Noname manuscript No.
(will be inserted by the editor)

Structural Parameterizations of Undirected Feedback Vertex Set: FPT Algorithms and Kernelization*

Diptapriyo Majumdar · Venkatesh Raman

Received: date / Accepted: date

Abstract A feedback vertex set in an undirected graph is a subset of vertices whose removal results in an acyclic graph. We consider the parameterized and kernelization complexity of feedback vertex set where the parameter is the size of some structure in the input. In particular, we consider parameterizations where the parameter is (instead of the solution size), the distance to a class of graphs where the problem is polynomial time solvable, and sometimes smaller than the solution size. Here, by distance to a class of graphs, we mean the minimum number of vertices whose removal results in a graph in the class. Such a set of vertices is also called the 'deletion set'. In this paper, we show that

- FVS is fixed-parameter tractable by an $\mathcal{O}(2^k n^{\mathcal{O}(1)})$ time algorithm, but is unlikely to have polynomial kernel when parameterized by the number of vertices of the graph whose degree is at least 4. This answers a question asked in an earlier paper. We also show that an algorithm with running time $\mathcal{O}((\sqrt{2}-\epsilon)^k n^{\mathcal{O}(1)})$ is not possible unless SETH fails.
- When parameterized by k, the number of vertices, whose deletion results in a split graph, we give an $\mathcal{O}(3.148^k n^{\mathcal{O}(1)})$ time algorithm.
- When parameterized by k, the number of vertices whose deletion results in a cluster graph (a disjoint union of cliques), we give an $\mathcal{O}(5^k n^{\mathcal{O}(1)})$ algorithm.

Regarding kernelization results, we show that

- When parameterized by k, the number of vertices, whose deletion results in a pseudo-forest, FVS has an $\mathcal{O}(k^7)$ vertices kernel improving from the previously known $\mathcal{O}(k^{10})$ bound.
- When parameterized by the number k of vertices, whose deletion results in a mock-d-forest, we give a kernel with $\mathcal{O}(k^{3d+3})$ vertices. We also prove a lower bound of $\Omega(k^{d+2})$ size (under complexity theoretic assumptions). Mock-forest is a graph where each vertex is contained in at most one cycle. Mock-d-forest for a constant d is a mock-forest where each component has at most d cycles.

Keywords Parameterized Complexity · Kernelization · Feedback Vertex Set · Structural Parameterization · W-hardness

1 Introduction

In the early years of parameterized complexity and algorithms, problems were almost always parameterized by solution size. Recent research has focused on other parameterizations based on structural properties of the input [26, 14, 25, 13, 33] and above or below guaranteed optimum values [22]. Such 'non-standard' parameters are known to be small in practice. It is a natural question to identify the

* Preliminary versions of this paper appeared in proceedings of IPEC 2016 [31] and FAW 2017 [32]

Diptapriyo Majumdar
The Institute of Mathematical Sciences, HBNI, Chennai, India
E-mail: diptapriyom@imsc.res.in

Venkatesh Raman
The Institute of Mathematical Sciences, HBNI, Chennai, India
E-mail: vraman@imsc.res.in

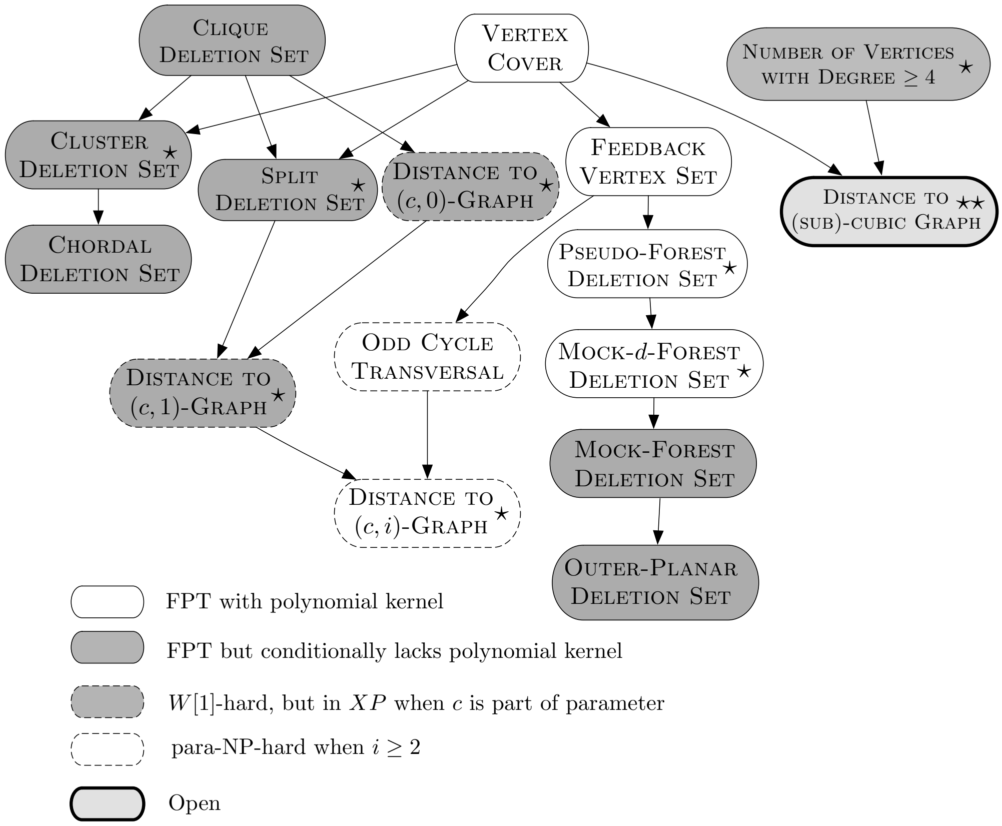

Fig. 1: Ecology of Parameters for Feedback Vertex Set. The parameter values are the minimum possible for a given graph. An arrow from parameter x to parameter y means that $x \geq y$ for a given graph. $\star$ indicates the results that are shown in this paper. $\star\star$ indicates the parameterizations for which the fixed-parameter tractability status is open.

smallest parameter under which a problem is fixed-parameter tractable and/or has a polynomial kernel. It means that once a problem is shown to be fixed-parameter tractable (and/or having a polynomial kernel) with respect to a parameter, it is a natural question whether it has a fixed-parameter algorithm or polynomial kernel with respect to a smaller parameter. Similarly, when a problem is W-hard or has no polynomial kernel then it is interesting to ask whether it is fixed-parameter tractable or admits a polynomial kernel when it is parameterized by a *structurally larger* parameter (See Figure 1 for the parameterizations considered in this paper). Kernelization is usually harder when the parameter is a structural parameter on the input as opposed to the solution size, where one can exploit the properties of an optimal (or k-sized) solutions.

Feedback Vertex Set in an undirected graph G asks whether G has a subset S of at most k vertices such that $G \setminus S$ is a forest, for a given integer k. The set S is called a feedback vertex set of the graph. The problem is known to be NP-Complete even on bipartite graphs [21] and in graphs of degree at most 4 [35], but is polynomial time solvable in sub-cubic graphs [37, 8, 7], asteroidal triple free graphs [30] and chordal bipartite graphs [27]. The problem is polynomial time solvable in pseudo-forests (graphs in which each component has at most one cycle), in mock-forests (graphs where each vertex is part of at most one cycle), in cliques and disjoint union of cliques. This is also one of the well-studied problems in parameterized complexity and when parameterized by solution size, it has an algorithm with running time $\mathcal{O}^*(3.619^k)$[1] [28] and a kernel with $\mathcal{O}(k^2)$ vertices and edges [36]. Cygan et al. [34, 9] have provided a randomized $\mathcal{O}^*(3^k)$ algorithm when k is solution size of the input graph. Cygan et al. [34] also have provided a randomized $\mathcal{O}^*(3^k)$ time algorithm when the parameter is the treewidth of the input graph. It is also known that Feedback Vertex Set admits a deterministic fixed-parameter tractable algorithm when parameterized by treewidth [2, 18].

[1] $\mathcal{O}^*$ notation suppresses the polynomial factors

Parameterizations of Feedback Vertex Set by the size of some structure in the input have also been explored. Feedback Vertex Set parameterized by the size of maximum induced matching (also maximum independent set and vertex clique cover) has been shown to be W[1]-hard but is in XP (See [26,1]) (See Section 2 for definitions). Bodlaender et al. [4] proved that Feedback Vertex Set parameterized by deletion distance to a cluster graph (disjoint union of cliques) has no polynomial kernel unless NP $\subseteq$ coNP/poly. If there is a set of at most k vertices whose deletion from G results in a graph of class $\mathcal{F}$, we say that "G is k-away from graph class $\mathcal{F}$". We use 'deletion distance' and 'k-away' alternatively to mean the same thing throughout the paper. We study such ecology of parameterization for Feedback Vertex Set continuing on the work in the survey by Jansen et al. [26].

Our Results: Jansen et al. suggested in [26], *"An interesting question in this direction is whether* Feedback Vertex Set *is* XP *or* FPT *when parameterized by the vertex deletion distance to sub-cubic graphs or alternatively, parameterized by the number of vertices of degree more than* 3". While the first question remains open, our first result is an answer to the latter question (FVS-High-Degree defined below). We answer it positively by providing a fixed parameter algorithm running in time $\mathcal{O}^*(2^k)$. We also prove that this problem has no polynomial kernel unless NP $\subseteq$ coNP/poly.

> FVS-High-Degree **Parameter:** k
> **Input:** An undirected graph G such that $|\{u \in V(G) | deg_G(u) > 3\}| \leq k$ and $\ell \in \mathbb{N}$.
> **Question:** Does G have a feedback vertex set of size at most ℓ?

We then study the parameterized complexity of Feedback Vertex Set when parameterized by the size of a split vertex deletion set. Feedback Vertex Set is polynomial time solvable on split graphs.

> FVS-SVD **Parameter:** k
> **Input:** An undirected multigraph G, $S \subseteq V(G)$ of size at most k such that $G \setminus S$ is a split graph and an integer ℓ.
> **Question:** Does G have a feedback vertex set of size at most ℓ?

Our algorithm for this problem runs in $\mathcal{O}^*(3.148^k)$ time.

We also completely characterize the parameterized complexity of Feedback Vertex Set when it is parameterized by the number of vertices whose deletion results in a (c, i)-graph (introduced in [29]) for different values of c and i. A graph is called a (c, i)-graph if its vertex set can be partitioned into c cliques and i independent sets. So a split graph is a $(1, 1)$-graph and a bipartite graph is a $(0, 2)$ graph. Hence some special cases of this parameterization include Feedback Vertex Set parameterized by split vertex deletion set and odd cycle transversal.

Next we consider the case when feedback vertex set is parameterized by the number of vertices whose deletion results in a disjoint union of cliques. Such a set of vertices is called a *cluster vertex deletion set*.

> FVS-CVD **Parameter:** k
> **Input:** An undirected multigraph G, $S \subseteq V(G)$ of size at most k such that every component of $G \setminus S$ is a clique and an integer ℓ.
> **Question:** Does G have a feedback vertex set of size at most ℓ?

We provide an algorithm with running time $\mathcal{O}^*(5^k)$ for this problem. It is known that FVS-CVD and FVS-SVD have no polynomial kernel [4] unless NP $\subseteq$ coNP/poly.

Our next set of results, which form the main results of the paper is on kernelization for some specific parameterizations for which FVS is known to be fixed-parameter tractable. To start with, we give an improved kernel for the following problem for which an $\mathcal{O}(k^{10})$ vertex kernel and a lower bound of $\Omega(k^4)$ (unless NP $\subseteq$ coNP/poly) were given by Jansen et al. [26].

> FVS-Pseudo-Forest **Parameter:** k
> **Input:** An undirected graph G, $S \subseteq V(G)$ of size at most k such that $G[V(G) \setminus S]$ is a graph in which every component has at most one cycle and an integer ℓ.
> **Question:** Does G have a feedback vertex set of size at most ℓ?

