Quadratic Kernel for Cliques or Trees Vertex Deletion

Soh Kumabe
CyberAgent, Tokyo, Japan

Abstract

We consider Cliques or Trees Vertex Deletion, which is a hybrid of two fundamental parameterized problems: Cluster Vertex Deletion and Feedback Vertex Set. In this problem, we are given an undirected graph G and an integer k, and asked to find a vertex subset X of size at most k such that each connected component of $G - X$ is either a clique or a tree. Jacob et al. (ISAAC, 2024) provided a kernel of $O(k^5)$ vertices for this problem, which was recently improved to $O(k^4)$ by Tsur (IPL, 2025).

Our main result is a kernel of $O(k^2)$ vertices. This result closes the gap between the kernelization result for Feedback Vertex Set, which corresponds to the case where each connected component of $G - X$ must be a tree.

Although both *cluster vertex deletion number* and *feedback vertex set number* are well-studied structural parameters, little attention has been given to parameters that generalize both of them. In fact, the lowest common well-known generalization of them is clique-width, which is a highly general parameter. To fill the gap here, we initiate the study of the *cliques or trees vertex deletion number* as a structural parameter. We prove that Longest Cycle, which is a fundamental problem that does not admit $o(n^k)$-time algorithm unless ETH fails when k is the clique-width, becomes fixed-parameter tractable when parameterized by the cliques or trees vertex deletion number.

2012 ACM Subject Classification Theory of computation → Fixed parameter tractability

Keywords and phrases Fixed-Parameter Tractability, Kernelization, Deletion to Scattered Graph Classes, Cluster Vertex Deletion, Feedback Vertex Set

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2025.48

Related Version *Full Version*: https://arxiv.org/abs/2509.16815

1 Introduction

Given a graph, can we remove at most k vertices so that the resulting graph belongs to a class Π of well-structured graphs? Such problems are called *vertex deletion problems* and include many well-studied parameterized problems. Indeed, Vertex Cover [12, 23], Feedback Vertex Set [11, 25, 34, 39], Cluster Vertex Deletion [4, 15, 24], Odd Cycle Transversal [32, 38], Interval Vertex Deletion [6], and Chordal Vertex Deletion [36] correspond to the cases where Π is the classes of collection of isolated vertices, collection of trees, collection of cliques, bipartite graphs, interval graphs, and chordal graphs, respectively.

However, why must Π be a single graph class? It is still reasonable to call a graph well-structured when all connected components are well-structured, even if different components belong to different classes. Jacob et al. [26] introduced the problem framework of *deletion to scattered graph classes* capturing this concept, asking: Can we remove at most k vertices from a given graph so that each connected component of the resulting graph belongs to one of the graph classes $(\Pi_1, \ldots, \Pi_p)$? Together with the subsequent paper [27], they investigated the parameterized complexity of this type of problems and provided both general and problem-specific algorithms.

This paper considers the following fundamental special case of deletion problems to scattered graph classes, called Cliques or Trees Vertex Deletion, which is first studied in [27].

36th International Symposium on Algorithms and Computation (ISAAC 2025).
Editors: Ho-Lin Chen, Wing-Kai Hon, and Meng-Tsung Tsai; Article No. 48; pp. 48:1–48:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

CLIQUES OR TREES VERTEX DELETION: Given an undirected graph $G = (V, E)$ and an integer $k \in \mathbb{Z}_{\geq 1}$, is there a vertex subset $X \subseteq V$ with $|X| \leq k$ such that each connected component of $G - X$ is either a clique or a tree?

This case is particularly interesting because it combines two of the most prominent parameterized problems, FEEDBACK VERTEX SET and CLUSTER VERTEX DELETION. Moreover, since both the *feedback vertex set number* and the *cluster vertex deletion number* are well-studied structural parameters, it is natural to expect that the *cliques or trees vertex deletion number*, which is the minimum k such that (G, k) becomes an yes-instance of CLIQUES OR TREES VERTEX DELETION, is likewise an interesting structural parameter. In particular, this number captures the structural simplicity of graphs that contain both dense parts (i.e., clique components) and sparse parts (i.e., tree components). This illustrates the benefit of considering deletion to scattered graph classes, as a deletion distance to a single dense or sparse class alone cannot capture such a property.

It has been proved that this problem is in FPT [26, 27]; that is, it admits an $f(k)n^c$-time algorithm for some computable function f and constant c. Accordingly, researchers are investigating kernelization algorithms, which are polynomial-time algorithms that reduce the given instance to an equivalent instance of size $g(k)$ for some function g. Jacob et al. [28] showed that this problem admits a kernel with $O(k^5)$ vertices, which was later improved to $O(k^4)$ vertices by Tsur [40]. However, there is still a significant gap when compared to the kernelization results of the original two problems, FEEDBACK VERTEX SET and CLUSTER VERTEX DELETION, which admit a kernel with $O(k^2)$ vertices [25, 39] and $O(k)$ vertices [4], respectively.

In this paper, we close this gap by proving the following.

► **Theorem 1.** *CLIQUES OR TREES VERTEX DELETION admits a kernel with $O(k^2)$ vertices.*

We remark that our result may surpass the expectations in the original research by Jacob et al. [28], as they stated the following in their conclusion section:

> One natural open question is to improve the size of our kernel, e.g. to $O(k^3)$ vertices. We believe that such a result is possible to achieve, but we suspect that it would require new techniques to develop such results.

We further initiate the study of CLIQUES OR TREES VERTEX DELETION as a structural parameter. As mentioned above, both the feedback vertex set number [18, 41] and the cluster vertex deletion number [10, 20] are well-studied structural parameters. However, their common generalization has received relatively little attention. Actually, the smallest well-investigated class that includes both graphs with bounded feedback vertex set number and graphs with bounded cluster vertex deletion number is the class of bounded *clique-width*, which is a very general parameter often placed at the top of diagrams illustrating the inclusion relationships among structural parameters of graphs (see Figure 1).

As the second contribution of this paper, we demonstrate that the cliques or trees vertex deletion number is indeed a useful structural parameter. Particularly, we demonstrate that when parameterized by the cliques or trees vertex deletion number, a fundamental problem that is hard when parameterized by clique-width becomes tractable. Specifically, we consider the following LONGEST CYCLE, one of the most well-investigated problems in the field of parameterized complexity under structural parameterizations [3, 8, 9, 10, 13, 14, 19, 21, 29, 33].

LONGEST CYCLE: Given an undirected graph $G = (V, E)$, find a cycle of G with the largest possible number of vertices.

