called an *ordering*) such that $\max_{\{u,v\} \in E(G)} |\pi(u) - \pi(v)| \leq b$. The main motivation behind its study dates back to over half a century: a closely related problem in the field of matrix theory was first studied in the 1950's, while in the 1960's it was formulated as a graph problem, finding applications in minimizing (average) absolute error in codes, and has been extensively studied ever since [6, 11, 12, 13, 22, 30].

Bandwidth is long known to be NP-complete [6, 43]; as a matter of fact, it remains NP-complete even on very restricted classes of graphs, such as trees of maximum degree 3 [27], caterpillars of hair length 3 [40] and cyclic caterpillars of hair length 1 [41]. Considering these NP-hardness results, in this paper we focus on the parameterized complexity of Bandwidth. When parameterized by the natural parameter b, Bandwidth is known to be in XP [29, 44], whilst it is W[t]-hard for all positive integers t, even when the input graph is a tree [3, 4]. In fact, Bandwidth cannot be solved in time $f(b)n^{o(b)}$ even on trees of pathwidth at most two, unless the Exponential Time Hypothesis fails [17]. Regarding structural parameterizations, the previously mentioned results imply that Bandwidth is para-NP-complete when parameterized by the pathwidth or the treewidth plus the maximum degree of the input graph; the latter implies NP-completeness also on graphs of constant tree-cut width [26]. Moreover, it is known to be W[1]-hard parameterized by the treedepth of the input graph [28]. In contrast, the problem does become fixed-parameter tractable (FPT) when parameterized by the vertex cover number [20], the neighborhood diversity [1], or the max leaf number [19] of the input graph.

In the last few years, a plethora of structural parameters have been introduced, in an attempt to precisely determine the limits of tractability of algorithmic problems that are FPT by vertex cover, yet become W[1]-hard when parameterized by more general parameters, such as treewidth or clique-width. Some of the most well-studied such parameters are treedepth [42], twin cover number [24], cluster vertex deletion number [16], vertex integrity [28], shrub-depth [25], neighborhood diversity [35], and modular-width [23]. The tractability of Bandwidth with respect to those parameters has remained largely unexplored, with the exception of treedepth [28] and neighborhood diversity [1].

Cluster vertex deletion number lies between clique-width and vertex cover number (more precisely twin cover number), and is defined as the minimum size of a set of vertices whose removal induces a cluster graph, i.e. all of its components are cliques. It was first considered as a structural parameter in [16], and has been used to show parameterized (in)tractability results in multiple occasions ever since [2, 5, 7, 8, 33, 34, 38]. Notice that Bandwidth is trivial on cluster graphs; it suffices to check that the clique number is at most $b + 1$, as any optimal ordering places the vertices of every clique consecutively, for some ordering of the cliques. Therefore, its tractability when parameterized by the cluster vertex deletion number of the input graph poses a very natural question.

Our contribution. In the current work, we present both tractability and intractability results for Bandwidth when cluster vertex deletion number is a parameter of the problem (see Figure 1 for an overview of our results and the relationships between the structural parameters mentioned). We first prove that Bandwidth is FPT when parameterized by $\text{cvd} + \omega$, where cvd and ω denote the cluster vertex deletion number and clique number of the input graph respectively. This generalizes the tractability result for vertex cover number

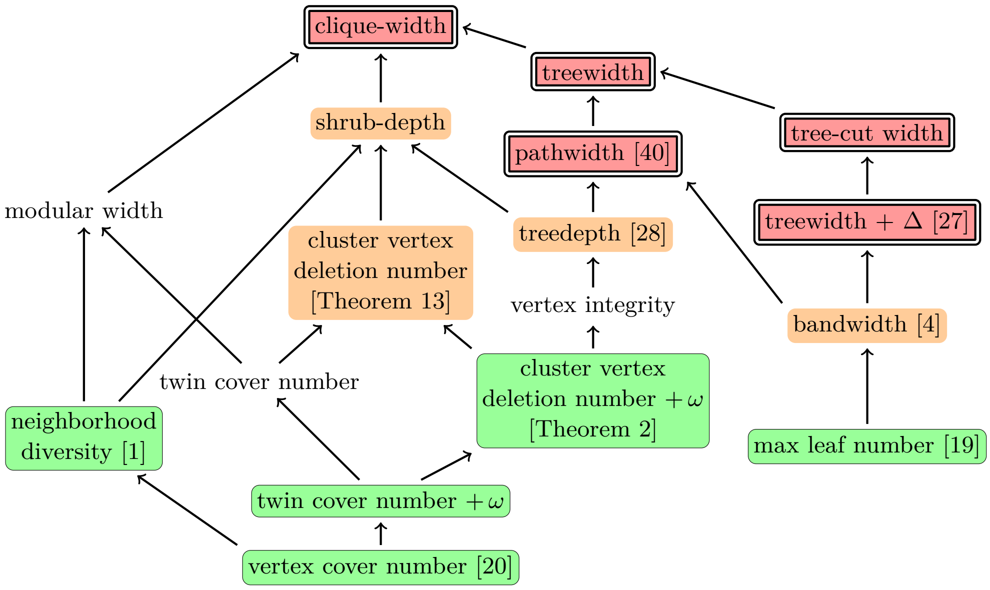

Figure 1 Our results and hierarchy of some related structural graph parameters, where ω and Δ denote the clique number and the maximum degree of the input graph, respectively. Arrows between parameters indicate generalization relations, that is, for any graph, if the parameter at the tail of an arrow is a constant then the parameter at the head of the arrow is also a constant. The reverse does not hold in this figure. The framed green, frameless orange, and double framed red rectangles indicate fixed-parameter tractable, W[∗]-hard, and NP-complete cases, respectively.

of [20], and follows the same idea of encoding the problem as an *integer linear program* (ILP) of a small number of variables. Solving said ILP, one can verify whether there exists any ordering π of the vertices of G such that a) $|\pi(v) - \pi(u)| \leq b$ for all $\{u, v\} \in E(G)$, and b) π is "nice", where an ordering is nice if it has some specific properties. Proving that there exists a nice ordering π that minimizes $\max_{\{u,v\} \in E(G)} |\pi(v) - \pi(u)|$ then yields the stated result.

A natural question that arises from the previous result is whether it is necessary for both cvd and ω to be parameters of the problem in order to assure fixed-parameter tractability. Notice that BANDWIDTH is NP-complete even when $\omega \leq 2$, since that is the case for trees. Therefore, we proceed by studying the problem's tractability when parameterized only by cvd. In this setting, we show that BANDWIDTH is W[1]-hard via a reduction from UNARY BIN PACKING, thus positively answering the previous question. Note that the W[1]-hardness of BANDWIDTH when parameterized by treedepth is also shown via a reduction from UNARY BIN PACKING [28].