We give a kernel on $\mathcal{O}(k^7)$ vertices, narrowing the gap between upper and lower bound for the size of the kernel. Then, we consider a variation of mock-forests (called mock-d-forest) where each component has at most d cycles, where d is a constant, and consider the kernelization complexity of FVS parameterized by the deletion distance to mock-d-forests. It is easy to see that FVS is fixed-parameter tractable when parameterized by the deletion distance to mock-d-forest (or any mock-forest) as any mock-forest has treewidth at most 2. When d is not bounded, then we know that this problem has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ [26]. We consider the case when the number of cycles in a mock-forest is bounded by a constant.

> FVS-MOCK-d-FOREST WHERE $d \geq 2$ **Parameter:** k
> **Input:** An undirected graph G, $S \subseteq V(G)$ of size at most k such that $G[V(G) \setminus S]$ is a mock-forest where every component has at most d cycles and an integer ℓ.
> **Question:** Does G have a feedback vertex set of size at most ℓ?

Here, we provide a $\mathcal{O}(k^{3d+3})$ vertex kernel for this problem when d is a constant. And we also prove that a kernel consisting of $\mathcal{O}(k^{d+2-\epsilon})$ size is unlikely for any $\epsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. Deletion distance to a mock-forest is smaller than the deletion distance to mock-d-forest. Also deletion distance to the class of pseudo-forests is larger than deletion distance to mock-d-forests (See Figure 1 for the hierarchy of parameters).

Note that both the above mentioned parameters, i.e. *pseudo-forest deletion set, mock-d-forest deletion set* are provably smaller than the solution size. However, *mock-d-forest deletion set* is provably larger than *mock-forest deletion set* but provably smaller than *pseudo-forest deletion set.* We assume that for all our parameterizations, the deletion set is given with the input. For some of these parameters, this assumption is not critical. We discuss about the algorithms to find those deletion sets in the appropriate sections. See Figure 1 for a hierarchy of parameters considered in the paper. Sometimes, we call these deletion sets as "modulator" (See more results in [24, 33, 16, 14, 4, 25]).

We organize our paper as follows. In Section 2, we introduce the required notation. In Section 3, we provide the FPT Algorithms for FVS-HIGH-DEGREE, FVS-SVD, FVS-DELETION-TO-(c,i)-GRAPH and FVS-CVD. In Section 4, we provide the improved polynomial kernel for FVS-PSEUDO-FOREST and a polynomial kernel for FVS-MOCK-d-FOREST. In Section 4, we also prove that FVS-HIGH-DEGREE has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

2 Preliminaries and Notations

By $[r]$, we mean the set $\{1, 2, \ldots, r\}$. We use $A \uplus B$ to mean $A \cup B$ when $A \cap B = \emptyset$. We denote the *feedback vertex set number* (the size of a minimum feedback vertex set) by $fvs(G)$ or sometimes simply fvs when the context of the graph is clear. Let S be a set of vertices. By $\binom{S}{r}$, we denote the family of subsets of S containing *exactly* r vertices. By $\binom{S}{\leq r}$, we denote the family of subsets of S containing *at most* r vertices. By $\binom{S}{\geq r}$, we denote the family of subsets of S containing *at least* r vertices.

We allow our input graph to be a multigraph allowing multiple edges between a pair of vertices. We call a pair of vertices (u, v) a *double edge* if there are 2 edges between u and v. Otherwise we call (u, v) a *non-double-pair.* While computing the degree of a vertex, we take the multiplicity of edges into account. For a vertex $u \in V(G)$ and a subgraph H of G, by $deg_H(u)$, we denote the degree of u in H. A set of vertices $V' \subseteq V(G)$ is a *degree-2-path* if V' induces an acyclic path and every vertex of the path has degree exactly 2 in G. A degree-2-path is maximal if no proper superset of V' is a degree-2-path. Let G be a graph and (u, v) be an edge. We denote $G' = G/(u, v)$ as the graph created by contraction of the edge (u, v). Let uv be the contracted vertex as a result of contraction. Then, $N_{G'}(uv) = (N_G(u) \cup N_G(v)) \setminus \{u, v\}$. We denote $G[B]$ by the graph induced on the vertex set $B \subseteq V(G)$. We say that $G[B]$ is a *double-clique* if there are at least 2 edges between every pair of vertices in B.

We give the definitions of fixed-parameter tractability, W-hardness, kernelization, polynomial parameter transformation and their related facts.

2.1 Definitions and Properties

A language is L is called a parameterized language if its input instance consists of a pair (x, k) where $x \in \Sigma^*$ and $k \in \mathbb{N}$.

Definition 1 (Fixed-Parameter Tractability) Let $L \subseteq \Sigma^* \times \mathbb{N}$ is a parameterized language. L is said to be fixed-parameter tractable (or FPT) if there exists an algorithm $\mathcal{B}$, a constant c and a computable function f such that for all $x \in \Sigma^*$, for all $k \in \mathbb{N}$, algorithm $\mathcal{B}$ on input (x, k) runs in at most $f(k)|x|^c$ time and outputs $(x, k) \in L$ if and only if $\mathcal{B}([x, k]) = 1$. We call the algorithm $\mathcal{B}$ as fixed-parameter algorithm. Note that c is a constant that is independent of $|x|$ and k.

Definition 2 (Slice-Wise Polynomial (XP)) Let $L \subseteq \Sigma^* \times \mathbb{N}$ is a parameterized language. L is said to be Slice-Wise Polynomial (or in XP) if there exists an algorithm $\mathcal{B}$, a constant c and computable functions f, g such that for all $x \in \Sigma^*$, forall $k \in \mathbb{N}$, algorithm $\mathcal{B}$ on input (x, k) runs in at most $f(k)|x|^{g(k)+c}$ time and outputs $(x, k) \in L$ if and only if $\mathcal{B}([x, k]) = 1$. We call the algorithm $\mathcal{B}$ as an XP Algorithm.

Definition 3 (Parameterized Reduction) Let $L_1, L_2 \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized languages. We say that there exists a parameterized reduction from L_1 to L_2 if there exists a constant c, an algorithm $\mathcal{B}$ and computable functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $\mathcal{B}$ on input instance (x, k) of L_1 outputs an instance (x', k') of L_2 such that

- $\mathcal{B}$ runs in $f(k)n^c$ time.
- $k' = g(k)$.
- $(x, k) \in L_1$ if and only if $(x', k') \in L_2$.

W-hardness: In order to classify parameterized problems as being FPT or not, the W-hierarchy is defined as $\mathsf{FPT} \subseteq \mathsf{W}[1] \subseteq \mathsf{W}[2] \subseteq \cdots \subseteq \mathsf{XP}$ using Boolean circuits and parameterized reductions. A parameterized language $L \subseteq \Sigma^* \times \mathbb{N}$ is said to be W[i]-hard for a given $i \geq 1$ when for every parameterized language $L' \in \mathsf{W}[i]$, there exists a parameterized reduction from L' to L in $g(k)n^{\mathcal{O}(1)}$ time. It is believed that the subset relations in this sequence are all strict and a parameterized problem that is hard for some complexity class above FPT in this hierarchy is unlikely to be FPT. A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is said to be para-NP-hard if it is not in XP unless $P = \mathsf{NP}$. The complexity classes FPT and para-NP can be viewed as the parameterized analogues of P and NP. For more details about W-hierarchy, please refer to [15]. There are problems that are W-hard but contained in XP.

A concept closely associated with fixed-parameter tractability is the notion of kernelization.

Definition 4 (Kernelization) Let $L \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized language. Kernelization is a procedure that replaces the input instance (I, k) by a reduced instance (I', k') such that

- $k' \leq f(k)$, $|I'| \leq g(k)$ for some computable functions f, g depending only on k.
- $(I, k) \in L$ if and only if $(I', k') \in L$.

The reduction from (I, k) to (I', k') must be computable in $poly(|I| + k)$ time. If $f(k) + g(k) = k^{\mathcal{O}(1)}$ then we say that L admits a *polynomial kernel.*

It is well-known that a decidable parameterized problem is fixed-parameter tractable if and only if it has a kernel though the kernel size is exponential. Kernels are obtained using what are called reduction rules which replace the given input by an equivalent input.

Definition 5 (Soundness/Safeness of Reduction Rule) A reduction rule that replaces an instance (I, k) of a parameterized language L by a reduced instance (I', k') is said to be *sound* or *safe* if $(I, k) \in L$ if and only if $(I', k') \in L$.

Definition 6 (Polynomial parameter transformation (PPT)) Let P_1 and P_2 be two parameterized languages. We say that P_1 is polynomial parameter reducible to P_2 if there exists a polynomial time computable function (or an algorithm) $f : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$, a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that $(x, k) \in P_1$ if and only if $f(x, k) \in P_2$ and $k' \leq p(k)$ where $f[(x, k)] = (x', k')$. We call f to be a polynomial parameter transformation from P_1 to P_2.

Definition 7 (Parameter Preserving Transformation) Let P_1 and P_2 be two parameterized languages. We say that there exists a parameter preserving transformation from P_1 to P_2 if there exists a polynomial time computable function (or algorithm) $f : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$ such that $(x, k) \in P_1$ if and only if $f(x, k) \in P_2$ and $k' \leq ck$ where $f[(x, k)] = (x', k')$ and c is a constant independent of $|x|$ and k. We call f to be a parameter preserving transformation from P_1 to P_2.

The following proposition gives the use of the polynomial parameter transformation for obtaining kernels for one problem from another.

Proposition 1 ([5]) *Let $P, Q \subseteq \Sigma^* \times \mathbb{N}$ be two parameterized problems and assume that there exists a PPT from P to Q. Furthermore, assume that classical version of P is NP-hard and Q is in NP. Then if Q has a polynomial kernel then P has a polynomial kernel.*

In our kernelization algorithms, we use *Menger's Theorem* the statement of which is as follows.

Theorem 1 (Menger's Theorem [12]) *Let $G = (V, E)$ be an undirected graph with a pair of specified vertices $x, y \in V(G)$. There exists a polynomial time algorithm that finds the maximum number of internally vertex disjoint paths from x to y in G.*

Some of our lower bounds are shown using what is called the *Strong Exponential Time Hypothesis (SETH).*

Conjecture 1 (Strong Exponential Time Hypothesis (SETH) [23]) *For every $\epsilon > 0$,* CNF-SAT *cannot be solved in time $\mathcal{O}^*((2-\epsilon)^n)$ time where n is the number of variables in the input formula.*

2.2 Initial Preprocessing Rules

For the algorithms in Section 3 and 4, we use the following well known reduction rules. See Chapter 3, 4 of [8] for safeness of these Reduction Rules. Here ℓ is the size of the solution (fvs) being sought. The following reduction rules can be implemented in polynomial time. We use $\leftarrow$ in the reduction rules (or pseudocodes of algorithm) to mean that the variable on the left is assigned the value in the right of the arrow.

Reduction Rule 1 *If there exists $u \in V(G)$ such that u has a self loop, then $G' \leftarrow G \setminus \{u\}, \ell' \leftarrow \ell - 1$.*

Reduction Rule 2 *If there exists a vertex v such that $deg_G(v) \leq 1$, then $G' \leftarrow G \setminus v, \ell' \leftarrow \ell$.*

Reduction Rule 3 *If there exists a vertex v such that $deg_G(v) = 2$ and $N_G(v) = \{u, w\}$ (u and w could be the same vertex in case of parallel edge (u, v)), then delete the vertex v and add an extra edge (u, w) into G.*

Note that Reduction Rule 3 can create parallel edges or self loops. Also at intermediate steps of our algorithm, u and/or w may be forced not to be in the solution in which case we cannot apply Reduction Rule 3.

Reduction Rule 4 *If there exists an edge (u, v) whose multiplicity is more than 2, then reduce its multiplicity to 2 by deleting the other edges.*

3 Fixed Parameter Algorithms

In this section, we describe fixed-parameter algorithms for FVS when parameterized by the number of vertices of degree more than three, the size of a split vertex deletion set and the size of a cluster vertex deletion set. Before describing all the fixed-parameter algorithms of this section, we provide some general subroutines used in the structural parameterizations we consider in Sections 3.2 and 3.3.