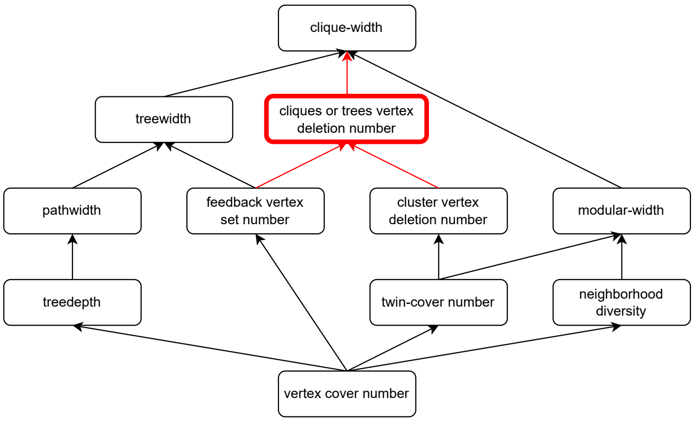

Figure 1 Diagram of structural parameters including the cliques or trees vertex deletion number.

It is known that, when k is the clique-width, Longest Cycle (and even the special case Hamiltonian Cycle) does not admit an $n^{o(k)}$-time algorithm unless ETH fails [13, 14]. We prove the following.

▶ **Theorem 2.** *Longest Cycle can be solved in $2^{O(k \log k)}$-time, where k is the cliques or trees vertex deletion number.*

Note that this result generalizes, without losing time complexity, the FPT algorithm for Hamiltonian Cycle parameterized by the cluster vertex deletion number given by Doucha and Kratochvíl [10] in the following two aspects. First, we solve Longest Cycle, which is a generalization of Hamiltonian Cycle. Second, our algorithm is parameterized by cliques or trees vertex deletion number, which is a more general parameter. We further note that it is not hard to see that the clique-width of graphs with bounded cliques or trees vertex deletion number is bounded.

1.1 Related Work

The literature on Π-deletion problems has studied several variants of Feedback Vertex Set and Cluster Vertex Deletion. Examples of extensions of Feedback Vertex Set are the cases where Π consists of graphs with treewidth at most η [16, 31] and graphs that are at most l edges away from forests [37]. Examples of variants of Cluster Vertex Deletion are the cases where Π consists of s-plexes [35], s-clubs [22], and graphs with a small dominating set number [2].

Jacob et al. [26] introduced the notion of *deletion to scattered graph classes*. In that paper, they obtained two general tractability results, stating that $(\Pi_1, \ldots, \Pi_p)$-Vertex Deletion (i) is fixed-parameter tractable when each Π_i-Vertex Deletion is fixed-parameter tractable, and (ii) admits a $2^{poly(k)}$-time algorithm when each Π_i is defined by a finite set of forbidden subgraphs. Jansen et al. [30] improved the time complexity of result (ii) to $2^{O(k)}$. In the subsequent work [27], Jacob et al. considered several specific cases and obtained efficient FPT algorithms and approximation algorithms. In particular, for Cliques or Trees Vertex Deletion, they obtained an $O^*(4^k)$-time FPT algorithm and a polynomial-time 4-approximation algorithm. Jacob et al. [28] provided a kernel with $O(k^5)$ vertices for this problem. Tsur [40] improved both of FPT and kernelization results by presenting a deterministic algorithm running in $O(3.46^k)$ time, a randomized algorithm running in $O(3.103^k)$ time, and a kernel with $O(k^4)$ vertices.

Several studies have investigated the parameterized complexity of LONGEST CYCLE and its special case, HAMILTONIAN CYCLE, under structural parameterizations. Parameterized by the cluster vertex deletion number, Doucha and Kratochvíl [10] presented an FPT algorithm for HAMILTONIAN CYCLE. Parameterized by the clique-width, Fomin et al. [13, 14] showed that no $n^{o(k)}$-time algorithm for HAMILTONIAN CYCLE exists unless ETH fails. Bergougnoux et al. [3] proved this bound is tight by giving an $n^{O(k)}$-time algorithm. The cases parameterized by treewidth [9], pathwidth [8], twin-cover number [19], and proper interval deletion number [21] are also investigated, as well as directed tree-width [33] and directed feedback vertex set number [29] for the directed version.

1.2 Technical Overview

Here, we provide a technical overview of Theorems 1 and 2. Section 1.2.1 also contains a brief explanation of how our kernelization algorithm differs from the existing kernelization algorithms by Jacob et al. [28] and Tsur [40].

1.2.1 Overview of Theorem 1

Here, we briefly explain our idea toward Theorem 1. Our approach, at the highest level, is based on the following simple observation. Let v be a vertex, and let $N_G(v) := \{u \in V : (u, v) \in E\}$ be the set of neighbors of v. Assume there is a feasible solution[1] X such that v is in a connected component of $G-X$ that is a clique. Then, the neighbors of v in $G-X$ induce a clique. In particular, $N_G(v)$ contains a clique of size at least $|N_G(v) \setminus X| \geq |N_G(v)| - k$. Similarly, assume there is a feasible solution X such that v is in a connected component of $G - X$ that is a tree. Then, the neighbors of v in $G - X$ induce an independent set. In particular, $N_G(v)$ contains an independent set of size at least $|N_G(v) \setminus X| \geq |N_G(v)| - k$.

The core observation is that if $|N_G(v)|$ is large, say at least $2k + 2$, these two situations cannot occur simultaneously, as a clique and an independent set cannot share an edge. In other words, if the degree of v is large, we can determine that v is either

- always in a clique component of $G - X$ unless $v \in X$, or
- always in a tree component of $G - X$ unless $v \in X$,

for all feasible solutions X.

In our algorithm, we first partition the vertices into three subsets. Let $c = 7$. Set V_{ld} of vertices with a high degree ($> ck$) and dense neighbors, set V_{ls} of vertices with a high degree ($> ck$) and sparse neighbors, and set V_{small} of vertices with a low degree ($\leq ck$). Using the above observation, we can claim that, for any feasible solution X, vertices in V_{ld} cannot be in a tree component of $G - X$, and vertices in V_{ls} cannot be in a clique component of G of $G - X$. This observation, very roughly speaking, enables us to apply reduction rules to V_{ld} as if we were solving CLUSTER VERTEX DELETION. Similarly, we can apply reduction rules to V_{ls} as if we were solving FEEDBACK VERTEX SET. Furthermore, we can reduce the number of vertices in V_{small} using the fact that they have small degrees.

Here we go into a little more detail. Our kernelization algorithm proceeds as follows. First, we reduce the degrees of vertices in V_{ls} into $O(k)$. To acheive this, we directly apply reduction rules used in the celebrated quadratic kernelization of FEEDBACK VERTEX SET by Thomassé [39]. Note that this direct applicability is already a benefit of partitioning the vertex set. Indeed, the original kernelization algorithm by Jacob et al. [28] also relied

[1] We say a solution X is *feasible* if $|X| \leq k$ and each connected component of $G - X$ is either a clique or a tree.

on Thomassé's quadratic kernelization in its "tree part" (and Tsur's improvement [40] did not touch this part), but extended the algorithm and analysis, including the use of a new version of the expansion lemma by Fomin et al. [15]. In contrast, ours is almost identical to Thomassé's, and its analysis is also nearly the same except for a few technical details, making our approach significantly simpler.