Related work. BANDWIDTH is one of the so-called *graph layout* problems (see the survey of [14]). As far as the structural parameterized complexity of such problems is concerned, Fellows, Lokshtanov, Misra, Rosamond, and Saurabh [20] were the first to prove FPT results for a multitude of them when parameterized by the vertex cover number of the input graph, making use of ILP formulations. Since then, not much progress has been made on that front, with a notable exception being IMBALANCE, which was shown to be FPT when parameterized by twin cover number plus ω [39], vertex integrity [28], or tree-cut width [26], while it belongs to XP when parameterized by twin cover [39]. MINIMUM LINEAR ARRANGEMENT is known to be FPT parameterized by max leaf number, or edge clique number of the input graph [18], as well as by the vertex cover number [37]. Lastly, as far as Cutwidth is concerned, a $2^{\mathcal{O}(\mathrm{vc})}n^{\mathcal{O}(1)}$ time algorithm was presented in [10], improving over the ILP formulation of [20], where vc denotes the vertex cover number of the input graph.

Organization. In Section 2 we discuss the general preliminaries, followed by the FPT algorithm in Section 3 and the hardness result in Section 4. Lastly, in Section 5 we present the conclusion as well as some directions for future research. Proofs marked with ($\star$) are in the full version of the paper.

2 Preliminaries

Throughout the paper we use standard graph notation [15], and we assume familiarity with the basic notions of parameterized complexity [9]. We assume that $\mathbb{N}$ is the set of all non-negative integers. All graphs considered are undirected without loops. The *clique number* of a graph G, denoted by $\omega(G)$, is the size of its largest induced clique. For $x, y \in \mathbb{Z}$, let $[x, y] = \{z \in \mathbb{Z} : x \leq z \leq y\}$, while $[x] = [1, x]$. For $\mathcal{I}_i = [a_i, b_i]$, we say that intervals $\mathcal{I}_1, \dots, \mathcal{I}_k$ *partition* interval $\mathcal{I} = [a, b]$ if $\mathcal{I} = \bigcup_{i \in [k]} \mathcal{I}_i$ and $\mathcal{I}_i \cap \mathcal{I}_j = \emptyset$, for any $1 \leq i < j \leq k$. Additionally, let $\mathcal{I}_i < \mathcal{I}_j$ if $b_i < a_j$. For a function $f : A \to B$ and $A' \subseteq A$, let $f(A') = \{f(a) \in B : a \in A'\}$. Moreover, let $\max(f(A')) = \max\{f(a) : a \in A'\}$ and $\min(f(A'))$ defined analogously.

Let G be a graph and $\pi : V(G) \to [n]$ an ordering of its vertices. We define the *stretch* of an edge $e = \{u, v\} \in E(G)$ with regard to π as $\mathrm{stretch}_\pi(e) = |\pi(u) - \pi(v)|$. We define the *stretch* of π to be the maximum stretch of the edges of G, i.e. $\mathrm{stretch}(\pi) = \max_{e \in E(G)} \mathrm{stretch}_\pi(e)$. The *bandwidth* of G, denoted $\mathrm{bw}(G)$, is the minimum stretch of any vertex ordering $\pi : V(G) \to [n]$.

▶ Remark 1. Note that the stretch of a vertex ordering is invariant under isomorphism, which means in particular that $\mathrm{stretch}(\pi) = \mathrm{stretch}(\pi \circ f)$ for any vertex ordering $\pi : V(G) \to [n]$ and any automorphism $f : V(G) \to V(G)$ of G.

A *cluster deletion set* of a graph G is a set $S \subseteq V(G)$ such that every component of $G - S$ is a clique. If S is a cluster deletion set, we call the components of $G - S$ *clusters*. The *cluster vertex deletion number* of a graph G, denoted by $\mathrm{cvd}(G)$, is the size of its minimum cluster deletion set.

In the Unary Bin Packing problem, we are given a set of integers $A = \{a_j : j \in [n]\}$ in unary, as well as $k \in \mathbb{N}$, and are asked to determine whether there exists a partition $(S_1, \dots, S_k)$ of A such that $\sum_{a_j \in S_i} a_j = \sum_{j \in [n]} a_j / k$ for every $i \in [k]$. Unary Bin Packing can be solved in time $n^{\mathcal{O}(k)}$ by employing dynamic programming, while it is known to be W[1]-hard parameterized by k [31].

The feasibility variant of *integer linear programming* (ILP) is to decide, given a set X of variables and a set C of linear constraints (i.e. inequalities) over the variables in X with integer coefficients, whether there is an assignment $\alpha : X \to \mathbb{Z}$ of the variables satisfying all constraints in C. It is known that the feasibility of an instance of (ILP) can be tested in $\mathcal{O}(p^{2.5p + o(p)} \cdot L)$ time, where p is the number of variables and L is the size of the input [21, 32, 36]. In other words, computing the feasibility of an ILP formula is FPT parameterized by the number of variables. Moreover, a solution can be computed in the same time if it exists.

3 An FPT-algorithm parameterized by cluster vertex deletion number plus clique number

In this section, we prove that Bandwidth is FPT when parameterized by the cluster vertex deletion number plus the clique number of the input graph.

▶ **Theorem 2.** *Bandwidth is fixed parameter tractable when parameterized by* $\mathrm{cvd} + \omega$*, where* cvd, ω *denote the cluster vertex deletion number and clique number of the input graph respectively.*

Our proof is a generalization of the FPT result for vertex cover number from [20]. The general idea for obtaining an ILP encoding of Bandwidth given a vertex cover S is to augment S by a small number (dependent only on the vertex cover number) of representative vertices of every neighborhood-type. It can be easily seen that we can modify any ordering π in such a way that the leftmost and rightmost neighbor of any vertex in S is contained in this augmented set S' without increasing the stretch. For any ordering σ of S' we can decide whether we can extend σ to an ordering of $V(G)$ of stretch at most b by encoding how vertices of certain neighborhood-types are distributed into the gaps between the vertices of S' into an ILP. By ensuring that we distribute the vertices in such a way that the leftmost and rightmost neighbor of any vertex in S is contained in S' we can bound the stretch of every edge by using one linear constraint for every edge in $G[S']$.

In our setting we can use the vertex cover approach to bound the stretch of all edges incident to the deletion set. To gain control over the stretch of edges within clusters, we show that we can convert any ordering into a nice ordering without increasing the stretch. Here niceness intuitively means, that we can order the vertices in between any two vertices of S' in such a way that vertices of the same type appear consecutively, where the type now depends on the isomorphism-type of the cluster union the deletion set. This will allow us to bound the stretch of such edges by a linear constraint as well.

3.1 Types and buckets

Let G be a graph and S a cluster deletion set of G of size k. For any vertex $v \in V(G - S)$, let $N_S(v) = N(v) \cap S$ be its *S-neighborhood*. Let $\mathcal{K} \subseteq \mathbb{N}^{2^k}$ be the set of non-negative integer vectors κ with 2^k entries for which $\|\kappa\|_1 \leq \omega(G)$. Here $\|\cdot\|_1$ denotes the 1-norm, i.e. the sum of the absolute value of the entries. We assume that the entries of the vectors in $\mathcal{K}$ are indexed by the subsets of S. We say that a cluster C has *cluster-type* $\kappa \in \mathcal{K}$ if $|\{v \in V(C) : N_S(v) = N\}| = (\kappa)_N$ for every $N \subseteq S$ where $(\kappa)_N$ denotes the entry of κ corresponding to N. We further let $\#\kappa$ denote the number of clusters of cluster-type κ in G. We say that a set $\mathcal{C}$ of clusters is *representative* if it consists of $\min\{2|S|, \#\kappa\}$ distinct clusters of type κ for every cluster-type κ. We further say that a set S' is an *extended deletion set* if $S' = S \cup \bigcup_{C \in \mathcal{C}} V(C)$ for a representative set $\mathcal{C}$ of clusters.