3.1 Disjoint and Special Disjoint Feedback Vertex Set

The following is a problem that we have to solve for both FVS-High-Degree as well as FVS-SVD.

Disjoint Feedback Vertex Set	**Parameter:** $\lvert S_1 \rvert$
Input: An undirected graph $G = (V, E), S_1 \cup S_2 = V(G), S_1 \cap S_2 = \emptyset$, $G[S_1]$ is a forest, S_2 is an independent set and an integer ℓ'.	
Question: Is there a feedback vertex set W of G such that $W \cap S_1 = \emptyset$ and $\lvert W \rvert \leq \ell'$?	

We show in Sections 3.2 and 3.3 that the above problem reduces to the following special case when every vertex in S_2 has exactly three neighbors in S_1 and all such neighbors are in different components of $G[S_1]$.

Special Disjoint Feedback Vertex Set
Input: A simple undirected graph $G = (V, E), S_1 \cup S_2 = V(G), S_1 \cap S_2 = \emptyset$, $G[S_1]$ is a forest, S_2 is an independent set and every vertex of S_2 has exactly 3 neighbors and all are in different components of $G[S_1]$.
Goal: Find a minimum feedback vertex set W of G such that $W \cap S_1 = \emptyset$.

The following Lemma due to Kociumaka and Pilipczuk [28] uses matroid techniques to give a polynomial time algorithm for the above problem.

Lemma 1 *[28] Let (G, S_1, S_2) be an instance of* Special Disjoint Feedback Vertex Set. *Then there exists a polynomial time algorithm that solves the problem.*

We use this polynomial time algorithm as a subroutine in some of our fixed-parameter tractable algorithms.

3.2 Feedback Vertex Set Parameterized by the number of vertices of degree more than 3

FVS-High-Degree	**Parameter:** $\|\{u \in V(G) \| deg_G(u) > 3\}\| \leq k$
Input: An undirected multigraph $G = (V, E)$ and an integer ℓ.	
Question: Does G have a feedback vertex set of size at most ℓ?	

We prove that this problem is fixed parameter tractable and has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. Let $S = \{u \in V(G) | deg_G(u) > 3\}$. Throughout the Sections 3 and 4, we use F to denote $G[V(G) \setminus S]$ (or in other words $G \setminus S$).

Fixed Parameter Algorithm

Theorem 2 *There exists an algorithm that runs in $\mathcal{O}(2^k \cdot n^{\mathcal{O}(1)})$ time for* FVS-High-Degree *problem.*

Proof First we apply Reduction Rules 1, 2, 3, 4 in sequence. It is easy to see that these reduction rules do not increase the degree of any vertex. So, the parameter does not increase when these reduction rules are applied. Once the graph has minimum degree three, the algorithm works as follows (see Algorithm 1 for a detailed pseudo-code).

We guess a subset $S' \subseteq S$ that intersects S with an ℓ sized feedback vertex set we are seeking for. If $G[S \setminus S']$ has a cycle, then we move on to the next guess. Otherwise, let $S'' = S \setminus S'$ and $G[S'']$ is a forest. We update our budget from ℓ to ℓ'. We denote $\ell' = \ell - |S'|$. Now, we aim to find a *minimum feedback vertex set* D of $G \setminus S'$ such that $S'' \cap D = \emptyset$.

Note that every vertex in F has degree at most three in G and also in $G \setminus S'$. Now, we subdivide every edge $(u, v) \in E(F)$, by adding a new vertex $e_{u,v}$ and we add $e_{u,v}$ to S''. As the cycle structures are preserved, this subdivision creates an equivalent instance where our solution is allowed to contain vertices from F only as a feedback vertex set need not pick the newly created vertices. Let the resulting graph be G''. Let $T' = S'' \cup \{e_{u,v} | (u, v) \in E(F)\}$. Note that u and v are the only two neighbors of $e_{u,v}$. Hence we have that for every vertex $u \in V(F)$, u has no neighbor in F. Let $I = V(G'') \setminus T'$. Note that I is an independent set. Our goal is to find a feedback vertex set of G'' of at most ℓ' vertices that is disjoint from T'.

Note that after deleting S', some of the vertices of I can become degree at most 2 in G''. So, we apply Reduction Rule 2 to get rid of the vertices having degree at most one. As discussed in Subsection 2.2, Reduction Rule 3 is not applicable when the neighbors of the degree two vertex are forced to be not in the solution we seek. Hence, we have the following two special rules (Reduction Rule 5 and Reduction Rule 6) to deal with degree two vertices in this case. These two reduction rules are also discussed in [28].

Reduction Rule 5 *If there exists $u \in I$ such that $G''[T' \cup \{u\}]$ has a cycle, then delete u from G'' and reduce ℓ' by 1.*

Safeness of this rule is easy to see. Notice that Reduction Rule 5 is also applicable when a vertex $u \in I$ has exactly two neighbors in T' that are in the same component of $G''[T']$. Also note that this reduction rule gets rid of all the multiple edges from the graph.

It is possible that after exhaustive application of Reduction Rule 5, there can exist a vertex $u \in I$ such that u has exactly 2 neighbors in T' and those are in different components of $G''[T']$.

Reduction Rule 6 *If there exists a vertex $u \in I$ such that u has exactly 2 neighbors in T' and those two neighbors are in different components of $G''[T']$, then move u to T'.*

Lemma 2 *Reduction Rule 6 is safe.*

Proof Reduction Rule 6 is applied only when Reduction Rules 2 and 5 are not applicable. Then $G'[T' \cup \{u\}]$ does not create a cycle for any vertex $u \in I$. The intuition behind this reduction rule is that there exists an optimal solution that does not contain u. Suppose that some optimal solution D contains u. Let t_1, t_2 be the two neighbors of u in $G''[T']$. As D is a minimal feedback vertex set, $G'' \setminus (D \setminus \{u\})$ has a cycle containing u (and hence t_1 and t_2) and no other vertex of D. In this cycle, there is a path between t_1 and t_2 not containing u. But t_1 and t_2 are in different components of $G''[T']$, and hence that path must contain a vertex $u' \neq u$ from I. Then $(D \cup \{u'\}) \setminus \{u\}$ is also a minimum feedback vertex set of G''. □

When Reduction Rules 2, 5, 6 are not applicable, our problem reduces to the Special Disjoint Feedback Vertex Set problem which is polynomial time solvable due to Lemma 1.

Now, if the size of the solution W returned by Lemma 1 is at most ℓ', then we output Yes. Otherwise we repeat the above steps for another subset of S. There are at most $2^{|S|}$ many such subset S' of S and after guessing the subset and some preprocessing the problem is polynomial time solvable. It is easy to see that all the above mentioned reduction rules can be implemented in polynomial time. If for every subset of S, it is seen that the solution size is more than ℓ', then we output No. Therefore, we have an algorithm that runs in time $\mathcal{O}^*(2^k)$. Please refer to Algorithm 1 which gives a pseudocode of the algorithm. This completes the proof. □

But it is still open whether Feedback Vertex Set is FPT when parameterized by the number of vertices whose deletion results in a graph of degree at most 3. Note that this parameter (the minimum number of vertices whose deletion results in a (sub)-cubic graph) is provably smaller than the number of vertices with degree more than 3 of the graph. For, if we remove all vertices with degree more than 3, then the resulting graph becomes (sub)-cubic. However it is possible that among them, some of them become degree at most 3 even when we remove some of their neighbors.

We show that the related problem of Feedback Vertex Set when parameterized by the number of *edges* whose removal results in a graph of degree at most 3, is fixed-parameter tractable. This follows as a consequence of Theorem 2.

Corollary 1 *Let $G = (V, E)$ be a graph with a set of edges E' such that $G' = (V, E \setminus E')$ is a (sub)-cubic graph. Then,* Feedback Vertex Set *is fixed parameter tractable when it is parameterized by $|E'|$.*

Proof As deletion of E' results in a (sub)-cubic graph, the end points of E' are the only vertices of G that have degree at least four. Now, the number of end points of E' is at most $2|E'|$. So, the number of vertices of G with degree more than three is at most $2|E'|$. By Theorem 2, there exists an algorithm that runs in time $\mathcal{O}(2^{2|E'|} \cdot n^{\mathcal{O}(1)})$ and finds a minimum feedback vertex set. So, there exists an algorithm that solves Feedback Vertex Set in time $\mathcal{O}(4^{|E'|} \cdot n^{\mathcal{O}(1)})$ proving the corollary. □

3.3 Fixed-Parameter Algorithm for FVS-SVD

Recall that a graph G is called a *split graph* if $V(G) = C \uplus I$ where C is a clique and I is an independent set. A set of vertices $S \subseteq V(G)$ is called a *split vertex deletion set* if $G \setminus S$ is a *split graph*. In this section, we provide a fixed-parameter algorithm when feedback vertex set is parameterized by the size (number of vertices) of a *split vertex deletion set*.

FVS-SVD	**Parameter:** k
Input: An undirected multigraph G, $S \subseteq V(G)$ of size at most k such that $G \setminus S$ is a split graph and an integer ℓ	
Question: Does G have a feedback vertex set of size at most ℓ?	

Note that the size of the *split vertex deletion set* is incomparable to the solution size. We assume that the *split vertex deletion set* is also given with the input. Otherwise we use an algorithm by Cygan and Pilipczuk [10] that runs in $\mathcal{O}(1.2738^k \cdot k^{O(\log_2 k)} \cdot n^{\mathcal{O}(1)})$ time to determine the existence of a split vertex deletion set of size at most k.

We use (G, S, ℓ) to denote the input instance and set $F = G \setminus S$. F is a split graph whose vertices can be partitioned into a clique and an independent set. So, we denote $F = (C, I)$ where $V(F) = C \uplus I$. Note that $S \cup C$ is a feedback vertex set of G as I is an independent set. So, if $\ell \geq |S| + |C|$, then (G, S, ℓ) is an Yes-Instance. Thus the following reduction rule is easy to see.

Algorithm 1: FVS-PARAM-HIGH-DEGREE-VERTICES

input : $G = (V, E)$ and $\ell \in \mathbb{N}$
output : YES if $\exists C \subseteq V(G), |C| \leq \ell$ such that $G \setminus C$ is a forest, NO otherwise

$S \leftarrow \{u \in V(G) | deg_G(u) \geq 4\}$;
$\ell' \leftarrow \ell$;
Apply Reduction Rules 1 2, 3 and 4 exhaustively;
for *every* $S' \subseteq S$ **do**
    if $G[S \setminus S']$ *is a forest* **then**
        $S'' \leftarrow S \setminus S'$;
        $\ell' \leftarrow \ell - |S'|$;
        $F = G \setminus S$;
        $T = \emptyset$;
        for *each* $(u, v) \in E(F)$ **do**
            $T \leftarrow T \cup \{e_{u,v}\}$;
        $T' \leftarrow T \cup S''$;
        $E' = E(G[S'']) \cup \{(u, e_{u,v}) | (u, v) \in E(F)\}$;
        $G'' = (T', E')$;
        Apply Reduction Rules 2, 5, 6 in this sequence and keep updating ℓ' appropriately.;
        When Reduction Rules 2, 5, 6 are not applicable, run algorithm for Lemma 1 and get W;
        if $|W| \leq \ell'$ **then**
            Return YES
Return NO;

Reduction Rule 7 *If $\ell \geq |S| + |C|$, then return YES.*

So, we can assume that $\ell \leq |S| + |C| - 1$. Now we apply Reduction Rules 1, 2, 3 and 4 exhaustively to make the graph minimum degree three. It is easy to see that these rules do not increase the size of S, the parameter.