Next, we reduce the number of vertices in V_{ld}. The main technical contribution of this paper lies in this part, since the kernelization algorithms by Jacob et al. [28] and Tsur [40] bound the number of vertices in the "clique part" by $O(k^5)$ and $O(k^4)$, respectively, which dominates the overall kernel size. Both of those algorithms are based on a marking procedure, whereas ours is not. As in those algorithms, we first compute a constant factor approximate solution S in polynomial time and reduce the number of clique components of $G - S$ by $O(k)$ using the textbook reduction [17] for Cluster Vertex Deletion using the expansion lemma. Now, we can argue that there are only $O(k^2)$ vertices in V_{ld} that are adjacent to some vertex outside V_{ld} as follows. First, any such vertex must

(i) belongs to S,

(ii) be adjacent to a vertex in $S \setminus V_{\mathrm{ld}}$, or

(iii) belongs to a clique component of $G - S$ that contains a vertex outside V_{ld}.

The number of vertices satisfying (i) is $O(k)$ by definition. Since every vertex outside V_{ld} has degree $O(k)$, the number of vertices satisfying (ii) is bounded by $O(k^2)$. Furthermore, each clique component that appears in (iii) contains at most $O(k)$ vertices because it contains a vertex with degree $O(k)$. Together with the fact that the number of clique components is $O(k)$, the number of vertices satisfying (iii) is bounded by $O(k^2)$. Now, we concentrate on reducing the number of vertices in V_{ld} that are adjacent only to vertices in V_{ld} by $O(k^2)$. To do this, we first borrow ideas from the quadratic kernelization of the 3-Hitting Set by Abu-Khzam [1] and construct a list $\mathcal{P}$ of induced P_3s (that are, induced subgraphs that are paths of length two) such that no two different P_3 share more than one vertex. A standard argument similar to that used in Buss Kernel [5] for Vertex Cover reduces the size of $\mathcal{P}$ to $O(k^2)$. Let $V_{\mathrm{ldmod}} := V_{\mathrm{ld}} \setminus \bigcup_{P \in \mathcal{P}} P$. By introducing additional structural observations, we can state that

- there are at most $O(k)$ connected components in V_{ldmod}, and
- each connected component C of V_{ldmod} is a *clique-module* of V_{ld}, that is, the set $N_{V_{\mathrm{ld}}}(v) \cup \{v\}$ is same for all $v \in C$ and includes C.

The important observation is that if G contains a clique-module of V of size at least $k + 4$, then one of its vertices can be safely removed. This rule seems to bound the size of each clique-module by $O(k)$ and bound the size of V_{ldmod} by $O(k^2)$. However, it is still insufficient because we want clique-modules of V for the reduction, whereas the observation above only provides clique-modules of V_{ld}. Nevertheless, if a clique-module in V_{ld} consists only of vertices with no neighbor outside V_{ld}, then it is also a clique-module of V. Therefore, we can reduce the number of vertices in each clique-module that have no neighbor outside V_{ld} to $O(k)$, obtaining the desired bound. (Precisely speaking, we need to perform a slightly more careful argument to deal with multi-edges that may be introduced in reductions for V_{ls}.)

We reduce the number of vertices in $V \setminus V_{\mathrm{ld}}$ to $O(k^2)$ to complete the kernelization. By an argument similar to that for (iii) above, we can bound the number of vertices in $V \setminus V_{\mathrm{ld}}$ that belong to clique components of $G - S$ by $O(k^2)$. Therefore, it remains to bound the number of vertices that belong to tree components. To do this, we apply a few reduction rules based on local structure. Most of these rules appear in [28], but one is new. We apply the following standard argument used in the kernelization of Feedback Vertex Set: If a graph with minimum degree at least 3 can be turned into a forest by removing k vertices of degree at most t, then the number of vertices in the original graph was $O(tk)$. However, in this case, we cannot completely eliminate vertices of degree 1 or 2, so this argument cannot be applied directly. Nevertheless, we can bound the number of such low-degree vertices using the numbers of vertices of degree 3 and at least 4. By extending the argument for Feedback Vertex Set using these fine-grained bounds, we can bound the number of vertices in $V \setminus V_{\mathrm{ld}}$ by $O(k^2)$, which completes the analysis.

1.3 Overview of Theorem 2

Here, we briefly explain our ideas toward Theorem 2. Let $G = (V, E)$ be a graph and $X \subseteq V$ be a given vertex subset with $|X| = k$ such that each connected component of $G - X$ is either a clique or a tree. We begin by brute-force the order in which the desired cycle P visits the vertices in X. If P is disjoint from X, the problem is trivial, so we may assume that P intersects X, and denote the vertices in $X \cap P$ by $v_1, \ldots, v_l$ in the order they appear along P. Let P_i be the v_i, v_{i+1}-path appearing on P (indices modulo l). Then, unless P_i has length 1, the internal vertices of P_i belong to a single connected component C_i of $G - X$.

The next step is limiting the candidates of C_i. Here, we can state that, as candidates of C_i, it is sufficient to consider the top k components that admit the longest v_i, v_{i+1}-paths. Then, we can brute-force over all tuples $(C_1, \ldots, C_l)$ within a total cost of $2^{O(k \log k)}$. Now, the problem is reduced to solving the following problem for each connected component H of $G - X$, where we set $J_H := \{i \colon C_i = H\}$.

> Given a list of vertex subset pairs $\{(V_{i1} := N(v_i) \cap C_i,\ V_{i2} := N(v_{i+1}) \cap C_i)\}_{i \in J_H}$, compute the maximum total length of vertex-disjoint paths $\{P_i\}_{i \in J_H}$ such that each P_i is a path from a vertex in V_{i1} to a vertex in V_{i2}.

We give FPT algorithms for this problem on both cliques and trees. First, we explain the algorithm for cliques. We brute-force over flags $f \in \{0,1\}^{J_H}$, where $f_i = 0$ represents that P_i consists of a single vertex, and $f_i = 1$ indicates that P_i contains at least two vertices. The problem of whether a family of paths satisfying the conditions defined by f exists can be reduced to the bipartite matching problem and, thus, solved in polynomial time. If such a family of paths exists for some flag f such that $f_i = 1$ holds for some $i \in J_H$, we can extend the family to cover all vertices in H by appropriately adding internal vertices to the paths. Thus, in this case, the answer is $|H| - |J_H|$. If such a family of paths exists only for $f = (0, \ldots, 0)$, the answer is zero. If no such family of paths exists for any f, then the problem is infeasible. The time complexity is $O^*(2^k)$.

Now, we explain the algorithm for trees. Our algorithm uses dynamic programming. We omit the formal details here, but the intuition is as follows. Regard H as a rooted tree. For a vertex v and a set $Z \subseteq J_H$, we define $\mathsf{DP}[v][Z]$ denote the maximum total length of a family of paths $\{P_i\}_{i \in Z}$ that can be packed into the subtree rooted at v. We compute these values in a bottom-up manner. We can analyze that such dynamic programming can be implemented to work in $O^*(3^k)$ time.