▶ **Lemma 3** ($\star$). *For every extended deletion set* S' *there is an ordering* $\pi : V(G) \to [n]$ *such that* $\mathrm{stretch}(\pi) = \mathrm{bw}(G)$ *and for every* $s \in S$*, the set* S' *contains vertices* $v^s_{\min}$ *and* $v^s_{\max}$*, where* $\pi(v^s_{\min}) = \min(\pi(N(s)))$ *and* $\pi(v^s_{\max}) = \max(\pi(N(s)))$*, i.e.* S' *contains the leftmost and rightmost neighbor of* s*.*

We say that an ordering $\pi : V(G) \to [n]$ is *S'-extremal* if the second property in Lemma 3 is satisfied for π.

▶ **Observation 4.** *The size of any extended deletion set* S' *is at most* $|S| + 2|S|\omega(G) \cdot 2^{2|S| \cdot \omega(G)}$.

Let $\mathcal{C}$ be a representative set of clusters, $S' = S \cup \bigcup_{C \in \mathcal{C}} V(C)$ the extended deletion set containing vertices from $\mathcal{C}$ and S and set $k' = |S'|$. A *bucket distribution* of S' is a partition $\mathcal{B} = (B_0, \ldots, B_{k'})$ of the vertices of $G - S'$. Fix a bucket distribution $\mathcal{B} = (B_0, \ldots, B_{k'})$ of S'. We call the subsets B_i *buckets* of $\mathcal{B}$.

Let $\mathcal{T} \subseteq \mathbb{N}^{2^k \times (k'+1)}$ be the set of matrices τ with $\|\tau\|_1 \leq \omega(G)$. We assume that the rows of matrices are indexed with subsets of S and the columns with $[0, k']$. We say that a cluster $C \notin \mathcal{C}$ has *distribution-type* $\tau \in \mathcal{T}$ in $\mathcal{B}$ if $|\{v \in V(C) \cap B_i : N_S(v) = N\}| = (\tau)_{N,i}$ for every $N \subseteq S$ and every $i \in [0, k']$. For every $\kappa \in \mathcal{K}$, let $\mathcal{T}_\kappa \subseteq \mathcal{T}$ denote the set of distribution-types τ such that $\sum_{i \in [0,k']} (\tau)_{N,i} = (\kappa)_N$ for every $N \subseteq S$, i.e., the set of $\tau \in \mathcal{T}$ such that any cluster of distribution-type τ has cluster-type κ.

▶ **Observation 5.** *The number of distribution-types is at most $\omega(G)^{2^{|S|} \cdot (|S'|+1)}$ for any extended deletion set S'.*

Let $\sigma : S' \to [k']$ be an ordering of the vertices of S'. We say that a vertex ordering $\pi : V(G) \to [n]$ is *compatible* with σ if for any $s_1, s_2 \in S'$ it holds that $\pi(s_1) < \pi(s_2)$ if and only if $\sigma(s_1) < \sigma(s_2)$. We say that a vertex ordering $\pi : V(G) \to [n]$ is *compatible* with σ and $\mathcal{B}$ if π is compatible with σ and $B_0 = \{v \in V(G) : \pi(v) < \pi(\sigma^{-1}(1))\}$, $B_{k'} = \{v \in V(G) : \pi(v) > \pi(\sigma^{-1}(k'))\}$ and $B_i = \{v \in V(G) : \pi(\sigma^{-1}(i)) < \pi(v) < \pi(\sigma^{-1}(i+1))\}$ for $i \in [k'-1]$.

3.2 Nice orderings

Let G be a graph and S a cluster deletion set of G. Furthermore, let $\mathcal{C}$ be a representative set of clusters, $S' = S \cup \bigcup_{C \in \mathcal{C}} V(C)$ the extended deletion set containing vertices from $\mathcal{C}$ and S and $k' = |S'|$. Additionally, we fix a bucket distribution $\mathcal{B} = (B_0, \ldots, B_{k'})$ of S' and an ordering $\sigma : S' \to [k']$.

To obtain our nice ordering we use a series of exchange arguments that will not increase the stretch. We call an ordering *nice* if it has properties (Π_1), (Π_2) and (Π_3). We will first give some intuition regarding the properties, before defining them formally.

Assume $\pi : V(G) \to [n]$ is an optimal ordering minimizing the number of edges of maximum stretch. Furthermore, let $v \in V(C)$ be a vertex which is contained in an edge of maximum stretch with regards to π and the cluster C containing v is distributed over more than one bucket. In this case, v must be either the leftmost or the rightmost vertex of C. Assuming v is the leftmost vertex of C (the other case is analogous), we can observe that every vertex $v' \in B_i$ appearing further to the right than v must have a neighbor contained in a bucket to the right of B_i and no neighbor to the left of v. Otherwise, we can reduce the stretch of the edge containing v without increasing the stretch of any edge incident to v' (and hence reducing the number of edges of maximum stretch without increasing the maximum stretch) by exchanging v and v'. Using this observation, we can assume that each bucket is partitioned into a left, a middle and a right part and every vertex with only neighbors to the left of B_i appears in the left part and every vertex having only neighbors to the right of B_i appears in the right part. Additionally, the above observation allows us to assume that within each bucket the vertices of one cluster appear consecutively (property (Π_1)).

Now assume that $\{v, w\}$ is an edge of maximum stretch as before (v appears left of w in π) and $\{v', w'\}$ is another edge such that v' appears in the same bucket as v and w' in the same bucket as w. If v' appears before v then w' has to appear before w as $\{v, w\}$ is of maximum stretch. On the other hand, if v' appears after v then w' must appear after w as otherwise exchanging w and w' either reduces the number of edges of maximum stretch or reduces the maximum stretch itself. Hence, we can assume that the relative order of the leftmost vertices of a set of clusters is the same as the relative order of the rightmost vertices of the same clusters (property (Π_2)).

Lastly, assume that C and C' are clusters of type $\tau \in \mathcal{T}$ which are not contained in just one bucket and appear next to each other (in their leftmost bucket). Assume B_ℓ is the bucket containing the leftmost vertex of C and C' and B_r the bucket containing the rightmost vertex of C and C'. We can essentially exchange $V(C) \cap B_\ell$ with $V(C') \cap B_\ell$ and at the same time $V(C) \cap B_r$ with $V(C') \cap B_r$ if certain properties about the size of these sets hold. This allows us to order the buckets in such a way, that clusters whose intersection with the leftmost (rightmost, respectively) bucket they intersect is of the same size, appear consecutively (property (Π_3)).