Now, our algorithm proceeds as follows. We guess a subset $S' \subseteq S$ and a subset $C' \in \binom{C}{\geq |C|-2}$ that intersect the feedback vertex set of G we are looking for. Note that $|C'| \geq |C| - 2$ as C is a clique. If $G[(S \setminus S') \cup (C \setminus C')]$ is not a forest, then clearly the guess is wrong, we move on to a different subset. So assume that $G[(S \setminus S') \cup (C \setminus C')]$ is a forest. Now, we have to find a subset I' from I with at most $\ell' = \ell - |S'| - |C'| \leq |S| + |C| - 1 - |C| + 2 - |S'| = |S| - |S'| + 1$ vertices such that $G \setminus (S' \cup I' \cup C')$ becomes a forest. We define a measure $\mu(G') = \ell' + d(G')$ where $d(G')$ is the number of components in $G'[S'' \cup C'']$ to analyze the algorithm. We will use μ and $\mu(G')$ interchangeably as we describe the rest of the parts for our algorithm. Let $|S'| = i$. Note that $|S'' \cup C''| = k - i + 2$. But, one (or two vertices) from C also is part of $S'' \cup C''$. So, $d(G') \leq k - i + 1$, and $\ell' \leq k - i + 1$. Hence $\mu \leq 2(k - i) + 2$.

Thus our problem now reduces to the DISJOINT FEEDBACK VERTEX SET problem (defined in Section 3.1) on the instance $(G', (S \setminus S') \cup (C \setminus C'), I, |S| - |S'| + 1)$. Let $C'' = C \setminus C'$ and $S'' = S \setminus S'$. Let $T' = S'' \cup C''$. We use $S'' \cup C''$ and T' interchangeably for the rest of this algorithm. Let G' be the graph obtained after deletion of $C' \cup S'$ from G. Now, we apply Reduction Rules 2, 5, 6 in G' for vertices of I in sequence. Now, we argue that the measure does not increase when Reduction Rules 2, 5, 6 are applied.

Lemma 3 *Application of Reduction Rules 2, 5, 6 does not increase $\mu(G')$.*

Proof Reduction Rule 2 does not increase ℓ. A vertex $u \in I$ on which this rule has been applied can be adjacent to only one vertex of T'. Such a vertex does not increase $d(G')$ also. So, $\mu(G')$ does not increase. Reduction Rule 5 deletes a vertex from I, reduces ℓ, but does not increase $d(G')$. So, $\mu(G')$ does not increase by the application of this rule. Reduction Rule 6 does not increase ℓ, but it decreases d as two components in $G'[T']$ merges into one single component after pushing such a vertex u from I to T'. So, again μ does not increase. □

Now, when Reduction Rules 2, 5, 6 are not applicable, every vertex of I has at least three neighbors that are in T' and all these neighbors are in different components of $G'[T']$.

If there exists a vertex $u \in I$ that has at least four neighbors in $S'' \cup C''$ and all are in different components and we apply the following branching rule.

Branching Rule 1 *If there exists a vertex $u \in I$ such that u has at least four neighbors all of whom are in different components of $G'[T']$, then in one branch, we pick u into the solution and in another branch, we push the vertex u from I to $S'' \cup C''$.*

Clearly the branching rule is exhaustive. When Branching Rule 1 is applied, in the first branch ℓ decreases by 1. So, $\mu(G')$ drops by 1 in the first branch. In the other branch, $d(G')$ decreases by at least 3 because u has at least 4 neighbors and all the neighbors are in different connected components. So, in the other branch, $\mu(G')$ drops by at least 3 (as four components get merged into a single one). So, we get the following recurrence for this branching rule.

$$T(\mu) \leq T(\mu - 1) + T(\mu - 3)$$

Solving this recurrence, we get

$$T(\mu) \leq 1.4656^{\mu} \leq 2.148^{k-i+1}.$$

If for every vertex $u \in I$, u has exactly three neighbors all in different components of $G'[T']$, then the instance is an instance of the Special Disjoint Feedback Vertex Set problem which is polynomial time solvable due to Lemma 1. So, we get an algorithm with running time $\mathcal{O}^*(2.148^{k-i+1})$ for Disjoint Feedback Vertex Set problem implying the following lemma.

Lemma 4 *Given an instance (G, X, Y, ℓ) of* Disjoint Feedback Vertex Set *problem where $\ell \leq |X| - 1$, there exists an algorithm that runs in $\mathcal{O}(2.148^{|X|} \cdot n^{\mathcal{O}(1)})$ time and solves this problem.*

As this algorithm for Disjoint Feedback Vertex Set is run over all subsets of S, the algorithm runs in time $\sum_{i=0}^{k} \binom{k}{i} 2.148^{k-i+1} \cdot n^{\mathcal{O}(1)}$ which is $\mathcal{O}(3.148^k \cdot n^{\mathcal{O}(1)})$. Given k, finding a split vertex deletion set of size at most k takes $\mathcal{O}^*(1.2738^k \cdot k^{\mathcal{O}(\log_2 k)})$ time [10]. So, we have the following theorem.

Theorem 3 Feedback Vertex Set *parameterized by* Split Vertex Deletion Set *(* FVS-SVD *) can be solved in $\mathcal{O}(3.148^k \cdot n^{\mathcal{O}(1)})$ time.*

Now we observe that this algorithm can be generalized even when the vertex set can be partitioned into $c > 1$ cliques. A graph is called a (c, i)-graph (introduced in [29]) if its vertex set can be partitioned into c cliques, and i independent sets.

> FVS-deletion to (c, i)-graph
> **Input:** An undirected graph $G = (V, E), S \subseteq V(G)$ such that $G \setminus S$ is a graph for which $V(G) \setminus S = A_1 \uplus \ldots \uplus A_c \uplus B_1 \uplus \ldots \uplus B_i$ where the induced subgraph on $A_1, \ldots, A_c$ are cliques, and the induced subgraph on $B_1, \ldots, B_i$ are independent sets and an integer ℓ.
> **Goal:** Is there a feedback vertex set of G with at most ℓ vertices?

First we explain how our algorithm for FVS-SVD generalizes to an algorithm with running time $\mathcal{O}(3.148^{k+c} n^{\mathcal{O}(c)})$ when F is a $(c, 1)$-graph. Observe that determining whether a given graph is a $(c, 0)$-graph is the same as determining whether the complement graph is c-colorable, which is NP-hard for $c \geq 3$. There is also an easy polynomial time reduction from recognition of $(c, 0)$-graph to the recognition of $(c, 1)$-graph. So for this problem, we need to assume that the deletion set ($S \subseteq V(G)$) and the partition of $V(G) \setminus S = A_1 \uplus \ldots \uplus A_c \uplus I$ are given with the input.

So, when $c \geq 3$, then we cannot even hope to have an FPT algorithm that outputs a $(c, 0)$ (or a $(c, 1)$) deletion Set. When $c \leq 2$, Kolay and Panolan [29] provided an algorithm with runtime $\mathcal{O}^*(3.314^k)$ to find an $S \subseteq V(G)$ of size at most k such that $G \setminus S$ is a $(c, 1)$-graph.

The algorithm for FVS-deletion to $(c, 1)$-graph follows a similar strategy. Let $G \setminus S = A_1 \uplus \ldots \uplus A_c \uplus B$ where each of $A_1, \ldots, A_c$ induces a clique, $c \geq 1$ and B induces an independent set. Any feedback vertex set of G intersects A_j in at least $|A_j| - 2$ vertices for all $j \in [c]$. If $\ell \geq |S| + \sum_{j=1}^{c} |A_j|$, then it is a yes-instance, and so we modify the Reduction Rule 7 appropriately and assume that $\ell \leq |S| + \sum_{j=1}^{c} |A_j| - 1$. Now we make the minimum degree of G to three by using Reduction Rules 1, 2, 3 and 4.

Let D be a feedback vertex set of G. Then, for every $j \in [c]$, $|D \cap A_j| \geq |A_j| - 2$. So, we have $\prod_{j=1}^{c} (1 + |A_j| + \binom{|A_j|}{2})$, i.e. $\mathcal{O}(n^{2c})$ many choices of intersections of a feedback vertex set of G with $A_1, \ldots, A_c$. Now, fix one such choice $(A'_1, \ldots, A'_c)$ from A_j's. And we guess a subset $S' \subseteq S$ that intersects with the feedback vertex set in G. Note that we have excluded the vertices of $A_j \setminus A'_j$ from the feedback vertex set. If $G[(S \setminus S') \cup (\bigcup_{j=1}^{c} (A_j \setminus A'_j))]$ is not a forest, then we move to the next guess.

Otherwise $G[(S \setminus S') \cup (\bigcup_{j=1}^{c} (A_i \setminus A'_j))]$ is a forest. We update $\ell' = |S| + \sum_{j=1}^{c} |A_j| - 1 - \sum_{j=1}^{c} |A'_j| \leq k + 2c - 1$. Now, our goal is to identify whether there exists a feedback vertex set of size at most ℓ' contained in B, after deletion of vertices of $A'_1, \ldots, A'_c, S'$. This is again the Disjoint Feedback Vertex Set problem where $V(G') = S'' \uplus B$ such that S'' induces a forest, B is an independent set problem. Let $|S'| = p$. As $\ell \leq |S| + \sum_{j=1}^{c} |A_j|$, we have that $\ell' \leq |S| - |S'| + 2c - 1 \leq k - p + 2c - 1$ as at least two vertices from each of $A_1, \ldots, A_c$ have been deleted. Let $S'' = (S \setminus S') \cup (\bigcup_{j=1}^{c} (A_j \setminus A'_j))$. Also, the number of connected components in $G[(S \setminus S') \cup (\bigcup_{j=1}^{c} (A_j \setminus A'_j))]$ is $c(S'')$ which is at most $|S| - |S'| + c - 1 \leq k + c - p - 1$ there are c components of them having at least two vertices. We define a measure $\mu(G') = \ell' + c(S'') \leq 2k + 3c - 2p - 1$. Now we run the Disjoint Feedback Vertex Set algorithm as described earlier on this instance. So, we get an algorithm with running time $\mathcal{O}^*(1.4656^{\mu})$ which is $\mathcal{O}^*(3.148^c \cdot 2.148^{k-j-1})$ The total running time of the algorithm is as follows.

$$3.148^c n^{2c+\mathcal{O}(1)} \cdot \sum_{j=0}^{k} \binom{k}{j} \cdot 2.148^{k-j+1} = n^{2c+\mathcal{O}(1)} 3.148^c \cdot \sum_{j=0}^{k} \binom{k}{k-j} \cdot 2.148^{k-j}$$

$$= 3.148^{k+c} \cdot n^{2c+\mathcal{O}(1)}$$

Theorem 4 FVS deletion to $(c,1)$-graph *admits an algorithm with running time* $\mathcal{O}(3.148^{k+c} \cdot n^{2c+\mathcal{O}(1)})$, *i.e. the problem admits an* XP *algorithm when parameterized by c and k.*

Note that the same algorithm without the call to Disjoint Feedback Vertex Set works when B, the independent set is empty, i.e. $i = 0$. Thus we have an algorithm running in time $\mathcal{O}(2^k n^{2c+\mathcal{O}(1)})$ when $i = 0$.

Theorem 5 FVS deletion to $(c,0)$-graph *admits an algorithm with running time* $\mathcal{O}(2^k \cdot n^{2c+\mathcal{O}(1)})$, *i.e. the problem admits an* XP *algorithm when parameterized by c and k.*

The above algorithm (for $i = 0$) generalizes the XP algorithm for FVS parameterized by the number c of cliques in a vertex clique cover by Jansen et al. [26]. Here we have generalized for the case when G is k away from a graph with vertex clique cover number c.

FVS-Deletion $(c,0)$-graph **when parameterized by** c **and** k**:** Jansen et al. [26] also proved that FVS parameterized by the number of cliques in a vertex clique cover (a vertex clique cover is a set of cliques of a graph such that every vertex of the graph participates in some clique) of a graph is $\mathsf{W}[1]$-hard. So it follows that the same is true even when G is k away from a graph with clique cover number c.

Observation 1 FVS-deletion to $(c,0)$-graph *is* $\mathsf{W}[1]$*-hard when parameterized by c and k.*

It is easy to see that deletion distance to $(c,1)$-graph is provably smaller than deletion distance to $(c,0)$-graph. So, we have the following observation.