1.4 Organization

The rest of this paper is organized as follows. In Section 2, we introduce basic notation and well-known techniques in the literature on kernelization. In Section 3, we prove Theorem 1 by constructing a kernel for Cliques or Trees Vertex Deletion with $O(k^2)$ vertices. Due to the space limitation, some proofs are given in the full version. In the full version, we prove Theorem 2 by presenting an FPT algorithm for Longest Cycle parameterized by the cliques or trees vertex deletion number.

2 Preliminaries

In this paper, the term *graph* refers to an undirected graph, which does not contain self-loops but may contain multi-edges. For a graph $G = (V, E)$ and a vertex $v \in V$, we call a vertex belonging to $N_G(v) := \{u \in V : (u, v) \in E\}$ a *neighbor* of v in G, and the number of edges incident to v, denoted $d_G(v)$, is referred to as the *degree* of v. Note that $|N_G(v)|$ and $d_G(v)$ may differ due to multi-edges. Moreover, we denote by $\rho_G(v)$ the size of the set $\{\{u_1, u_2\} \subseteq N_G(v) : (u_1, u_2) \in E\}$. In other words, $\rho_G(v)$ represents the number of edges connecting the neighbors of v in G, where multi-edges are counted as a single edge. When the context is clear, we omit the subscript G and write $N(v)$, $d(v)$, and $\rho(v)$ for simplicity. A vertex v with $d(v) = 1$ is called a *pendant*.

For a vertex subset $Z \subseteq V$, we denote $E(Z) := \{e \in E : e \subseteq Z\}$. The *subgraph of G induced by Z* is the graph $(Z, E(Z))$. For simplicity, when there is no risk of confusion, we identify the vertex set Z with the subgraph induced by it: that is, when we refer to the "graph Z" for $Z \subseteq V$, we mean the subgraph induced by Z. Moreover, for a vertex subset $Z \subseteq V$, we write $G - Z$ to denote the subgraph induced by $V \setminus Z$. When Z consists of a single vertex z, we abbreviate $G - \{z\}$ as $G - z$. Similarly, we denote by $G - e$ the graph obtained by removing an edge $e \in E$ from G. A vertex subset $Z \subseteq V$ induces a *clique* if there is exactly one edge between any two different vertices in Z, an *independent set* if there is no edge between any two vertices in Z, and a *tree* if Z is connected and contains no cycle.

For a vertex $v \in V$ and $t \in \mathbb{Z}_{\geq 1}$, a *$v$-flower* of *order* t is a set of t cycles passing through v such that no two cycles share a common vertex other than v. The following is well-known.

▶ **Lemma 3** (Gallai's Theorem [7, 17, 39]). *Given an undirected graph $G = (V, E)$, a vertex $v \in V$, and an integer $t \in \mathbb{Z}_{\geq 1}$, there is a polynomial-time algorithm that computes either*

- *a v-flower of order $t + 1$, or*
- *a vertex set $B \subseteq V \setminus \{v\}$ of size at most $2t$ such that $G - B$ contains no cycle passing through v.*

For vertex subsets K, L and an integer $q \in \mathbb{Z}_{\geq 1}$, an edge set M is a *q-expansion of K into L* if

- exactly q edges of M are incident to each vertex in K, and
- exactly one edge of M is incident to each vertex in L.

The following lemma is also well-known in the literature on kernelization.

▶ **Lemma 4** (Expansion Lemma [7, 17]). *Let $H := (K \dot\cup L, E)$ be a bipartite graph with vertex bipartition (K, L) and $q \in \mathbb{Z}_{\geq 1}$. Assume $|L| \geq q|K|$ and L contains no isolated vertex. Then, there is a polynomial-time algorithm that computes a pair of non-empty vertex subsets $K' \subseteq K$ and $L' \subseteq L$ such that*

- *$N_H(L') \subseteq K'$, and*
- *there exists a q-expansion of K' into L'.*

3 Quadratic Kernel for Cliques or Trees Vertex Deletion

Let $G = (V, E)$ be a graph and $k \in \mathbb{Z}_{\geq 1}$. In this section, we prove Theorem 1 by constructing a quadratic kernel for Cliques or Trees Vertex Deletion. Due to the space limitation, we defer the proofs of the lemmas marked with an asterisk to the full version.

3.1 Partitioning Vertices

We begin by classifying the vertices into three categories. Let

$$\begin{aligned} V_{\mathrm{ls}} &:= \left\{ v \in V \colon |N(v)| > 7k \wedge \rho(v) \leq \frac{|N(v)|(|N(v)|-1)}{4} \right\}, \\ V_{\mathrm{ld}} &:= \left\{ v \in V \colon |N(v)| > 7k \wedge \rho(v) > \frac{|N(v)|(|N(v)|-1)}{4} \right\}, \\ V_{\mathrm{small}} &:= \{ v \in V \colon |N(v)| \leq 7k \}. \end{aligned}$$

The vertices belonging to the first, second, and third groups are referred to as *large-sparse*, *large-dense*, and *small*, respectively. The following lemma states that, for any feasible solution X, large-sparse vertices are included in either X or a tree component of $G - X$.

▶ **Lemma 5 (*).** *Let $v \in V_{\mathrm{ls}}$. Then, for any feasible solution X with $v \notin X$, v is in a tree component of $G - X$.*

The following lemma states a result symmetric to Lemma 5, that is, for any feasible solution X, large-dense vertices are included in either X or a clique component of $G - X$.

▶ **Lemma 6 (*).** *Let $v \in V_{\mathrm{ld}}$. Then, for any feasible solution X with $v \notin X$, v is in a clique component of $G - X$.*

Several times throughout this paper, we use the following type of alternative evidence for a vertex belonging to a clique component or a tree component.

▶ **Lemma 7 (*).** *Let $v \in V$ and X be a feasible solution with $v \notin X$. If $N(v)$ contains a clique of size $k + 2$, then v is in a clique component of $G - X$. Similarly, if $N(v)$ contains an independent set of size $k + 2$, then v is in a tree component of $G - X$.*

In the rest of this paper, we will use Lemmas 5, 6, and 7 as basic tools without specifically mentioning them.

3.2 Bounding Sizes of Neighbors of Vertices in V_{ls}

In this section, we reduce the size of $N(v)$ for large-sparse vertices $v \in V_{\mathrm{ls}}$. Most of the reduction rules in this section are the same as the quadratic kernelization of Feedback Vertex Set by Thomassé [39], while some details in the analysis require additional care in proofs. We begin with the following.

▶ **Reduction Rule 1.** *Let v be any large-sparse vertex. Apply Lemma 3 for v and $t = k$. If a v-flower of order $k + 1$ is found, remove v and decrease k by 1.*

▶ **Lemma 8 (*).** *Reduction Rule 1 is safe.*

Let v be a large-sparse vertex and assume Lemma 3 finds a vertex set B with size at most $2k$ that hits all cycles containing v. Let $\mathcal{C}_{\mathrm{tree}}$ be the family of connected components of $G - v - B$ that are trees and adjacent to v. Similarly, let $\mathcal{C}_{\mathrm{nontree}}$ be the family of connected components of $G - v - B$ that are not trees and adjacent to v. Since $G - B$ contains no cycle containing v, for each $C \in \mathcal{C}_{\mathrm{tree}} \cup \mathcal{C}_{\mathrm{nontree}}$, we have $|N(v) \cap C| = 1$. Particularly, $|N(v)| = |\mathcal{C}_{\mathrm{tree}}| + |\mathcal{C}_{\mathrm{nontree}}| + |N(v) \cap B|$. We bound $|N(v)|$ by bounding these three terms. Obviously, $|N(v) \cap B| \leq |B| \leq 2k$. To bound $|\mathcal{C}_{\mathrm{nontree}}|$, we use the following reduction rule.