To state the three properties formally we use the following notation. For a distribution-type $\tau \in \mathcal{T}$ and $i \in [0, k']$, we write τ_i to denote the column of τ which is indexed by i. We define $\mathrm{LB}(\tau)$ to be the largest index $i \in [0, k']$ such that $\|\tau_j\|_1 = 0$ for any $j \in [0, i-1]$, i.e. B_i is the leftmost bucket containing vertices from clusters of type τ. We define $\mathrm{RB}(\tau)$ analogously to be the minimum index $i \in [0, k']$ such that $\|\tau_j\|_1 = 0$ for any $j \in [i+1, k']$. Additionally, we let $\#\mathrm{L}(\tau)$ be $\|\tau_{\mathrm{LB}(\tau)}\|_1$ and $\#\mathrm{R}(\tau)$ be $\|\tau_{\mathrm{RB}(\tau)}\|_1$. For every $\ell \le r \in [0, k']$ and every $n_L, n_R \in [0, \omega(G)]$, we define

$$\mathcal{T}^{(\ell,r,n_L,n_R)} = \{\tau \in \mathcal{T} : \mathrm{LB}(\tau) = \ell, \mathrm{RB}(\tau) = r, \#\mathrm{L}(\tau) = n_L, \#\mathrm{R}(\tau) = n_R\}.$$

▶ **Definition 6** (Property (Π_1))**.** *We say that an S'-extremal ordering $\pi : V(G) \to [n]$ which is compatible with σ and $\mathcal{B}$ has property (Π_1) if for every $i \in [0, k']$*

1. *the vertices of $V(C) \cap B_i$ appear consecutively in π for every cluster $C \notin \mathcal{C}$,*
2. *we can partition the interval $\pi(B_i)$ into three (possibly empty) intervals $I_R^i < I_M^i < I_L^i$ such that for every $\tau \in \mathcal{T}$ and every cluster C of distribution-type τ*
   - *$\pi(V(C) \cap B_i) \subseteq I_R^i$ if $\mathrm{LB}(\tau) \neq i$ and $\mathrm{RB}(\tau) = i$,*
   - *$\pi(V(C) \cap B_i) \subseteq I_L^i$ if $\mathrm{LB}(\tau) = i$ and $\mathrm{RB}(\tau) \neq i$,*
   - *$\pi(V(C) \cap B_i) \subseteq I_M^i$ if either $\mathrm{LB}(\tau) \neq i$ and $\mathrm{RB}(\tau) \neq i$ or $\mathrm{LB}(\tau) = \mathrm{RB}(\tau) = i$.*

Notice that while I_R^i contains the leftmost ordered vertices of B_i, we use the index R since those vertices are the rightmost vertices of their corresponding cliques. Analogously, we use I_L^i for the rightmost ordered vertices of B_i.

▶ **Definition 7** (Property (Π_2))**.** *We say that an S'-extremal ordering $\pi : V(G) \to [n]$ which is compatible with σ and $\mathcal{B}$ has property (Π_2) if for any two distribution-types $\tau, \tau' \in \mathcal{T}$ and any two clusters C and C' of distribution-type τ and τ' respectively, the following holds.*

- *If either $\mathrm{LB}(\tau) = \mathrm{LB}(\tau')$ or $\mathrm{RB}(\tau) = \mathrm{RB}(\tau')$, then for any $v \in V(C) \cap B_{\mathrm{LB}(\tau)}$, $v' \in V(C') \cap B_{\mathrm{LB}(\tau')}$, $w \in V(C) \cap B_{\mathrm{RB}(\tau)}$, $w' \in V(C') \cap B_{\mathrm{RB}(\tau')}$ we have that $\pi(v) < \pi(v')$ if and only if $\pi(w) < \pi(w')$.*

Lastly, we want the buckets to be ordered by distribution-types which will enable us to express the stretch within clusters by linear constraints. To achieve this, we define two orderings of distribution-types, dictating in which order (in a nice, optimal vertex ordering) cliques of a certain type will appear within a bucket. First, let $\mathcal{T}_R^i = \bigcup_{\substack{\ell \in [0,i-1],\\ n_L, n_R \in [\omega(G)]}} \mathcal{T}^{(\ell,i,n_L,n_R)}$ and define the ordering $\rho_i : \mathcal{T}_R^i \to [|\mathcal{T}_R^i|]$ in the following way. For any $\tau \in \mathcal{T}^{(\ell,i,n_L,n_R)}$, $\tau' \in \mathcal{T}^{(\ell',i,n'_L,n'_R)}$, we have that $\rho_i(\tau) < \rho_i(\tau')$ if either

- $\ell < \ell'$ or
- $\ell = \ell'$, $n_L \ge n_R$ and $n'_L < n'_R$ or
- $\ell = \ell'$, $n_L \ge n_R$, $n'_L \ge n'_R$ and $n_R < n'_R$ or
- $\ell = \ell'$, $n_L < n_R$, $n'_L < n'_R$ and $n_L > n'_L$ or
- $\ell = \ell'$, $n_L \ge n_R$, $n'_L \ge n'_R$, $n_R = n'_R$ and $\tau \le_{\mathrm{lex}} \tau'$ or
- $\ell = \ell'$, $n_L < n_R$, $n'_L < n'_R$, $n_L = n'_L$ and $\tau \le_{\mathrm{lex}} \tau'$.

Here $\leq_{\text{lex}}$ refers to the lexicographic order on matrices in $\mathcal{T}$ where we read the entries by lines top to bottom. However, we can replace this by any total ordering ($\leq_{\text{lex}}$ is an arbitrary choice).

Moreover, let $\mathcal{T}_L^i = \bigcup_{\substack{r\in[i+1,k'],\\ n_L,n_R\in[\omega(G)]}} \mathcal{T}^{(i,r,n_L,n_R)}$ and define the ordering $\lambda_i : \mathcal{T}_L^i \to [|\mathcal{T}_L^i|]$ by letting $\lambda_i(\tau) < \lambda_i(\tau')$ for any $\tau \in \mathcal{T}^{(i,r,n_L,n_R)}$, $\tau' \in \mathcal{T}^{(i,r',n'_L,n'_R)}$ if either

- $r < r'$ or
- $r = r'$ and $\rho_i(\tau) < \rho_i(\tau')$.

▶ Remark 8. Note that we can compute all ρ_i and λ_i in time quadratic in the size of $\mathcal{T}$.

▶ **Definition 9** (Property (Π_3)). *We say that an S'-extremal ordering $\pi : V(G) \to [n]$ which is compatible with σ and $\mathcal{B}$ has property (Π_3) if for every $i \in [0, k']$ we can partition the interval $\pi(B_i)$ into (possibly empty) intervals*

$$J_R^{(i,1)} < \cdots < J_R^{(i,|\mathcal{T}_R^i|)} < J_M^i < J_L^{(i,1)} < \cdots < J_L^{(i,|\mathcal{T}_L^i|)}$$

such that for every distribution-type $\tau \in \mathcal{T}$ and every cluster C of type τ and every $j \in [|\mathcal{T}_R^i|]$, $j' \in [|\mathcal{T}_L^i|]$,

- $\pi(V(C) \cap B_i) \subseteq J_R^{(i,j)}$ *if $\rho_i(\tau) = j$ and*
- $\pi(V(C) \cap B_i) \subseteq J_L^{(i,j')}$ *if $\lambda_i(\tau) = j'$.*

▶ **Lemma 10** ($\star$). *Given an S'-extremal ordering $\pi : V(G) \to [n]$ which is compatible with σ and $\mathcal{B}$, there exists an S'-extremal ordering $\pi' : V(G) \to [n]$ of* $\text{stretch}(\pi') \leq \text{stretch}(\pi)$ *which is compatible with σ and $\mathcal{B}$ and has properties (Π_1), (Π_2) and (Π_3).*

3.3 ILP formulation

Let G be a graph and S a cluster deletion set of G. Furthermore, let $\mathcal{C}$ be a representative set of clusters, $S' = S \cup \bigcup_{C\in\mathcal{C}} V(C)$ the extended deletion set containing vertices from $\mathcal{C}$ and S and $k' = |S'|$.