Observation 2 *For* $i \leq 1$*,* FVS-deletion to (c,i)-graph *is* $\mathsf{W}[1]$*-hard when parameterized by c and k.*

Finally, we complete the picture for FVS deletion to (c,i)-graph for $i \geq 2$ by an easy observation. As Feedback Vertex Set on bipartite graphs is NP-Complete, we have the following.

Observation 3 *When* $c = 0, i \geq 2$*,* Feedback Vertex Set *is* NP*-Complete on* (c,i) *graphs. So, when the parameter is either c or k and* $i \geq 2$*,* FVS-deletion to (c,i)-graph *is para-*NP*-hard. When* $c = 0, i = 2$*, then* FVS-deletion to $(0,2)$-graph *is exactly the* Feedback Vertex Set *problem parameterized by the size of a smallest odd cycle transversal.*

3.4 FVS Parameterized by Cluster Vertex Deletion Set

Now, we consider the parameter which is the distance to a cluster graph. Here, S is the modulator, $|S|$ is the parameter and $G \setminus S$ is a collection of cliques, i.e. it is a $(c, 0)$ graph with the property that there are no edges between the c cliques. $G \setminus S$ is also called a cluster graph, and $S \subseteq V(G)$ is called a "cluster vertex deletion set". A set S is called a cluster vertex deletion set if every component of $G \setminus S$ is a clique. As before, we can omit the assumption that the cluster vertex deletion set (S) is given along with the input as there is an algorithm by Boral et al. [6] that runs in $\mathcal{O}(1.9106^k(n+m))$ time and either outputs a cluster vertex deletion set of size at most k (if one exists) or says that no cluster vertex deletion set of size at most k exists.

FVS-CVD	**Parameter:** k
Input: An undirected multigraph G, $S \subseteq V(G)$ of size at most k such that every component of $G \setminus S$ is a clique and an integer ℓ	
Question: Does G have a feedback vertex set of size at most ℓ?	

Central to our algorithm is a theorem (Theorem 4.5 of Bodlaender et al. [2]) which is used in the proof of our result. Before stating that theorem, we need the notions of a path decomposition, more specifically a nicer path decomposition of a graph and pathwidth of a graph.

Definition 8 (Path Decomposition) Let $G = (V, E)$ be an undirected graph. A path decomposition of G is a sequence of bags $\mathcal{P} = (X_1, \ldots, X_q)$ where $\forall i \in [q], X_i \subseteq V(G)$ such that the following properties are satisfied.

1. For every vertex $u \in V(G)$, there exists $i \in [q]$ such that $u \in X_i$.
2. For every edge $(u, v) \in E(G)$, there exists $i \in [q]$ such that $u, v \in X_i$.
3. For any vertex $u \in V(G)$, if $u \in X_i \cap X_k$ for some $i \leq k$, then $u \in X_j$ for all $i \leq j \leq k$.

Definition 9 (Pathwidth) Let $\mathcal{P} = (X_1, \ldots, X_q)$ be a path-decomposition of G. Then, the width of $\mathcal{P}$ is denoted as $pw(\mathcal{P}) = max_{i \in [q]}\{|X_i| - 1\}$. The path-width of G is the minimum width over all possible path-decompositions of G. More specifically, if $\mathring{A}$ is the set of path decompositions of G, then $pw(G) = min_{\mathcal{P} \in \mathring{A}} pw(\mathcal{P})$.

A path decomposition is called *nice* if all of its bags are one of the following types.

- Introduce Bag: A bag X_{i+1} is called an *introduce bag* if $X_{i+1} = X_i \cup \{u\}$ where $u \notin X_i$.
- Forget Bag: A bag X_{i+1} is called a *forget bag* if $X_{i+1} = X_i \setminus \{u\}$ where $u \in X_i$.
- $X_1 = X_q = \emptyset$.

We also have the following lemma which is Lemma 7.2 of [8].

Lemma 5 *If a graph G admits a path decomposition of width p, then it also admits a nice path decomposition of width p. Moreover, given a path decomposition $\mathcal{P} = (X_1, \ldots, X_q)$ of G of width at most p, one can in time $\mathcal{O}(p^2 max(q, |V(G)|))$ compute a nice path decomposition of G of width p.*

Furthermore, we can convert a nice path decomposition into a *nicer path decomposition* in polynomial time where the bags are of three types (see again [8]).

- Introduce Vertex Bag: A bag X_{i+1} is called an *introduce vertex bag* if $X_{i+1} = X_i \cup \{u\}$ where $u \notin X_i$.
- Introduce Edge Bag: We say a bag X_i is *introduce edge bag* if it is labelled by an edge (u, v) and $X_i = X_{i-1}$. Note that in such case $u, v \in X_i$. Also note that an edge is introduced exactly once in the entire decomposition.
- Forget Bag: A bag X_{i+1} is called a *forget bag* if $X_{i+1} = X_i \setminus \{u\}$ where $u \in X_i$.

Let $V(G_i) = (\bigcup_{j=1}^{i} X_i)$ and $E(G_i) = \{(u, v) \in E(G) | (u, v)$ is introduced in one of the bags X_j for some $j \in [i]\}$. It is easy to see that a given a nice path decomposition can be converted into a *nicer path decomposition* also in $\mathcal{O}(p^2 max(q^2, |V(G)|^2))$ time (See [8]).

Now, we get back to our problem FVS-CVD. Let $C_1, \ldots, C_c$ be the set of connected components of $G \setminus S$. We argue that $(S \cup C_1, \ldots, S \cup C_c)$ is a path-decomposition.

Lemma 6 *For the problem* FVS-CVD, *$(S \cup C_1, \ldots, S \cup C_c)$ forms a path-decomposition of G.*

Proof Consider any vertex $u \in V(G)$. If $u \in S$, then $u \in S \cup C_i$ for all $i \in [q]$. So, both property 1 and 3 are satisfied for all $u \in S$. If $u \in C_i$ for some $i \in [c]$, then $u \in S \cup C_i$ and there is exactly one $i \in [c]$ in which u exists. So, property 1 and 3 are satisfied for all $u \in V(G)$. Consider any edge $(u, v) \in E(G)$. If $u, v \in S$, then $u, v \in S \cup C_1$. If $u \in S, v \in C_i$ for some $i \in [c]$, then $u, v \in S \cup C_i$. If $u, v \in C_i$, then $u, v \in S \cup C_i$. So, property 2 is satisfied. □

Now we are ready to state Theorem 4.5 of Bodlaender et al. [2].

Theorem 6 *Let G be a graph given with a nicer path decomposition $\mathcal{P} = (X_1, \ldots, X_q)$ such that the intersection of any feedback vertex set with any bag X_i has at most $2^k \cdot (|X_i| - k)^{\mathcal{O}(1)}$ possibilities. Then,* Feedback Vertex Set *can be solved in $\mathcal{O}(5^k \cdot n^{\mathcal{O}(1)})$ time.*

Now, notice that in our path decomposition (Lemma 6), a feedback vertex set of G can intersect any bag $(S \cup C_i)$ in at most $2^{|S|} \cdot |C_i|^2$ possibilities, as any feedback vertex set intersects C_i in at least $|C_i - 2|$ vertices. It can be easily shown that this path decomposition can be converted into a nicer path decomposition of the same width by introducing (and forgetting) the vertices of each of the C_i's one by one in polynomial time. Each bag in that nicer path decomposition is $S \cup C'_i$ where C'_i is a set of vertices of C_i. It is easy to see that in this nicer path decomposition too, a feasible feedback vertex set of G can intersect any bag $(S \cup C'_i)$ in at most $2^{|S|} \cdot |C'_i|^3$ vertices. Now, using this property and Theorem 6 we get the following theorem.

Theorem 7 FVS-CVD *admits an algorithm that runs in $\mathcal{O}(5^k \cdot n^{\mathcal{O}(1)})$ time.*

4 Kernelizations Algorithms: Upper and Lower Bounds

In the previous section, we have discussed FPT algorithm for FVS when it is parameterized by the size of a cluster vertex deletion set and the size of a set whose deletion results in a $(c, 1)$-graph where c is a constant. In this section, we discuss upper and lower bounds of kernelization for Feedback Vertex Set for structural parameterizations. It is known by Bodlaender et al. [4] that Feedback Vertex Set parameterized by the size of a clique deletion set (number of vertices whose deletion results in a clique) has no polynomial kernel unless NP ⊆ coNP/poly. Any clique deletion set is also a (c, i)-deletion set with $c = 1, i = 0$. So, it follows easily that Feedback Vertex Set does not have a polynomial kernel (unless NP ⊆ coNP/poly) when it is parameterized by the size of a set whose deletion results in a $(c, 1)$-graph (or disjoint union of cliques) even when c is a constant and $c \geq 1$.

4.1 Kernelization Lower Bound for FVS-High-Degree

We restate the problem definition for FVS-High-Degree. We prove that this problem has no polynomial kernel unles NP ⊆ coNP/poly.

FVS-High-Degree	**Parameter:** k
Input: An undirected graph G such that $\lvert\{u \in V(G) \mid deg_G(u) > 3\}\rvert \leq k$ and $\ell \in \mathbb{N}$.	
Question: Does G have a feedback vertex set of size at most ℓ?	

Now, to show that FVS-High-Degree has no polynomial kernel unless NP ⊆ coNP/poly, we use the following theorem.

Theorem 8 *([19]) Let ϕ be a boolean formula in CNF form with n variables and m clauses. CNF-SAT parameterized by n has no polynomial kernel unless* NP ⊆ coNP/poly.

Jansen et al. [26] provided a polynomial parameter transformation from CNF-SAT parameterized by the number of variables n, to Feedback Vertex Set parameterized by deletion distance to Mock-Forest. Recall that a graph is a mock-forest if every vertex participates in at most one cycle. In that construction, the size of the deletion distance to Mock-Forest is at most $4n$. In the same construction, the number of vertices of the graph whose degree is at least 4 is $2n$. So, the same transformation is also a polynomial parameter transformation from CNF-SAT to FVS-High-Degree. So, we have the following lemma. A brief proof sketch is presented here for completeness. For details, see Section 6 in [26].

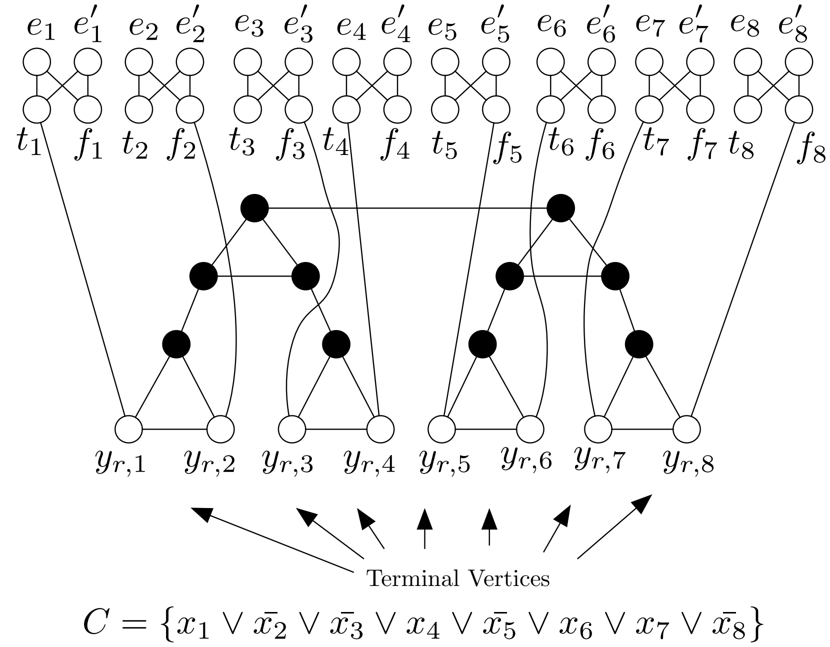

Fig. 2: Illustration of Gadget Constructions for a clause C with 8 literals. This is the r'th copy of clause gadget for clause C. Literal x_i (or $\bar{x_i}$) is the i'th literal and it corresponds to the vertex $y_{r,i}$