▶ **Reduction Rule 2.** *If $|\mathcal{C}_{\mathrm{nontree}}| \geq k + 1$, remove v and decrease k by 1.*

▶ **Lemma 9 (*).** *Reduction Rule 2 is safe.*

Now we bound $|\mathcal{C}_{\text{tree}}|$ by $4k$ using the expansion lemma. We construct an auxiliary bipartite graph H. The vertex set of H is $B \dot\cup \mathcal{C}_{\text{tree}}$ with bipartition $(B, \mathcal{C}_{\text{tree}})$. We add an edge between $b \in B$ and $C \in \mathcal{C}_{\text{tree}}$ if and only if $N_G(b) \cap C \neq \emptyset$. We use the following reduction rule to ensure the part $\mathcal{C}_{\text{tree}}$ does not contain isolated vertices.

▶ **Reduction Rule 3.** *If there is a component $C \in \mathcal{C}_{\text{tree}}$ that has no neighbor in B, remove all vertices of C.*

▶ **Lemma 10 (*).** *Reduction Rule 3 is safe.*

Assume $|\mathcal{C}_{\text{tree}}| \geq 4k$. We apply 2-expansion lemma to H and obtain vertex sets $\mathcal{C}' \subseteq \mathcal{C}_{\text{tree}}$ and $B' \subseteq B$ such that there is a 2-expansion of B' into $\mathcal{C}'$. We have $|B'| \leq k$ because otherwise we obtain a v-flower of order $k+1$. We apply the following reduction.

▶ **Reduction Rule 4.** *Remove each edge between v and $\mathcal{C}'$. Then, connect v and each vertex in B' by a double-edge.*

▶ **Lemma 11 (*).** *Assume $|\mathcal{C}_{\text{tree}}| \geq 4k$. Then, Reduction Rule 4 is safe.*

Since all the above reduction rules reduce the number of pairs of vertices connected by at least one edge, the reduction rules in this section can be applied only a polynomial number of times. Thus, we have the following.

▶ **Lemma 12 (*).** *Let G be the graph obtained by exhaustively applying all the above reduction rules. Then, for all vertex $v \in V \setminus V_{\text{ld}}$, we have $|N(v)| \leq 7k$ (and thus, $V_{\text{ls}} = \emptyset$).*

3.3 Bounding $|V_{\text{ld}}|$

In this section, we reduce the number of vertices in V_{ld}. This part is the main technical contribution of this paper. As in [28], we apply a 4-approximation algorithm for Cliques or Trees Vertex Deletion given in [27] and obtain an approximate solution $S \subseteq V$. We apply the following reduction rule to ensure $|S| \leq 4k$, which is clearly safe.

▶ **Reduction Rule 5.** *If $|S| > 4k$, return* no.

Let $\mathcal{C}_{\text{clique}}$ be the family of connected components of $G - S$ that are cliques of size at least 3. Similarly, let $\mathcal{C}_{\text{tree}}$ be the family of connected components of $G - S$ that are trees. The following rule ensures that each vertex $v \in V_{\text{ld}}$ is contained in a clique component of $G - S$ unless $v \in S$.

▶ **Reduction Rule 6.** *If there is a vertex $v \in V_{\text{ld}}$ that is in a tree component of $G - S$, remove v and decrease k by 1.*

▶ **Lemma 13.** *Reduction Rule 6 is safe.*

Proof. We prove that v is contained in all solutions X. Since $v \in V_{\text{ld}}$, v is in a clique component of $G - X$ unless $v \in X$. Since v is in a tree component of $G - S$, $N_{G-S}(v)$ is an independent set, whose size is at least $|N_G(v)| - |S| \geq 7k - 4k = 3k \geq k + 2$. Thus, v is in a tree component of $G - X$ unless $v \in X$, which leads to $v \in X$. ◀

We construct an auxiliary bipartite graph H as follows. The vertex set of H is $S \dot\cup \mathcal{C}_{\text{clique}}$ with bipartition $(S, \mathcal{C}_{\text{clique}})$. We add an edge between $s \in S$ and $C \in \mathcal{C}_{\text{clique}}$ if and only if $N_G(s) \cap C \neq \emptyset$. To ensure the part $\mathcal{C}_{\text{clique}}$ does not contain isolated vertices, we apply the following fundamental rule, which is clearly safe.

▶ **Reduction Rule 7.** *If G contains a connected component that is a clique or a tree, remove that component.*

Assume $|\mathcal{C}_{\text{clique}}| \geq 2|S|$. We apply 2-expansion lemma to H and obtain vertex sets $\mathcal{C}' \subseteq \mathcal{C}_{\text{clique}}$ and $S' \subseteq S$. The following reduction is proved to be safe in [28].

▶ **Reduction Rule 8.** *Remove vertices of S' and decrease k by $|S'|$.*

▶ **Lemma 14** ([28])**.** *Reduction Rule 8 is safe.*

Now we can assume $|\mathcal{C}_{\text{clique}}| \leq 8k$. We first bound the number of vertices in V_{ld} that are adjacent to some vertices outside V_{ld}. We have the following.

▶ **Lemma 15.** *After applying all the above reduction rules, there are at most $84k^2 + 4k$ vertices in V_{ld} that have some neighbor outside V_{ld}.*

Proof. A large-dense vertex v is adjacent to a vertex outside V_{ld} only when
(i) $v \in S$,
(ii) $v \notin S$ and v has a neighbor in $S \setminus V_{\text{ld}}$, or
(iii) $v \notin S$ and the component $C \in \mathcal{C}_{\text{clique}}$ containing v contains a vertex from $V \setminus V_{\text{ld}}$.
Reduction Rule 5 ensures that at most $|S| \leq 4k$ vertices satisfy condition (i). Moreover, since vertices in $V \setminus V_{\text{ld}}$ has degree at most $7k$, at most $7k|S| \leq 28k^2$ vertices satisfy condition (ii). Furthermore, if v satisfies condition (iii), C is a clique of size at most $7k + 1$ because it contains a vertex with degree at most $7k$. Therefore, at most $7k|\mathcal{C}_{\text{clique}}| \leq 56k^2$ vertices satisfy condition (iii). ◀

Now we bound the number of vertices $v \in V_{\text{ld}}$ with $N_G(v) \subseteq V_{\text{ld}}$. A vertex triplet (v_1, v_2, v_3) *induces* P_3 if $(v_1, v_2), (v_2, v_3) \in E$ and $(v_3, v_1) \notin E$. We remark that even if the edges (v_1, v_2) or (v_2, v_3) are multi-edges, we still consider them as inducing a P_3. The following lemma is analogous to the fundamental observation in the literature on Cluster Vertex Deletion.