For every ordering $\sigma : S' \to k'$, we will use an ILP to determine whether there is an S'-extremal ordering $\pi : V(G) \to [n]$ of stretch at most b which is compatible with σ. The ILP has two variables x_τ, y_τ for every distribution-type $\tau \in \mathcal{T}$. The variable x_τ expresses how many clusters of $G - S'$ have distribution-type τ in an optimal S'-extremal ordering compatible with σ. The variable y_τ is an indicator variable which is 1 if and only if $x_\tau > 0$ and 0 otherwise. We further use z_i for $i \in [0, k']$ in our ILP formulation as a placeholder for the expression $\sum_{\tau\in\mathcal{T}}(x_\tau \cdot \|\tau_i\|_1)$ which expresses the number of vertices in bucket i. For an assignment $\alpha : \{x_\tau, y_\tau : \tau \in \mathcal{T}\} \to \mathbb{N}$ of the variables of our ILP, we write $\alpha(z_i)$ to stand for the expression $\sum_{\tau\in\mathcal{T}}(\alpha(x_\tau) \cdot \|\tau_i\|_1)$.

We further need the leftmost and rightmost neighbor of any vertex of S in S', thus define $v^s_{\min,\sigma}, v^s_{\max,\sigma} \in S'$ such that $\sigma(v^s_{\min,\sigma}) = \min(\sigma(N(s)))$ and $\sigma(v^s_{\max,\sigma}) = \max(\sigma(N(s)))$, for every ordering $\sigma : S' \to k'$ and $s \in S$. Note that by choosing S to be minimum, we can assume that S contains no vertex with no neighbors in $G - S$ and hence $v^s_{\min,\sigma}$ and $v^s_{\max,\sigma}$ are well defined.

For a fixed ordering $\sigma : S' \to [k']$, we can now formulate our set of linear constraints. The first three constraints ensure that we choose the number of clusters that have a certain distribution-type in a feasible way. That is, (T1) ensures that the quantities of distribution-types corresponding to an assignment of the variables x_τ corresponds to a valid choice of allocating each available cluster in the input graph G a distribution-type. As for (T2), it ensures that $v^s_{\min,\sigma}$ is indeed the leftmost neighbor of s while $v^s_{\max,\sigma}$ is the rightmost neighbor of s for every $s \in S$ by ensuring that any distribution-type placing a neighbor of s in a bucket left of $v^s_{\min,\sigma}$ or right of $v^s_{\max,\sigma}$ does not occur. Finally, (T3) guarantees that y_τ indeed indicates whether or not distribution-type τ is used in the solution.

(T1) For every $\kappa \in \mathcal{K}$,

$$\#\kappa = \min\{\#\kappa, 2k\} + \sum_{\tau \in \mathcal{T}_\kappa} x_\tau.$$

(T2) For every $s \in S$ and every $\tau \in \mathcal{T}$ for which $\tau_{N,i} > 0$ for some $N \ni s$ and $i \in [0, \sigma(v^s_{\min,\sigma}) - 1] \cup [\sigma(v^s_{\max,\sigma}), k']$,

$$x_\tau = 0.$$

(T3) $x_\tau \cdot (1 - y_\tau) = 0$ and $(1 - x_\tau) \cdot y_\tau \leq 0$ for every $\tau \in \mathcal{T}$.

The purpose of all remaining constraints is to ensure that for the assignment of variables, which essentially corresponds to choosing a bucket distribution $\mathcal{B}$, there is an S'-extremal ordering $\pi : V(G) \to [n]$ which is compatible with σ and $\mathcal{B}$ for which $\mathrm{stretch}(\pi) \leq b$. (DS) expresses that the stretch of edges in $G[S']$ is bounded by b.

(DS) For every $s, s' \in S'$ with $\{s, s'\} \in E(G)$, $\sigma(s) < \sigma(s')$,

$$b \geq \sigma(s') - \sigma(s) + \sum_{i \in [\sigma(s), \sigma(s')-1]} z_i.$$

The last three constraints deal with bounding the stretch of edges within clusters. For this we assume that the S'-extremal ordering which is consistent with σ and $\mathcal{B}$ is nice, i.e. has properties (Π_1), (Π_2) and (Π_3). The first constraint (C1) is necessary to bound the stretch of clusters that are fully contained in one bucket. To bound the stretch of clusters contained in multiple buckets, we have one constraint for every distribution-type $\tau \in \mathcal{T}^{(\ell,r,n_L,n_R)}$ for any $\ell < r \in [0, k']$, $n_L, n_R \in [\omega(G)]$. By property (Π_3) we know that there are intervals $J_L^{(\ell,\lambda_\ell(\tau))}$ containing all vertices from $B_\ell \cap V(C)$ and $J_R^{(r,\rho_r(\tau))}$ containing all vertices $B_r \cap V(C)$ for every cluster C of distribution-type τ. The trick now is to observe that if $n_L \geq n_R$ then the first cluster appearing in $J_L^{(\ell,\lambda_\ell(\tau))}$ observes the maximum stretch while if $n_L < n_R$ it is the last clique. Using this we can express with constraints (C2) and (C3) that the stretch of every cluster of distribution-type τ is bounded by b.

(C1) ($b \geq \omega(G) - 1$.

(C2) For every $\ell < r \in [0, k']$, $n_L \geq n_R \in [\omega(G)]$ and $\tau \in \mathcal{T}^{(\ell,r,n_L,n_R)}$,

$$b \geq y_\tau \cdot \Bigg(\sum_{\tau' \in \lambda_\ell^{-1}\left([\lambda_\ell(\tau), |\mathcal{T}_L^\ell|]\right)} \#\mathrm{L}(\tau') \cdot x_{\tau'} + \sum_{\ell < i < r} z_i + (r - \ell) + \sum_{\tau' \in \rho_r^{-1}\left([1, \rho_r(\tau) - 1]\right)} \#\mathrm{R}(\tau') \cdot x_{\tau'} + n_R - 1 \Bigg).$$

(C3) For every $\ell < r \in [0, k']$, $n_L < n_R \in [\omega(G)]$ and $\tau \in \mathcal{T}^{(\ell,r,n_L,n_R)}$,

$$b \geq y_\tau \cdot \Bigg(n_L + \sum_{\tau' \in \lambda_\ell^{-1}\left([\lambda_\ell(\tau)+1, |\mathcal{T}_L^\ell|]\right)} \#\mathrm{L}(\tau') \cdot x_{\tau'} + \sum_{\ell < i < r} z_i + (r - \ell) + \sum_{\tau' \in \rho_r^{-1}\left([1, \rho_r(\tau)]\right)} \#\mathrm{R}(\tau') \cdot x_{\tau'} - 1 \Bigg).$$

▶ **Lemma 11** (⋆). *For any ordering $\sigma : S' \to [k']$, there is an S'-extremal ordering $\pi : V(G) \to [n]$ of stretch at most b which is compatible with σ if and only if the system of linear equation (T1, T2, T3, DS, C1, C2, C3) for σ admits a solution.*

Using Lemma 11 we obtain an FPT-algorithm, which computes S, $\#\kappa$ for every cluster-type κ and arbitrary picks an extended deletion set S'. The algorithm then for every ordering $\sigma : S' \to [k']$ verifies whether the ILP admits a solution in which case the input is a YES-instance of Bandwidth. Details are given in the full version of the paper.