Lemma 7 *Let (ϕ, n) be an input instance for CNF-SAT with n variables. There is a polynomial time reduction that transforms (ϕ, n) to $(G_\phi, 2n)$ where G_ϕ is an instance of* FVS-High-Degree *and $2n$ is the number of vertices having degree more than 3 in G_ϕ.*

Proof (Sketch) The reduction from CNF-SAT parameterized by number of variables to FVS parameterized by deletion distance to mock-forest is from Theorem 6 [26]. Let the number of variables in the formula ϕ be n and the number of clauses be m. Without loss of generality we can assume that the number of literals in each clause is a power of 2 (otherwise just duplicate literals). We provide a clause gadget of height j where $d = 2^j$. We create n^2 many copies for this clause gadget. In this gadget, the terminal vertices are the corresponding vertices of literals (see Figure 2). For clause C of length d with its r'th copy, we name literals as $y_{r,1}, \ldots, y_{r,d}$. And we create a variable gadget for variable x_i as a cycle of 4 vertices. Let $\{t_i, f_i, e_i, e'_i\}$ (See Figure 2) be those vertices. We define $S = \bigcup_{i=1}^{n} \{t_i, f_i\}$. Let $y_{r,j}$ be the vertex corresponding to the j'th literal of clause C. Let the variable corresponding to that literal is x_i. Then, if $y_{r,j}$ corresponds to the literal $\bar{x_i}$, then we connect $y_{r,j}$ with f_i by an edge. Otherwise the literal corresponding to $y_{r,j}$ is x_i. In such case, we connect $y_{r,j}$ with t_i by an edge. We do the same for each of the n^2 copies of the clause gadget. We set $\ell = n^2 \sum_{i=1}^{m} (d_i - 2)$ where d_i is the number of literals in i'th clause. Any terminal vertex is adjacent to exactly one vertex which is either t_i or f_i for some i (See Figure 2 for illustration). And every vertex in clause gadget has degree exactly 3 in the whole graph, while e_i, e'_i have degree exactly 2. Therefore, any vertex in $G \setminus S$ has degree at most 3 in the whole graph. It can be shown that any feedback vertex set of G of size ℓ (which is optimal) must contain either t_i or f_i for all $i \in [n]$ (See [26] for details). This feedback vertex set can be transformed into a satisfying assignment of the formula ϕ and vice versa. □

From Theorem 8, Lemma 7 and Proposition 1, we have that FVS-High-Degree has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. Consider the related problem in which Feedback Vertex Set is parameterized by the number of vertices whose deletion results in a (sub)-cubic graph. The existence of an FPT algorithm for that problem still remains open. Note that number of vertices that has degree more than three is provably larger than the number of vertices whose deletion results in a (sub)-cubic graph. So, we can conclude that FVS has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ when it is parameterized by the deletion distance to (sub)-cubic graph (even if it turns to be FPT). So, we have the following theorem.

Theorem 9 FVS-High-Degree *has no polynomial kernel unless* NP $\subseteq$ coNP/poly*. In particular,* Feedback Vertex Set *also has no polynomial kernel unless* NP $\subseteq$ coNP/poly *when parameterized by the deletion distance to (sub)cubic graph.*

Note that in this polynomial time reduction, the parameter is transformed from n to $2n$. So, assuming Conjecture 1 and Lemma 7, we have the following corollary.

Corollary 2 *For any $\epsilon > 0$, there does not exist any algorithm for* FVS-High-Degree *with running time $\mathcal{O}((\sqrt{2}-\epsilon)^k \cdot n^{\mathcal{O}(1)})$ (where k is the number of vertices of degree more than three) unless SETH (or Conjecture 1) fails.*

Proof Suppose, there exists an algorithm $\mathcal{B}$ for FVS-High-Degree running in $\mathcal{O}^*((\sqrt{2}-\epsilon)^k)$ time. Now, we take an instance ϕ of CNF-SAT problem. Let ϕ has k variables. We apply the polynomial time reduction from CNF-SAT to FVS-High-Degree. We get the graph G_ϕ where the number of vertices with degree more than three is $2k$. We run algorithm $\mathcal{B}$ on G_ϕ. If $\mathcal{B}$ outputs "No", then we say that ϕ is unsatisfiable. Otherwise we get a feedback vertex set of size at most ℓ. Clearly, the feedback vertex set is optimal. From this feedback vertex set, we can construct an assignment using Lemma 7 for ϕ which satisfies ϕ. Now $(\sqrt{2}-\epsilon)^{2k}$ is $\mathcal{O}((2-\epsilon')^k)$ for some $\epsilon' > 0$ as $\epsilon > 0$. So, we get an algorithm with running time $\mathcal{O}^*((2-\epsilon')^k)$ for CNF-SAT contradicting SETH. □

4.2 Some General Reduction Rules

In the next two subsections, we describe the kernelization algorithms for FVS-Pseudo-Forest and FVS-Mock-d-Forest. Here, we first describe the common rules and prove some general properties that we use to get the kernels. Recall that a pseudo-forest is a graph where every component has at most one cycle and a mock-forest is a graph where every vertex participates in at most one cycle. A mock-forest is called *mock-d-forest* if each of its connected components have at most d cycles. We use F to denote $G \setminus S$ for both FVS-Pseudo-Forest and FVS-Mock-d-Forest problems, i.e. F is either a pseudo-forest or a mock-d-forest.

Let $C_1, \ldots, C_r$ be a collection of r cycles in a graph G such that for all $i \neq j, V(C_i) \cap V(C_j) = \{u\}$ for some $u \in V(G)$. Then we say that G has an *r-flower with core u* and $C_1, \ldots, C_r$ is an *r-flower with core u*.

Reduction Rule 8 (Flower Rule) *Let $x \in S$ and F' be an induced subgraph of F such that there are c cycles in F containing some vertex of F'. If $G[\{x\} \cup V(F')]$ has an $(|S|+c+1)$-flower with core x, then*

- $G' \leftarrow G \setminus \{x\}$.
- $\ell' \leftarrow \ell - 1$

Lemma 8 *Reduction Rule 8 is safe and can be implemented in polynomial time.*

Proof Safeness of the reduction rule is based on the fact that any minimum feedback vertex set of G must contain x when the precondition of the reduction rule is satisfied. Suppose not. Then, there exists a minimum feedback vertex set D such that $x \notin D$ and the precondition applies. As D is a minimum feedback vertex set, we know that $|D| = fvs(G) \leq |S| + fvs(F)$. Now, as $x \notin D$, D must contain at least $|S|+c+1$ vertices from F'. Note that $fvs(G[V(F') \cup \{x\}]) \leq |S| + c$. Let $fvs(F \setminus F') = c'$. It means that there are c' cycles in F that do not contain any vertex from F'. So, $fvs(F) = c' + c$. Now D has to pick at least one vertex from each of the cycles in $F \setminus F'$ also. So, D must pick c' other vertices from $F \setminus F'$. So, $|D| \geq |S| + c + 1 + c' = |S| + fvs(F) + 1$. This is a contradiction to the fact that D is an optimal feedback vertex set. So, the reduction rule is safe.
To check whether this flower rule is applicable for a vertex $x \in S$, we need to run a polynomial time algorithm that detects a flower, if exists, with core x, using Gallai's Theorem (See Chapter 9 in [8] for more details). □

Reduction Rule 9 (Vertex-Disjoint-Path-Rule) *Let $x, y \in S$ such that (x, y) is not a double-edge and F' be an induced subgraph of F. Let c be the number of cycles in F containing the vertices of F'. If there are at least $|S|+c+2$ pairwise internally vertex disjoint paths from x to y in $G[V(F') \cup \{x, y\}]$, then make (x, y) a double-edge.*

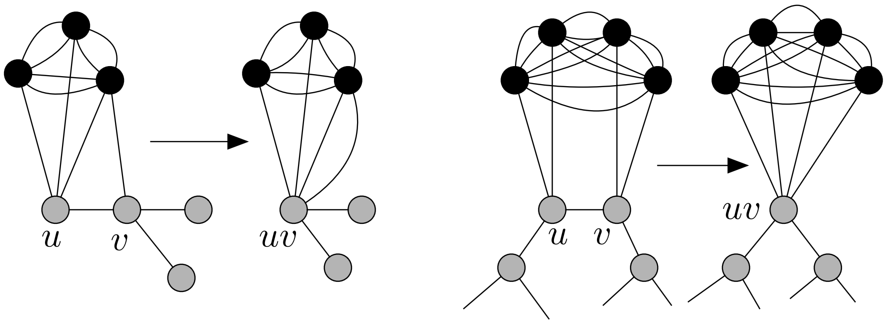

Fig. 3: An illustration of Reduction Rule 10

Lemma 9 *Reduction Rule 9 is safe and can be implemented in polynomial time.*

Proof It suffices to prove that any minimum feedback vertex set of G contains at least one vertex from x and y. Suppose not. Then, there exists a minimum feedback vertex set D such that $x, y \notin D$. We also know at $|D| = fvs(G) \leq |S| + fvs(F)$ as D is a minimum feedback vertex set. As D is a feedback vertex set of G, there is at most one path between x and y in $G \setminus D$. By the precondition of the reduction rule, there are $|S| + c + 2$ internally vertex disjoint paths in G. So, D must contain at least $|S| + c + 1$ vertices from those paths. Now c cycles of F contain the vertices of F'. Let $fvs(F \setminus F') = c'$. So, $fvs(F) = c + c'$. So, there are c cycles that do not contain any vertex of F'. So, D must contain c' other vertices from $F \setminus F'$. Then, $|D| \geq |S| + c + 1 + c' = |S| + fvs(F) + 1$ which is a contradiction to the fact that D is an optimal feedback vertex set. So, this reduction rule is safe.
Checking whether the precondition is satisfied or not requires to compute the number of internally vertex disjoint paths from x to y in the subgraph $G[F' \cup \{x, y\}]$. We can find this in polynomial time using Theorem 1 (Menger's Theorem). □

Though the above reduction rule does not decrease the size of the graph, it helps to apply some other reduction rules, e.g. Reduction Rule 10. We call a cycle C in F as 2-cycle if C is a double-edge.

Reduction Rule 10 (Edge-Rule) *1. If there exists a vertex $u \in F$ such that $deg_F(u) = 0$, and there is no double edge attached to u and if $N_G(u) \cap S$ forms a double-clique, then $G' \leftarrow G \setminus \{u\}$.*
2. If there exists $u \in F$ such that $deg_F(u) = 1$, and there is no double edge attached to u and $N_G(u) \cap S$ forms a double-clique, then $G' \leftarrow G/(u, v)$ (recall the notation of contracting an edge) where $\{v\} = N_G(u) \cap F$. However, the multiple edges created because of contraction should not be deleted.
3. Let $(u, v) \in E(F)$ such that (u, v) is not a double-edge and both $deg_F(u) = deg_F(v) = 2$. So, u and v have exactly one other neighbor in F each with multiplicity 1. If $(N_G(u) \cap S) \cap (N_G(v) \cap S) = \emptyset$, no double edge is attached to either of u or v and $G[(N_G(u) \cup N_G(v)) \cap S]$ forms a double-clique, then $G' \leftarrow G/(u, v)$. However, multiple edges or self loops created because of contraction should not be deleted.
(See Figure 3 for illustration)

Lemma 10 *Reduction Rule 10 is safe and can be implemented in polynomial time.*

Proof We formally prove the safeness in the given order.

1. The intuition is that as u becomes a degree 1 vertex after having all but at most one vertex from $N_G(u) \cap S$. So, we can delete u.
   ($\Rightarrow$) If D is a feedback vertex set of G of size at most ℓ, then $D \setminus \{u\}$ is a feedback vertex set of $G \setminus \{u\}$. Therefore, $G \setminus \{u\}$ has a feedback vertex set of size at most ℓ.
   ($\Leftarrow$) Let D be a minimum feedback vertex set of $G \setminus \{u\}$. Note that there is no double edge attached to u and $N_G(u) \cap S$ is a double clique. Then, any feedback vertex set of $G \setminus \{u\}$ must contain at least $|N_G(u) \cap S| - 1$ many vertices. Now consider $S' = S \setminus D$. Clearly $N_G(u) \cap S'$ has at most one vertex say x and (u, x) is not a double edge. Therefore, adding u into $V(G) \setminus D$ does not create any cycle. Therefore, D is a minimum feedback vertex set of G as well.