▶ **Lemma 16.** *Let X be a feasible solution and P be a subset of V_{ld} that induces P_3. Then, $X \cap P \neq \emptyset$.*

Proof. Assume otherwise and let C be the connected component of $G - X$ that contains P. From Lemma 6, C is a clique component. However, cliques cannot contain induced P_3, leading to a contradiction. ◀

This leads to the following reduction rule, which is clearly safe.

▶ **Reduction Rule 9.** *Let $v \in V_{\text{ld}}$. If there is a collection of $k+1$ induced P_3s in V_{ld} such that any two of them intersect only at $\{v\}$, then remove v and decrease k by 1.*

Whether $v \in V$ satisfies the condition of Reduction Rule 9 can be checked by computing the maximum matching on the graph $(N_G(v), E_v)$, where E_v is the set of pairs of the vertices (u_1, u_2) such that $\{v, u_1, u_2\}$ induces P_3. Therefore, this rule can be applied in polynomial time. Let $\mathcal{P}$ be a maximal collection of induced P_3s in V_{ld} such that any two induced P_3s in the collection have an intersection of size at most 1. The idea to construct this $\mathcal{P}$ and the following reduction rule are borrowed from the quadratic kernelization of 3-Hitting Set by Abu-Khzam [1].

▶ **Reduction Rule 10.** *If $|\mathcal{P}| > k^2$, return* no*.*

▶ **Lemma 17.** *Reduction Rule 10 is safe.*

Proof. Assume $|\mathcal{P}| > k^2$ and let X be a feasible solution. From the assumption that Reduction Rule 9 cannot be applied, each vertex in X hits at most k induced P_3s in $\mathcal{P}$. Therefore, there exists an induced P_3 that is disjoint from X, which contradicts Lemma 16. ◀

Now we can assume $\left|\bigcup_{P\in\mathcal{P}} P\right| \leq 3|\mathcal{P}| \leq 3k^2$. The remaining task is to bound the number of vertices in $V_{\text{ldmod}} := V_{\text{ld}} \setminus \bigcup_{P\in\mathcal{P}} P$. We first bound the number of connected components.

▶ **Lemma 18.** *V_{ldmod} contains at most $12k$ connected components.*

Proof. A clique can intersect at most one connected component of V_{ldmod}. Since Reduction Rule 8 is exhaustively applied, V_{ld} can be partitioned into at most $8k$ cliques and $|S| \leq 4k$ vertices. Therefore, there can be at most $8k + 4k = 12k$ connected components in V_{ldmod}. ◀

We state that each connected component of V_{ldmod} has a specific structure. A vertex set Z is *extended clique-module* in a graph $G' = (V', E')$ if

(i) for each different $u, v \in Z$, $(u, v) \in E'$, and

(ii) for each $u, v \in Z$, $N_{G'}(u) \cup \{u\} = N_{G'}(v) \cup \{v\}$.

In other words, an extended clique-module is a *clique-module* [17] in the graph obtained by reducing the multiplicity of each multi-edge to 1. We can prove that each connected component of V_{ldmod} is actually an extended clique-module in V_{ld}.

▶ **Lemma 19.** *Each connected component of V_{ldmod} is an extended clique-module in V_{ld}.*

Proof. Let C be a connected component of V_{ldmod}. C should satisfy condition (i), because otherwise C would contain an induced P_3, contradicting the maximality of $\mathcal{P}$. Moreover, C should satisfy condition (ii), because otherwise there would exist $u \in \bigcup_{P\in\mathcal{P}} P$ and $v_1, v_2 \in C$ such that $(u, v_1) \in E$ and $(u, v_2) \notin E$, which would form an induced P_3, again contradicting the maximality of $\mathcal{P}$. ◀

Let E_{mul} be the set of multi-edges of G. Since any feasible solution should contain at least one endpoint of each edge in E_{mul}, the graph $G_{\text{mul}} := (V, E_{\text{mul}})$ should have a vertex cover of size k. This observation leads to the following two reduction rules called *Buss rule* [5, 7, 17] in the literature of kernelization of Vertex Cover, which are clearly safe.

▶ **Reduction Rule 11.** *If there is a vertex $v \in V$ with $|N_{G_{\text{mul}}}(v)| > k$, remove v and decrease k by 1.*

▶ **Reduction Rule 12.** *If $|E_{\text{mul}}| > k^2$, return* no.

We finish the analysis by bounding the number of vertices in each connected component of V_{ldmod} that is neither adjacent to a vertex outside V_{ld} nor incident to a multi-edge. We have the following.

▶ **Lemma 20.** *Let $u, v \in V_{\text{ld}}$ with $(u, v) \in E$ and assume neither u nor v is incident to a multi-edge. Assume $N_G(u) \cup \{u\} = N_G(v) \cup \{v\}$. Then, for any minimum feasible solution X, either $\{u, v\} \subseteq X$ or $\{u, v\} \cap X = \emptyset$ holds.*

Proof. Assume a feasible solution X satisfies $u \in X$ and $v \notin X$. It is sufficient to prove that $X \setminus \{u\}$ is still feasible. Let C be a connected component of $G - (X \setminus \{u\})$ containing both u and v. Since X is feasible, all connected components of $G - (X \setminus \{u\})$ other than C are either cliques or trees. It suffices to show that C is a clique. Since $v \in V_{\text{ld}}$, the connected component C' of $G - X$ that contains v is a clique. Since $N_{G-(X\setminus\{u\})}(u) = N_{G-(X\setminus\{u\})}(v) \cup \{v\} \setminus \{u\} = N_{G-X}(v) \cup \{v\} \setminus \{u\} = C'$, we obtain $C = C' \cup \{u\}$. Since there are no multi-edges between u and C', C is a clique. ◀

Now, we apply the following reduction rule.

▶ **Reduction Rule 13.** *Assume a connected component of V_{ldmod} contains $k+4$ vertices that are not adjacent to vertices outside V_{ld} and incident to no multi-edges. Then, remove one of those $k+4$ vertices.*

▶ **Lemma 21.** *Reduction Rule 13 is safe.*

Proof. Let Z be a set of vertices satisfying the assumption of the rule. Clearly, Z induces a clique. Since each vertex in Z has no neighbor outside V_{ld}, from Lemma 19, the set $N_G(v) \cup \{v\}$ is same for all $v \in Z$. Let X be a minimum feasible solution for G. Then, Lemma 20 ensures $Z \cap X = \emptyset$ because $|Z| > k$. Therefore, X is still a feasible solution for $G - v$ for any $v \in Z$. Conversely, let $v \in Z$ and X' be a feasible solution for $G - v$. Since $Z \setminus \{v\}$ is a clique of size $k+3$, $Z - v - X'$ is contained in a single clique component C' of $G - v - X'$. Moreover, we have $N_{G-X'}(v) = N_{G-X'}(u) \cup \{u\} \setminus \{v\}$ for all $u \in Z \setminus (X' \cup \{v\})$, and thus, $N_{G-X'}(v) = C'$. Particularly, $C' \cup \{v\}$ is a clique component of $G - X'$. Therefore, X' is still a feasible solution for G. ◀

Now we can bound $|V_{\mathrm{ld}}|$ by $O(k^2)$.