▶ Remark 12. Using a minimization ILP, we can in fact *construct* an ordering of *minimum* stretch (and not just argue about the existence of an ordering of stretch at most b), since all the exchange arguments of Section 3.2 are constructive.

4 W[1]-hardness parameterized by cluster vertex deletion number

In this section, we prove that Bandwidth is W[1]-hard when parameterized by the cluster vertex deletion number of the input graph. In order to do so, we reduce from an instance of Unary Bin Packing. Before we present the details of the construction, we first give some high-level intuition.

For an instance (A, k) of Unary Bin Packing we want to construct an equivalent instance (G, b) of Bandwidth, such that $\text{cvd}(G) = f(k)$ for some function f. Roughly, the graph G consists of cliques representing the items of the Unary Bin Packing instance and cliques that act as delimiters separating the items contained in some bucket from the items contained in the next bucket. However, in order to guarantee that the entirety of every item clique is placed in between two consecutive delimiter cliques and that the values of the items in between two delimiter cliques add up to B (the capacity of the bins in the Unary Bin Packing instance (A, k)), some extra structure is needed. First we introduce two cliques of size $b + 1$ that will be used as boundaries. By making each item clique and each delimiter clique of the graph adjacent to some vertex in both of the boundary cliques, it follows that in any ordering of stretch at most b, all item cliques and all delimiter cliques of the graph will be positioned in between the two boundary cliques.

As the size of the deletion set cannot depend on the number or values of the items, item cliques cannot be incident to individual deletion set vertices. This makes it tricky to enforce that every vertex of an item clique is contained in between the same two delimiter cliques as a majority of the item cliques would not be incident to any edge of maximum stretch and therefore allow them a lot of freedom of movement. In order to cope with this issue, we introduce a perfect copy of the delimiter and item cliques, as well as edges between the original cliques and their copies resulting in them becoming twice as big consisting of a left part, the original vertices, and a right part, the copy vertices. The left part of all cliques will be connected to the left boundary clique and will therefore appear to the left of the right parts. The right part will be connected to the right hand boundary cliques. The item cliques will now be kept in place by having maximum stretch between the vertices of the left part and the vertices of the right part.

▶ **Theorem 13.** *Bandwidth is W[1]-hard when parameterized by the cluster deletion number of the input graph.*

Construction. Let (A, k) be an instance of Unary Bin Packing, where $A = \{a_1, \dots, a_n\}$. Moreover, let $B = \sum_{j \in [n]} a_j / k$ be the capacity of every bin, where $B \in \mathbb{N}$, since otherwise this would have been a trivial instance. Set $b = 2kB + B - 1$. We will construct an equivalent instance (G, b) of Bandwidth as follows.

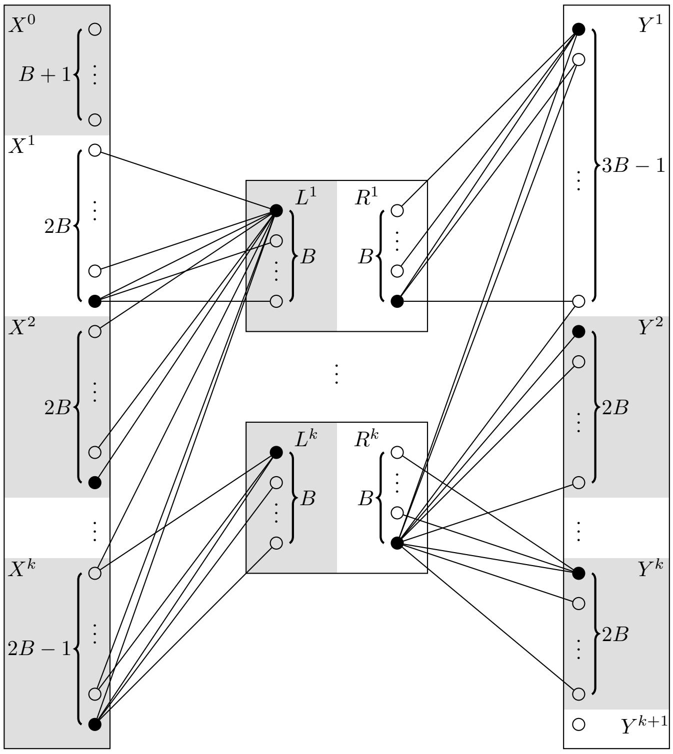

Figure 2 Part of G, showing only the boundary and the delimiter cliques. Rectangles denote cliques, brackets denote number of vertices and black vertices compose a cluster deletion set.

Boundary cliques. First, we create two cliques X and Y, referred to as *boundary cliques*, where $V(X) = \{x_1, \dots, x_{2kB+B}\}$ and $V(Y) = \{y_1, \dots, y_{2kB+B}\}$. We consider the following partition of the vertices of X: let $X^0 = \{x_1, \dots, x_{B+1}\}$ and for every $i \in [k-1]$ we denote the set $\{x_{2iB-B+2}, \dots, x_{2iB+B+1}\}$ by X^i, while $X^k = \{x_{2kB-B+2}, \dots, x_{2kB+B}\}$. Note that $|X^0| = B+1$, $|X^k| = 2B-1$ and $|X^i| = 2B$, for all $i \in [k-1]$. Moreover, we partition the vertices of Y in a similar but slightly asymmetric way: let $Y^1 = \{y_1, \dots, y_{3B-1}\}$ and for every $i \in [2,k]$ we denote the set $\{y_{2iB-B}, \dots, y_{2iB+B-1}\}$ by Y^i, while $Y^{k+1} = \{y_{2kB+B}\}$. Note that $|Y^1| = 3B-1$, $|Y^{k+1}| = 1$ and $|Y^i| = 2B$, for all $i \in [2,k]$.

Delimiter cliques. For every $i \in [k]$ we create a clique on vertex set $\{\ell^i_1, \dots, \ell^i_B, r^i_1, \dots, r^i_B\}$ of size $2B$. We denote the set $\{\ell^i_1, \dots, \ell^i_B\}$ by L^i and the set $\{r^i_1, \dots, r^i_B\}$ by R^i. Moreover, let $L = \bigcup_{i=1}^k L^i$ and $R = \bigcup_{i=1}^k R^i$. We add the following edges:

- For every $i \in [k]$, $x \in \bigcup_{j=i}^k X^j$, we add the edge $\{\ell^i_1, x\}$.
- For every $i \in [k-1]$, $\ell \in L^i$, we add the edge $\{x_{2iB+B+1}, \ell\}$. Moreover, we add an edge between x_{2kB+B} and every vertex of L^k.
- For every $i \in [k]$, $y \in \bigcup_{j=1}^i Y^j$, we add the edge $\{r^i_B, y\}$.
- For every $i \in [2,k]$, $r \in R^i$, we add the edge $\{y_{2iB-B}, r\}$. Moreover, we add an edge between y_1 and every vertex of R^1.