2. The intuition is that as u becomes a degree-2-vertex after having all but at most one vertex from $N_G(u) \cap S$, we can contract the edge (u, v).
   ($\Leftarrow$) Let D be a minimum feedback vertex set of $G' = G/(u, v)$. Since $N_G(u) \cap S$ forms a double clique, D must contain at least $|N_G(u) \cap S| - 1$ vertices from $N_G(u) \cap S$. Let $V' = V(G') \setminus D$. $G'[V']$ be a forest. Suppose $G[V' \cup \{u\}]$ has a cycle. Then, note that u can have at most 2 neighbors in $V' \cup \{u\}$. One of them is v and other one is exactly one vertex $x \in N_G(u) \cap S$. Then, there exists a path from v to x in G. Note that the same path exists in G' where uv be the vertex resulted after contracting edge (u, v). Moreover, (v, x) forms an edge in G' because of contraction. Therefore, $G[V']$ contains a cycle which is a contradiction. Therefore, $G[V' \cup \{u\}]$ is also a forest and hence D is a feedback vertex set of G as well.
   ($\Rightarrow$) Let D be a feedback vertex set of G. Then $G \setminus D$ is a forest. If both the vertices u and v are in $G \setminus D$, then the edge (u, v) exists in $G \setminus D$, then $(N_{G \setminus D}(u) \setminus \{v\}) \cap (N_{G \setminus D}(v) \setminus \{u\}) = \emptyset$. Then contracting the edge (u, v) does not make $(G \setminus D)/(u, v)$ into a graph containing some cycle. Otherwise one of u and v is in D. Now, we say that $(D \setminus \{u, v\}) \cup \{uv\}$ is a feedback vertex set of $G/(u, v)$. $|(D \setminus \{u, v\}) \cup \{uv\}| \leq |D|$. So, the reduction rule is safe.
3. The intuition is that either u or v becomes a vertex of degree 2 after having all but at most one vertex from $(N_G(u) \cup N_G(v)) \cap S$. So, we can contract the edge (u, v).
   ($\Rightarrow$) Let D be a minimum feedback vertex set of G. By a similar argument as before, $G/(u, v)$ also has a feedback vertex set of $|D|$ vertices.
   ($\Leftarrow$) Let (u, v) is an edge satisfying the precondition of the rule such that $N_G(u) \cap N_G(v) \cap S = \emptyset$ and $(N_G(u) \cup N_G(v)) \cap S$ forms a double-clique. Then, let D be a feedback vertex set of $G' = G/(u, v)$ where uv be the contracted vertex as a result of contracting the edge (u, v). Note that D contains at least $|(N_G(u) \cup N_G(v)) \cap S| - 1$ vertices from $(N_G(u) \cup N_G(v)) \cap S$. Now there are two cases.
   - If $uv \in D$, then by the precondition of the reduction rule, there is at most one vertex z from S such that $z \in V(G') \setminus D$ and z is a neighbor of uv. Now, either $z \in N_G(u) \cap S$ or $z \in N_G(v) \cap S$ but not both. If $z \in N_G(u) \cap S$, then we say that $(D \setminus \{uv\}) \cup \{u\}$ is a feedback vertex set of G and clearly $|D'| = |D|$. Note that in this case v cannot have more than one neighbor in $G' \setminus D$ and hence $G \setminus D'$ is also acyclic. Similarly, we can prove that when $z \in N_G(v) \cap S$, then $D' = (D \setminus \{uv\}) \cup \{v\}$ is a feedback vertex set of G and again $|D'| = |D|$.
   - If $uv \notin D$, then suppose there exists $z \in N_{G'}(uv) \cap S$. Then, we claim that D is a feedback vertex set of G as well. The reason is that $z \in N_G(u) \cap S$ or $z \in N_G(v) \cap S$ but not both. Therefore, consider $G \setminus D$. Neither u nor v is in D. Then when there is a cycle in $G \setminus D$ consisting of both u and v, then that cycle was there in $G' \setminus D$ consiting of uv which is a contradiction to the fact that D is a feedback vertex set of G'. Therefore, the reduction rule is safe.

This completes the proof. □

Note that variants of Reduction Rules 8, 9 and 10 also appeared in [3] but in different forms. We recall the following concept due to Jansen et al. [26]. The following definition will be used crucially to get kernel upper bounds for FVS-Pseudo-Forest and FVS-Mock-d-Forest.

Definition 10 Let C be a connected component in F that has cycle. Let $X \subseteq N_G(C) \cap S$. We say that C *can be resolved with respect to* X if there exists $A \subseteq C$ such that A is a minimum feedback vertex set of $G[C]$ and for every connected component C' in $C \setminus A$, $|N_G(C') \cap X| \leq 1, |N_G(X) \cap C'| \leq 1$ and $G[(C \setminus A) \cup X]$ has no cycle.

The intuition behind this idea is that if a component C can be resolved with respect to its neighborhood in S, then we can just delete that component and reduce the budget by $fvs(C)$.

Definition 11 Let (G, S, ℓ) be an instance of FVS-Pseudo-Forest. Let $X \subseteq S$ be such that t connected components in F cannot be resolved with respect to X, then we say that X *is saturated by* t *unresolvable components* in F.

Let (G, S, ℓ) be an instance of FVS-$\mathcal{F}$-Deletion parameterized by $|S|$ where S is a set of vertices whose deletion results in a hereditary class of graph $\mathcal{F}$. Moreover, assume that FVS is polynomial time solvable on $\mathcal{F}$. Also, let us assume the following assumption holds true.

Assumption 1 *Let C be any component of $G \setminus S$. Then if C cannot be resolved with respect to $X \subseteq N_G(C) \cap S$, then C cannot be resolved with respect to $X' \subseteq X$ such that $|X'| \leq t_{\mathcal{F}}$ for some constant $t_{\mathcal{F}}$.*

For both FVS-Pseudo-Forest and FVS-Mock-d-Forest, we will prove that the above assumption holds true with different values of $t_{\mathcal{F}}$. If a very large number of components cannot be resolved by A for some $A \subseteq S$, then that A must intersect any optimal solution of G. Using this intuition, we have the following lemma.

Lemma 11 *Let (G, S, ℓ) be an instance of* FVS-$\mathcal{F}$-Deletion. *Let $A \subseteq S, |A| \leq t_{\mathcal{F}}$ as used in Assumption 1 and A is saturated by $|S| + \binom{t_{\mathcal{F}}}{2} + 1$ components of F. Then any minimum feedback vertex set of G must intersect A.*

Proof The intuition for the proof of this lemma is the following. Suppose D is a minimum feedback vertex set of G. Clearly $|D| \leq |S| + fvs(G \setminus S)$. Let $A \in \binom{S}{t_{\mathcal{F}}}$ be saturated by $|S| + \binom{t_{\mathcal{F}}}{2} + 1$ components of $G \setminus S$. For the sake of contradiction, suppose $D \cap A = \emptyset$. Let $p = \binom{t_{\mathcal{F}}}{2}$. There are $|S| + p + 1$ components that cannot be resolved with respect to A. As $|D| \leq |S| + fvs(G \setminus S)$, there are at most $|S|$ components $C_1, \ldots, C_{|S|}$ in $G \setminus S$ having more than $fvs(C_i)$ many vertices from C_i in D. So, there are at least $p + 1$ components in $G \setminus S$ that cannot be resolved with respect to A and have exactly $fvs(C)$ vertices from C into D. Let $\mathcal{Z}$ be the set of such components. Let $C \in \mathcal{Z}$ be one such component. As C cannot be resolved with respect to A, for all $B \subseteq C$ such that B is a minimum feedback vertex set of C, either $G[(C \setminus B) \cup A]$ has a cycle or there exists a component C' of $C \setminus B$ such that $|N_G(C') \cap A| \geq 2$ or $|N_G(A) \cap C'| \geq 2$. Let $B = D \cap C$. The case "$G[(C \setminus B) \cup A]$ has a cycle" cannot happen as it contradicts that D is a feedback vertex set. It also cannot be the case that for some vertex $x \in A$ and some component C' in $C \setminus B$, $|N_G(x) \cap C'| \geq 2$, then $G[C' \cup \{x\}]$ will form a cycle and will contradict the fact that D is a feedback vertex set. So, as there exists a component C' of $C \setminus D$ such that $|N_G(C') \cap A| \geq 2$ or $|N_G(A) \cap C'| \geq 2$. In either case, there are two vertices in A between which there is a path in $G \setminus D$ and that path avoids D and uses vertices of $C \setminus D$ only. Now, as $C \in \mathcal{Z}$ was arbitrary, this holds true for other components in $\mathcal{Z}$ as well. There are at most $\binom{t_{\mathcal{F}}}{2}$ such pairs in A. But there are $\binom{t_{\mathcal{F}}}{2} + 1$ components in $\mathcal{Z}$ for each of which there is a path between some pair of vertices from A. Then, by pigeon hole principle, there exists a pair $a, b \in A$ such that there are two components $P_1, P_2 \in \mathcal{Z}$ such that there is one path from a to b using vertices of $P_1 \setminus D$ and the other path from a to b uses vertices of $P_2 \setminus D$. This creates a cycle containing a and b in $G \setminus D$. Now, this is a contradiction to the fact that D is a feedback vertex set. So, $D \cap A \neq \emptyset$. □

Now, we have the following reduction rule. We will use the above lemma and the following reduction rule with different values of $t_{\mathcal{F}}$ for both FVS-Pseudo-Forest and FVS-Mock-d-Forest.

Reduction Rule 11 *Let C be a connected component of $G \setminus S$ and let Assumption 1 holds true. If for each $A \in \binom{N_G(C) \cap S}{\leq t_{\mathcal{F}}}$, either component C can be resolved with respect to A or A is saturated by at least $|S| + \binom{t_{\mathcal{F}}}{2} + 2$ components of $G \setminus S$, then delete C and reduce ℓ by $fvs(C)$.*

Lemma 12 *Reduction Rule 11 is safe.*

Proof ($\Rightarrow$) Let (G, S, ℓ) be an Yes-Instance, then any feedback vertex set D of size at most ℓ must have at least $fvs(C)$ vertices from C. So, $G \setminus C$ has a feedback vertex set of size $\ell - fvs(C)$.

($\Leftarrow$) Let D' be a minimum feedback vertex set of $G \setminus C$. Consider $S' = S \setminus D'$. We claim that C can be resolved with respect to S'. Suppose not. Then, by Assumption 1 (as we assume that it holds true for this problem), there exists $S'' \in \binom{S'}{\leq t_{\mathcal{F}}}$ such that C cannot be resolved with respect to S''. But, by the precondition of the reduction rule, S'' is saturated by $|S| + \binom{t_{\mathcal{F}}}{2} + 2$ components of $G \setminus S$ in G. So, even in $G \setminus C$ we have that S'' is saturated by $|S| + \binom{t_{\mathcal{F}}}{2} + 1$ components in $G \setminus (S \cup C)$. So, $D' \cap S'' \neq \emptyset$ by Lemma 11. This contradicts our choice of D' as $D' \cap S'' = \emptyset$ by our choice. So, C can be resolved with respect to S' and we can add exactly $fvs(C)$ vertices into D to get a feedback vertex set of size at most $|D'| + fvs(C)$. In particular, we have proved that $OPT(G) = OPT(G \setminus C) + fvs(C)$. □

4.3 Polynomial Kernel when parameterized by Deletion Distance to Pseudo-Forest

FVS-Pseudo-Forest	**Parameter:** k
Input: An undirected graph G, $S \subseteq V(G)$ of size at most k such that $G[V(G) \setminus S]$ is a graph in which every component has at most one cycle, and an integer ℓ.	
Question: Does G have a feedback vertex set of size at most ℓ?	