▶ **Lemma 22.** *After applying all the above reduction rules, we have $|V_{\mathrm{ld}}| \leq 101k^2 + 40k$.*

Proof. A vertex v is in V_{ld} only when
- **(i)** it has a neighbor outside V_{ld},
- **(ii)** it is in $\bigcup_{P \in \mathcal{P}} P$,
- **(iii)** there is a multi-edge incident to v, or
- **(iv)** it does not satisfy neither of condition (i), (ii), and (iii).

Lemma 15 states that at most $84k^2 + 4k$ vertices satisfy condition (i). Reduction Rule 10 ensures that at most $3k^2$ vertices satisfy condition (ii). Reduction Rule 12 ensures that at most $2k^2$ vertices satisfy condition (iii). Lemma 18 and Reduction Rule 13 ensures at most $12k \cdot (k+3) = 12k^2 + 36k$ vertices satisfy condition (iv). Therefore, we have $|V_{\mathrm{ld}}| \leq (84k^2 + 4k) + 3k^2 + 2k^2 + (12k^2 + 36k) \leq 101k^2 + 40k$. ◀

3.4 Bounding Number of Vertices

Now, we finish the overall analysis by bounding the number of vertices in $V \setminus V_{\mathrm{ld}}$. We first bound the number of edges between S and the tree components of $G - S$ (remember, S is a 4-approximate solution). Let $V_{\mathrm{tree}} := \bigcup_{C \in \mathcal{C}_{\mathrm{tree}}} C$. We begin by the following.

▶ **Reduction Rule 14.** *If there is a vertex $v \in V_{\mathrm{ld}} \cap S$ such that $N(v)$ contains at least $2k+4$ vertices from V_{tree}, remove v and decrease k by 1.*

▶ **Lemma 23 (*).** *Reduction Rule 14 is safe.*

Thus, we can bound the number of vertices in V_{tree} adjacent to some vertex in S.

▶ **Lemma 24 (*).** *After exhaustively applying all the above reduction rules, at most $28k^2$ vertices in V_{tree} are adjacent to some vertex in S.*

We apply the following reduction rules by local structures. Most of them appear in [28], but Reduction Rule 19 is new.

▶ **Reduction Rule 15 ([28]).** *If a vertex v is adjacent to at least two pendant vertices, remove one of them.*

▶ **Reduction Rule 16** ([28]). *If there are multi-edges with a multiplicity of at least* 3*, reduce the multiplicity to* 2*.*

▶ **Reduction Rule 17** ([28]). *If there are three different vertices* (v_1, v_2, v_3) *such that* $(v_1, v_2), (v_2, v_3) \in E$*,* $(v_1, v_3) \notin E$*,* $d(v_2) = 2$*, and* $d(v_3) = 1$*, remove* v_3*.*

▶ **Reduction Rule 18** ([28]). *If there are five different vertices* $(v_1, v_2, v_3, v_4, v_5)$ *such that* (v_1, v_2)*,* (v_2, v_3)*,* (v_3, v_4)*,* (v_4, v_5) *are in* E *and* $d(v_2) = d(v_3) = d(v_4) = 2$*, remove* v_3 *and add an edge* (v_2, v_4)*.*

▶ **Reduction Rule 19.** *If there are four different vertices* (v_1, v_2, v_3, v_4) *such that* $(v_i, v_j) \in E$ *if and only if* $i = 1$*,* $d(v_1) = 3$*, and* $d(v_4) = 1$*, remove* v_4*.*

▶ **Lemma 25** (*). *Reduction Rule 19 is safe.*

Now, we bound $|V_{\text{tree}}|$. We begin with the following.

▶ **Lemma 26** (*). *After applying all the above reduction rules,* V_{tree} *contains at most* $28k^2$ *pendant vertices that are adjacent to a vertex of degree* 3*.*

Now, we can prove the following.

▶ **Lemma 27** (*). *We have* $|V_{\text{tree}}| \leq 1232k^2$*.*

Now, we can bound the size of the whole graph, which directly proves Theorem 1.

▶ **Lemma 28** (*). *After applying all the above reduction rules, we have* $|V| \leq 1389k^2 + 52k$*.*

References

1. Faisal Abu-Khzam. A kernelization algorithm for d-hitting set. *Journal of Computer and System Sciences*, 76(7):524–531, 2010. `doi:10.1016/j.jcss.2009.09.002`.
2. Matthias Bentert, Michael Fellows, Petr Golovach, Frances Rosamond, and Saket Saurabh. Breaking a graph into connected components with small dominating sets. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 24–1, 2024. `doi:10.4230/LIPIcs.MFCS.2024.24`.
3. Benjamin Bergougnoux, Mamadou Moustapha Kanté, and O-joung Kwon. An optimal XP algorithm for hamiltonian cycle on graphs of bounded clique-width. *Algorithmica*, 82(6):1654–1674, 2020. `doi:10.1007/s00453-019-00663-9`.
4. Stéphane Bessy, Marin Bougeret, Dimitrios Thilikos, and Sebastian Wiederrecht. Kernelization for graph packing problems via rainbow matching. In *Symposium on Discrete Algorithms (SODA)*, pages 3654–3663. SIAM, 2023. `doi:10.1137/1.9781611977554.ch139`.
5. Jonathan Buss and Judy Goldsmith. Nondeterminism within P^*. *SIAM Journal on Computing*, 22(3):560–572, 1993. `doi:10.1137/0222038`.
6. Yixin Cao and Dániel Marx. Interval deletion is fixed-parameter tractable. *ACM Transactions on Algorithms (TALG)*, 11(3):1–35, 2015. `doi:10.1145/2629595`.
7. Marek Cygan, Fedor Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized algorithms*, volume 5. Springer, 2015. `doi:10.1007/978-3-319-21275-3`.
8. Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast hamiltonicity checking via bases of perfect matchings. *Journal of the ACM (JACM)*, 65(3):1–46, 2018. `doi:10.1145/3148227`.
9. Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michał Pilipczuk, Johan MM Van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. *ACM Transactions on Algorithms (TALG)*, 18(2):1–31, 2022. `doi:10.1145/3506707`.

10 Martin Doucha and Jan Kratochvíl. Cluster vertex deletion: A parameterization between vertex cover and clique-width. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 348–359. Springer, 2012. doi:10.1007/978-3-642-32589-2_32.

11 Rod Downey and Michael Fellows. Fixed-parameter tractability and completeness i: Basic results. *SIAM Journal on computing*, 24(4):873–921, 1995. doi:10.1137/S0097539792228228.