For an illustration of the boundary and delimiter cliques, see Figure 2.

Item cliques. For element $a_i \in A$, we construct a clique A^i on vertex set $\{a^{i,L}_j, a^{i,R}_j : j \in [a_i]\}$ of size $2a_i$. We denote the set of vertices $\{a^{i,L}_j : j \in [a_i]\}$ by $A^{i,L}$ and the set of vertices $\{a^{i,R}_j : j \in [a_i]\}$ by $A^{i,R}$. We add edges $\{x_{2kB+B}, a\}$ for every $a \in \bigcup_{i\in[k]} A^{i,L}$ and edges $\{y_1, a\}$ for every $a \in \bigcup_{i\in[k]} A^{i,R}$. For an illustration, see Figure 3.

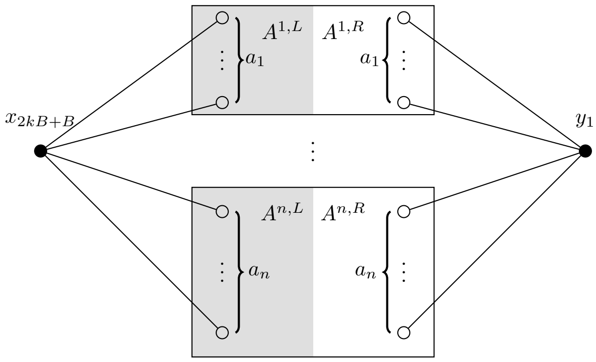

Figure 3 Rectangles denote cliques. Black vertices compose a cluster deletion set.

This concludes the construction of G. Figure 4 illustrates an example of an ordering of stretch b obtained by a YES-instance of Unary Bin Packing. In the following, we prove the equivalence of (G, b) to the initial instance of Unary Bin Packing.

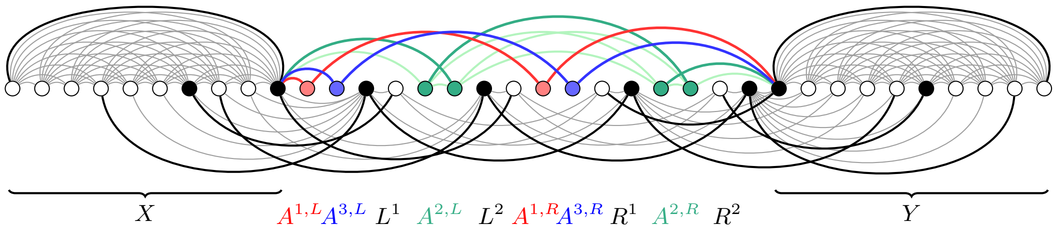

Figure 4 For the instance $(\{a_1, a_2, a_3\}, 2)$ of Unary Bin Packing with $a_1 = 1$, $a_2 = 2$ and $a_3 = 1$ the figure shows the graph G from the corresponding instance $(G, 9)$ of Bandwidth. Here the ordering of the vertices of G with stretch 9 corresponds to the solution of $(\{a_1, a_2, a_3\}, 2)$ in which a_1, a_3 are placed in the first bin and a_2 in the second.

▶ **Lemma 14** ($\star$). *If (A, k) is a YES-instance of Unary Bin Packing, then (G, b) is a YES-instance of Bandwidth.*

▶ **Lemma 15** ($\star$). *If (G, b) is a YES-instance of Bandwidth, then (A, k) is a YES-instance of Unary Bin Packing.*

▶ **Lemma 16** ($\star$). *It holds that* $\text{cvd}(G) = \mathcal{O}(k)$.

5 Conclusion

In the current work, we extend our understanding of Bandwidth in the setting of parameterized complexity. In particular, we have shown that the problem is FPT when parameterized by the cluster vertex deletion number cvd plus the clique number ω of the input graph, although it becomes W[1]-hard when parameterized only by cvd.

The most natural research direction would be to explore the tractability of the problem when parameterized by twin cover, modular-width or vertex integrity, given the lack of any relevant FPT/XP algorithms or hardness results. As a matter of fact, it is not even known whether the problem is in XP when parameterized by cvd or treedepth.

Finally, most tractability results for the various structural parameters rely on some ILP formulation. This raises the question of whether any other kind of approach is applicable, as is the case for Cutwidth [10].

References

1 Olav Røthe Bakken. Arrangement problems parameterized by neighbourhood diversity. Master's thesis, University of Bergen, 2018.

2 Aritra Banik, Prahlad Narasimhan Kasthurirangan, and Venkatesh Raman. Dominator coloring and CD coloring in almost cluster graphs. In *Algorithms and Data Structures - 18th International Symposium, WADS 2023*, volume 14079 of *Lecture Notes in Computer Science*, pages 106–119. Springer, 2023. doi:10.1007/978-3-031-38906-1_8.

3 Hans L. Bodlaender. Parameterized complexity of bandwidth of caterpillars and weighted path emulation. In *Graph-Theoretic Concepts in Computer Science - 47th International Workshop, WG 2021*, volume 12911 of *Lecture Notes in Computer Science*, pages 15–27. Springer, 2021. doi:10.1007/978-3-030-86838-3_2.

4 Hans L. Bodlaender, Michael R. Fellows, and Michael T. Hallett. Beyond np-completeness for problems of bounded width: hardness for the W hierarchy. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 449–458. ACM, 1994. doi:10.1145/195058.195229.

5 Henning Bruhn, Morgan Chopin, Felix Joos, and Oliver Schaudt. Structural parameterizations for boxicity. *Algorithmica*, 74(4):1453–1472, 2016. doi:10.1007/s00453-015-0011-0.

6 Phyllis Z. Chinn, J. Chvatalova, A. K. Dewdney, and Norman E. Gibbs. The bandwidth problem for graphs and matrices - a survey. *J. Graph Theory*, 6(3):223–254, 1982. doi:10.1002/jgt.3190060302.

7 Janka Chlebíková and Morgan Chopin. The firefighter problem: A structural analysis. In *Parameterized and Exact Computation - 9th International Symposium, IPEC 2014*, volume 8894 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 2014. doi:10.1007/978-3-319-13524-3_15.

8 Morgan Chopin, André Nichterlein, Rolf Niedermeier, and Mathias Weller. Constant thresholds can make target set selection tractable. *Theory Comput. Syst.*, 55(1):61–83, 2014. doi:10.1007/s00224-013-9499-3.

9 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. On cutwidth parameterized by vertex cover. *Algorithmica*, 68(4):940–953, 2014. doi:10.1007/s00453-012-9707-6.

11 Marek Cygan and Marcin Pilipczuk. Exact and approximate bandwidth. *Theor. Comput. Sci.*, 411(40–42):3701–3713, 2010. doi:10.1016/j.tcs.2010.06.018.

12 Marek Cygan and Marcin Pilipczuk. Bandwidth and distortion revisited. *Discret. Appl. Math.*, 160(4-5):494–504, 2012. doi:10.1016/j.dam.2011.10.032.

13 Marek Cygan and Marcin Pilipczuk. Even faster exact bandwidth. *ACM Trans. Algorithms*, 8(1):8:1–8:14, 2012. doi:10.1145/2071379.2071387.

14 Josep Díaz, Jordi Petit, and Maria J. Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, 2002. doi:10.1145/568522.568523.