12 Michael Fellows, Lars Jaffke, Aliz Izabella Király, Frances Rosamond, and Mathias Weller. What is known about vertex cover kernelization? In *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*, pages 330–356. Springer, 2018. doi:10.1007/978-3-319-98355-4_19.

13 Fedor Fomin, Petr Golovach, Daniel Lokshtanov, and Saket Saurabh. Almost optimal lower bounds for problems parameterized by clique-width. *SIAM Journal on Computing*, 43(5):1541–1563, 2014. doi:10.1137/130910932.

14 Fedor Fomin, Petr Golovach, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. Clique-width III: Hamiltonian cycle and the odd case of graph coloring. *ACM Transactions on Algorithms (TALG)*, 15(1):1–27, 2018. doi:10.1145/3280824.

15 Fedor Fomin, Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Subquadratic kernels for implicit 3-hitting set and 3-set packing problems. *ACM Transactions on Algorithms (TALG)*, 15(1):1–44, 2019. doi:10.1145/3293466.

16 Fedor Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar $\mathcal{F}$-deletion: Approximation, kernelization and optimal fpt algorithms. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 470–479. IEEE, 2012. doi:10.1109/FOCS.2012.62.

17 Fedor Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing.* Cambridge University Press, 2019.

18 Esther Galby, Liana Khazaliya, Fionn Mc Inerney, Roohani Sharma, and Prafullkumar Tale. Metric dimension parameterized by feedback vertex set and other structural parameters. *SIAM Journal on Discrete Mathematics*, 37(4):2241–2264, 2023. doi:10.1137/22m1510911.

19 Robert Ganian. Improving vertex cover as a graph parameter. *Discrete Mathematics & Theoretical Computer Science*, 17(2):77–100, 2015. doi:10.46298/dmtcs.2136.

20 Tatsuya Gima, Eun Jung Kim, Noleen Köhler, Nikolaos Melissinos, and Manolis Vasilakis. Bandwidth parameterized by cluster vertex deletion number. In *International Symposium on Parameterized and Exact Computation (IPEC)*, volume 285, pages 21:1–21:15, 2023. doi:10.4230/LIPIcs.IPEC.2023.21.

21 Petr Golovach, R Krithika, Abhishek Sahu, Saket Saurabh, and Meirav Zehavi. Graph hamiltonicity parameterized by proper interval deletion set. In *Latin American Symposium on Theoretical Informatics*, pages 104–115. Springer, 2020. doi:10.1007/978-3-030-61792-9_9.

22 Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. A more relaxed model for graph-based data clustering: s-plex cluster editing. *SIAM Journal on Discrete Mathematics*, 24(4):1662–1683, 2010. doi:10.1137/090767285.

23 David Harris and N. S. Narayanaswamy. A faster algorithm for vertex cover parameterized by solution size. In *International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 289, pages 40:1–40:18, 2024. doi:10.4230/LIPIcs.STACS.2024.40.

24 Falk Hüffner, Christian Komusiewicz, Hannes Moser, and Rolf Niedermeier. Fixed-parameter algorithms for cluster vertex deletion. *Theory of Computing Systems*, 47(1):196–217, 2010. doi:10.1007/s00224-008-9150-x.

25 Yoichi Iwata. Linear-time kernelization for feedback vertex set. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 80 of *LIPIcs*, pages 68:1–68:14, 2017. doi:10.4230/LIPIcs.ICALP.2017.68.

26 Ashwin Jacob, Jari JH de Kroon, Diptapriyo Majumdar, and Venkatesh Raman. Deletion to scattered graph classes I-case of finite number of graph classes. *Journal of Computer and System Sciences*, 138:103460, 2023. doi:10.1016/j.jcss.2023.05.005.

27 Ashwin Jacob, Diptapriyo Majumdar, and Venkatesh Raman. Deletion to scattered graph classes II-improved FPT algorithms for deletion to pairs of graph classes. *Journal of Computer and System Sciences*, 136:280–301, 2023. doi:10.1016/j.jcss.2023.03.004.

28 Ashwin Jacob, Diptapriyo Majumdar, and Meirav Zehavi. A polynomial kernel for deletion to the scattered class of cliques and trees. In *International Symposium on Algorithms and Computation (ISAAC)*, volume 322 of *LIPIcs*, pages 41:1–41:17, 2024. doi:10.4230/LIPIcs.ISAAC.2024.41.

29 Ashwin Jacob, Michal Wlodarczyk, and Meirav Zehavi. Finding long directed cycles is hard even when DFVS is small or girth is large. In *European Symposium on Algorithms (ESA)*, volume 274, pages 65:1–65:17, 2023. doi:10.4230/LIPIcs.ESA.2023.65.

30 Bart Jansen, Jari de Kroon, and Michal Wlodarczyk. Single-exponential FPT algorithms for enumerating secluded $\mathcal{F}$-free subgraphs and deleting to scattered graph classes. *Journal of Computer and System Sciences*, 148:103597, 2025. doi:10.1016/j.jcss.2024.103597.

31 Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. *ACM Transactions on Algorithms (TALG)*, 12(2):1–41, 2015. doi:10.1145/2797140.

32 Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 450–459. IEEE, 2012. doi:10.1109/FOCS.2012.46.

33 Michael Lampis, Georgia Kaouri, and Valia Mitsou. On the algorithmic effectiveness of digraph decompositions and complexity measures. *Discrete Optimization*, 8(1):129–138, 2011. doi:10.1016/j.disopt.2010.03.010.

34 Jason Li and Jesper Nederlof. Detecting feedback vertex sets of size k in $O^*(2.7^k)$ time. *ACM Transactions on Algorithms (TALG)*, 18(4):34:1–34:26, 2022. doi:10.1145/3504027.

35 Hong Liu, Peng Zhang, and Daming Zhu. On editing graphs into 2-club clusters. In *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management (FAW-AAIM)*, pages 235–246. Springer, 2012. doi:10.1007/978-3-642-29700-7_22.

36 Dániel Marx. Chordal deletion is fixed-parameter tractable. *Algorithmica*, 57:747–768, 2010. doi:10.1007/s00453-008-9233-8.

37 Ashutosh Rai and Saket Saurabh. Bivariate complexity analysis of almost forest deletion. *Theoretical Computer Science*, 708:18–33, 2018. doi:10.1016/j.tcs.2017.10.021.

38 Bruce Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Operations Research Letters*, 32(4):299–301, 2004. doi:10.1016/j.orl.2003.10.009.

39 Stéphan Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Transactions on Algorithms (TALG)*, 6(2):32:1–32:8, 2010. doi:10.1145/1721837.1721848.

40 Dekel Tsur. Faster algorithms and a smaller kernel for cliques or trees vertex deletion. *Information Processing Letters*, 190:106570, 2025. doi:10.1016/j.ipl.2025.106570.

41 Meirav Zehavi. Tournament fixing parameterized by feedback vertex set number is FPT. In *AAAI Conference on Artificial Intelligence*, pages 5876–5883, 2023. doi:10.1609/aaai.v37i5.25728.