15 Reinhard Diestel. *Graph Theory*, volume 173 of *Graduate texts in mathematics*. Springer, 2017. doi:10.1007/978-3-662-53622-3.

16 Martin Doucha and Jan Kratochvíl. Cluster vertex deletion: A parameterization between vertex cover and clique-width. In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012*, volume 7464 of *Lecture Notes in Computer Science*, pages 348–359. Springer, 2012. doi:10.1007/978-3-642-32589-2_32.

17 Markus Sortland Dregi and Daniel Lokshtanov. Parameterized complexity of bandwidth on trees. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014*, volume 8572 of *Lecture Notes in Computer Science*, pages 405–416. Springer, 2014. doi:10.1007/978-3-662-43948-7_34.

18 Michael R. Fellows, Danny Hermelin, Frances A. Rosamond, and Hadas Shachnai. Tractable parameterizations for the minimum linear arrangement problem. *ACM Trans. Comput. Theory*, 8(2):6:1–6:12, 2016. doi:10.1145/2898352.

19 Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Matthias Mnich, Frances A. Rosamond, and Saket Saurabh. The complexity ecology of parameters: An illustration using bounded max leaf number. *Theory Comput. Syst.*, 45(4):822–848, 2009. doi:10.1007/s00224-009-9167-9.

20 Michael R. Fellows, Daniel Lokshtanov, Neeldhara Misra, Frances A. Rosamond, and Saket Saurabh. Graph layout problems parameterized by vertex cover. In *Algorithms and Computation, 19th International Symposium, ISAAC 2008*, volume 5369 of *Lecture Notes in Computer Science*, pages 294–305. Springer, 2008. doi:10.1007/978-3-540-92182-0_28.

21 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7:49–65, 1987. doi:10.1007/BF02579200.

22 Martin Fürer, Serge Gaspers, and Shiva Prasad Kasiviswanathan. An exponential time 2-approximation algorithm for bandwidth. *Theor. Comput. Sci.*, 511:23–31, 2013. doi:10.1016/j.tcs.2013.03.024.

23 Jakub Gajarský, Michael Lampis, and Sebastian Ordyniak. Parameterized algorithms for modular-width. In *Parameterized and Exact Computation - 8th International Symposium, IPEC 2013*, volume 8246 of *Lecture Notes in Computer Science*, pages 163–176. Springer, 2013. doi:10.1007/978-3-319-03898-8_15.

24 Robert Ganian. Twin-cover: Beyond vertex cover in parameterized algorithmics. In *Parameterized and Exact Computation - 6th International Symposium, IPEC 2011*, volume 7112 of *Lecture Notes in Computer Science*, pages 259–271. Springer, 2011. doi:10.1007/978-3-642-28050-4_21.

25 Robert Ganian, Petr Hlinený, Jaroslav Nesetril, Jan Obdrzálek, Patrice Ossona de Mendez, and Reshma Ramadurai. When trees grow low: Shrubs and fast MSO1. In *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012*, volume 7464 of *Lecture Notes in Computer Science*, pages 419–430. Springer, 2012. doi:10.1007/978-3-642-32589-2_38.

26 Robert Ganian, Eun Jung Kim, and Stefan Szeider. Algorithmic applications of tree-cut width. *SIAM J. Discret. Math.*, 36(4):2635–2666, 2022. doi:10.1137/20m137478x.

27 M. R. Garey, R. L. Graham, D. S. Johnson, and D. E. Knuth. Complexity results for bandwidth minimization. *SIAM Journal on Applied Mathematics*, 34(3):477–495, 1978. doi:10.1137/0134037.

28 Tatsuya Gima, Tesshu Hanaka, Masashi Kiyomi, Yasuaki Kobayashi, and Yota Otachi. Exploring the gap between treedepth and vertex cover through vertex integrity. *Theor. Comput. Sci.*, 918:60–76, 2022. doi:10.1016/j.tcs.2022.03.021.

29 Eitan M. Gurari and Ivan Hal Sudborough. Improved dynamic programming algorithms for bandwidth minimization and the mincut linear arrangement problem. *J. Algorithms*, 5(4):531–546, 1984. doi:10.1016/0196-6774(84)90006-3.

30 L. H. Harper. Optimal assignments of numbers to vertices. *Journal of the Society for Industrial and Applied Mathematics*, 12(1):131–135, 1964. doi:10.1137/0112012.

31 Klaus Jansen, Stefan Kratsch, Dániel Marx, and Ildikó Schlotter. Bin packing with fixed number of bins revisited. *J. Comput. Syst. Sci.*, 79(1):39–49, 2013. doi:10.1016/j.jcss.2012.04.004.

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

33 Anjeneya Swami Kare and I. Vinod Reddy. Parameterized algorithms for graph burning problem. In *Combinatorial Algorithms - 30th International Workshop, IWOCA 2019*, volume 11638 of *Lecture Notes in Computer Science*, pages 304–314. Springer, 2019. doi:10.1007/978-3-030-25005-8_25.

34 Martin Kucera and Ondrej Suchý. Minimum eccentricity shortest path problem with respect to structural parameters. *Algorithmica*, 85(3):762–782, 2023. doi:10.1007/s00453-022-01006-x.

35 Michael Lampis. Algorithmic meta-theorems for restrictions of treewidth. In *Algorithms - ESA 2010, 18th Annual European Symposium*, volume 6346 of *Lecture Notes in Computer Science*, pages 549–560. Springer, 2010. doi:10.1007/978-3-642-15775-2_47.

36 Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983. doi:10.1287/moor.8.4.538.

37 Daniel Lokshtanov. Parameterized integer quadratic programming: Variables and coefficients. *CoRR*, abs/1511.00310, 2015. arXiv:1511.00310.

38 Diptapriyo Majumdar and Venkatesh Raman. FPT algorithms for FVS parameterized by split and cluster vertex deletion sets and other parameters. In *Frontiers in Algorithmics - 11th International Workshop, FAW 2017*, volume 10336 of *Lecture Notes in Computer Science*, pages 209–220. Springer, 2017. doi:10.1007/978-3-319-59605-1_19.

39 Neeldhara Misra and Harshil Mittal. Imbalance parameterized by twin cover revisited. *Theor. Comput. Sci.*, 895:1–15, 2021. doi:10.1016/j.tcs.2021.09.017.

40 Burkhard Monien. The bandwidth minimization problem for caterpillars with hair length 3 is np-complete. *SIAM Journal on Algebraic Discrete Methods*, 7(4):505–512, 1986. doi:10.1137/0607057.

41 David Muradian. The bandwidth minimization problem for cyclic caterpillars with hair length 1 is np-complete. *Theor. Comput. Sci.*, 307(3):567–572, 2003. doi:10.1016/S0304-3975(03)00238-X.

42 Jaroslav Nesetril and Patrice Ossona de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27(6):1022–1041, 2006. doi:10.1016/j.ejc.2005.01.010.

43 Christos H. Papadimitriou. The np-completeness of the bandwidth minimization problem. *Computing*, 16(3):263–270, 1976. doi:10.1007/BF02280884.

44 James B. Saxe. Dynamic-programming algorithms for recognizing small-bandwidth graphs in polynomial time. *SIAM J. Algebraic Discret. Methods*, 1(4):363–369, 1980. doi:10.1137/0601042.