Throughout the section for input (G, S, ℓ), we use F to denote $G[V(G) \setminus S]$. An $\mathcal{O}(k^{10})$ vertex kernel is provided by Jansen et al. [26]. We provide here an improved kernel. We assume that the *pseudo-forest deletion set* is given with the input. But, this assumption can be omitted as there are 2-factor approximation algorithms (see [17,20]) that provide a pseudo-forest deletion set of size at most $2k$ and this does not asymptotically increase the size bound we get. We first apply the Reduction Rules 1, 2, 3, 4. When Reduction Rules 1, 2, 3, 4 are not applicable, then every vertex of the graph has degree at least three and there are at most two edges between every pair of vertices. In particular, every vertex in $V(F)$ has at least one neighbor in S. We partition the vertices of F into H_1, H_2, H_3 as below. Let H_T be the set of vertices of F that do not participate in any cycle in F. And H_C be the set of vertices of F that participates in some cycle in F. We also partition the components of F into F_1, F_2, F_3. Formal definitions are given as follows. In particular, we use the following notations.

- $H_1 = \{u \in V(F) | deg_F(u) \leq 1\}$.
- $H_2 = \{u \in V(F) | deg_F(u) = 2\}$.
- $H_3 = \{u \in V(F) | deg_F(u) \geq 3\}$.
- $H_T = \{u \in V(F) | u$ does not participate in any cycle in $F\}$.
- $F_1 = \{$connected components of F that is a tree$\}$.
- $F_2 = \{$connected components of F that contains a vertex from H_1 and also contains a cycle$\}$. Let $c_2 = |F_2|$.
- $F_3 = \{$connected components of F that are induced cycles$\}$. Let $c_3 = |F_3|$.
- Let $\mathcal{P}$ be the collection of *maximal degree-2-paths* in $F_1 \cup F_2$. Let M be a maximum matching in $\mathcal{P} \cup F_3$.
- Let $\hat{c} = c_2 + c_3$.

We will use the above set of notations in the rest of the sections for our kernelization algorithm. Also note that $fvs(F) = \hat{c}$ because $\hat{c}$ is the number of cycles in F. We have the following observation which is easy to see.

Observation 4 $F = F_1 \uplus F_2 \uplus F_3$.

Proof It is clear that any connected component of $F_1 \uplus F_2 \uplus F_3$ is in F. So, $F_1 \uplus F_2 \uplus F_3 \subseteq F$. Let C be a connected component in F. It C is a tree, then $C \in F_1$. Otherwise, C is a cycle. In that case C has exactly one cycle. Then either C has a vertex with degree one in which case $C \in F_2$. Otherwise C is an induced cycle in which case $C \in F_3$. So, $F \subseteq F_1 \uplus F_2 \uplus F_3$. This completes the proof.

The following observation is also easy to see as a vertex of degree at most one does not participate in any cycle.

Observation 5 *Let H_1, H_T be the set of vertices as defined in the previous box. Then, $H_1 \subseteq H_T$.*

4.3.1 General Reduction Rules for FVS-Pseudo-Forest

Our first step is to devise some reduction rules to bound the number of vertices in H_1. As soon as the number of vertices in H_1 is bounded, the number of vertices in H_3 is also bounded by a pseudo-forest property (See Observations 6, 7 for more details). Now, to bound the number of vertices in H_2, we need to bound the number of edges in M, the number of maximal degree-2-paths in $\mathcal{P}$ and c_3. By the pseudo-forest property, the number of maximal degree-2-paths in $\mathcal{P}$ also becomes bounded once $|H_1|$ and $|H_3|$ are bounded. In order to define our reduction rules, we crucially use the fact that the minimum degree of G is at least 3. In particular, for every vertex $v \in H_1$, either there exists $x \in S$ such that (x, v) is a double-edge or there exists $x, y \in N_G(v) \cap S$. The reduction rules described in this subsection help to bound H_1 and also $M \cap (E(F_1) \cup E(F_2))$. Our first two reduction rules are similar to Reduction Rules 8 and 9. But, here we have to apply this rule for specific subgraphs of F, so we state them separately.

Reduction Rule 12 (Flower Rule) *Let $x \in S$. Then $G' \leftarrow G \setminus \{x\}, \ell' \leftarrow \ell - 1$ if any of the following happens.*

1. *$G[\{x\} \cup H_T]$ has $(|S| + 1)$-flower with core x.*
2. *$G[\{x\} \cup F]$ has $(|S| + \hat{c} + 1)$-flower with core x.*

Reduction Rule 13 (Vertex Disjoint Path Rule) *Let $x, y \in S$ such that (x, y) is not a double-edge. Then, make (x, y) into a double-edge if one of the following happens.*

1. *There are at least $|S|+2$ internally vertex disjoint paths from x to y in $G[H_T \cup \{x,y\}]$.*
2. *There are at least $|S|+\hat{c}+2$ internally vertex disjoint paths from x to y in $G[F \cup \{x,y\}]$.*

Note that vertices of H_T are not part of any cycle in F. So, we put $c=0$ in Reduction Rules 8 and 9 to get the first part of both the above rules. And subsequently the vertices of F intersects all the $\hat{c}$ cycles from F. So, we put $c=\hat{c}$ in Reduction Rules 8 and 9 to get the second part of both the above rules. So, safeness and polynomial running time of the above two reduction rules follow from the proof of Lemma 8 and 9. Even though Reduction Rule 13 does not reduce the size of the graph, it helps to capture some constraints and also helps to apply some other reduction rules (for example Reduction Rule 10). We apply Reduction Rule 10 after applying Reduction Rules 12 and 13.

4.3.2 Bounding $|H_1 \cup H_3|$

Now, we proceed to bound the number of vertices in F that have degree at most one and at least three. We know that F is a pseudo-forest. We need to use some structural properties of a pseudo-forest and also that the Reduction Rules 1, 2, 3, 4 12, 13, 10 are not applicabe. We start with showing that the following observation, which is a well-known property of forests, holds true also about pseudo-forests.

Observation 6 *Let $G=(V,E)$ be a pseudo-forest and let $V_1=\{v \in V(G) | deg_G(v) \leq 1\}, V_2=\{v \in V(G) | deg_G(v)=2\}$ and $V_3=\{v \in V(G) | deg_G(v) \geq 3\}$. Then, $|V_3| \leq |V_1|$.*

Proof Since G is a pseudo-forest, $|E| \leq |V|$. Let us first assume that G has no isolated vertices. Then by standard graph theoretic property, we have the following.

$$2|V| \geq 2|E| = \sum_{v \in V} deg_G(v) \geq |V_1| + 2|V_2| + 3|V_3|$$

By definition, we know that $|V| = |V_1|+|V_2|+|V_3|$. So, we have the following.

$$2(|V_1|+|V_2|+|V_3|) \geq |V_1| + 2|V_2| + 3|V_3|$$

$$|V_3| \leq |V_1|$$

Now, when it has isolated vertices, then they contribute to V_1. So, we have that $|V_3| \leq |V_1|$. □

Using Observation 6, we have the following lemma.

Lemma 13 *When Reduction Rules 1, 2, 3, 4, 12, 13 and 10 are not applicable, $|H_1 \cup H_3| \leq 2k^2 + 2(k+1)\binom{k}{2}$.*

Proof By definition of F_1, F_2, H_1, H_3, we know that $H_1 \subseteq V(F_1 \cup F_2)$ and $H_3 \subseteq V(F_1 \cup F_2)$ because F_3 and F_4 contains only induced cycles. So, we have that $H_1 \cup H_3 \subseteq V(F_1 \cup F_2)$. Also $H_1 \subseteq H_T$ by Observation 5. Since Reduction Rules 1, 2, 3 are not applicable, for every vertex in H_1 either there are least two neighbors in S or there is at least one neighbor in S which is connected by a double-edge. As Reduction Rule 10 is not applicable, for every vertex $v \in H_1$, we associate $z \in S$ when (x,z) is a double-edge. Otherwise we associate $(x,y) \in \binom{S}{2}$ for v, when $x,y \in N_G(v) \cap S$ and (x,y) is not a double-edge. For any $x \in S$, we define $H_{1,x} = \{u \in H_1 | (u,x)$ is a double-edge$\}$. If $|H_{1,x}| \geq |S|+1$, then in $G[H_T \cup \{x\}]$, there are at least $|S|+1$ cycles that pairwise intersect in x only. So, Reduction Rule 12 is applicable. This is a contradiction. So, for every $z \in S$, there are at most $|S|$ vertices in H_1 that are connected by a double-edge. Consider any $x,y \in S$ such that (x,y) is not a double-edge. If $|N_G(x) \cap N_G(y) \cap H_1| \geq |S|+2$, then there are at least $|S|+2$ internally vertex disjoint paths from x to y in $G[\{x,y\} \cup H_T]$. Then Reduction Rule 13 is applicable which is a contradiction. So, for every $(x,y) \in \binom{S}{2}$ where (x,y) is not a double-edge, $N_G(x) \cap N_G(y)$ contains at most $|S|+1$ vertices of H_1. Then $|H_1| \leq k^2 + (k+1)\binom{k}{2}$. By Observation 6, we know that $|H_3| \leq |H_1| \leq k^2 + (k+1)\binom{k}{2}$. So, $|H_1 \cup H_3| \leq 2k^2 + 2(k+1)\binom{k}{2}$. □

4.3.3 Bounding the number of components in F_3

Now, what remains is to bound $|H_2|$. Towards that, we first upper bound the number of induced cycles, i.e. c_3 which is the number of components in F_3. By definition, for any component of F_3, no vertex from F_3 has exactly one neighbor in F. In particular the graph induced on the set of components of F_3 is a two regular graph.

We recall the idea of Definition 10. The idea is that if a component C can be resolved with respect to its neighborhood in S, then we can just delete that component and reduce the budget by 1. Every connected component of F_3 is an induced cycle. When Reduction Rules 1 2, 3, 4 are not applicable, we show that the components in F_3 have the following properties. The following lemma is provided as Lemma 4 in [26] which we refer to for a proof. In particular, the following lemma provides a proof of the Assumption 1 for FVS-Pseudo-Forest with value of $t_{\mathcal{F}} \leq 4$ for components of F_3.

Lemma 14 *(Lemma 4 [26])*[2] *Let C be a connected component in F_3 and that Reduction Rules 2, 3 be not applicable. Then, if there exists $X \subseteq N_G(C) \cap S$ such that C cannot be resolved with respect to X then there exists $X' \subseteq X, |X'| \leq 4$ such that C cannot be resolved with respect to X'.*

The intuition behind the proof of Lemma 14 is that if for some $X \subseteq S, |X| \leq 4$, then there are a large number of components that cannot be resolved with respect to X. Hence any minimum feedback vertex set must intersect X. Now, we have the following lemma (a variant of which which is also there as Lemma 5 in [26]) which we get from Lemma 11 by plugging in $t_{\mathcal{F}} = 4$ for the components of F_3.

Lemma 15 *(Lemma 5 [26]) Let (G, S, ℓ) be an instance of* FVS-Pseudo-Forest *and $A \subseteq S, |A| \leq 4$ and A is saturated by $|S| + 7$ components in F_3, then any minimum feedback vertex set of G must intersect A.*

Now, we have the following Reduction Rule that we get from Reduction Rule 11 by plugging in $t_{\mathcal{F}} = 4$ for components of F_3. Note that the following reduction rule is available as Reduction Rule 1 in [26].

Reduction Rule 14 *Let C be a connected component of F_3. If for each $A \subseteq \binom{S \cap N_G(C)}{\leq 4}$, component C can be resolved with respect to A or A is saturated by at least $|S| + 8$ unresolvable components in F_3, then delete C and reduce ℓ by 1.*

Safeness of the above reduction rule is clear from the proof of Lemma 12. After exhaustive applications of the Reduction Rules 1, 2, 3, 4 and 14, we get the following lemma. The proof uses arguments similar to that of Lemma 7 in [26].

Lemma 16 *When Reduction Rule 1, 2, 4, 3, 14 are not applicable, then the number of connected components in F_3 is at most $(k + 7) \sum_{i=1}^{4} \binom{k}{i}$.*

Proof Assume that the condition holds. Since Reduction Rule 14 is not applicable, for each component C in F_3, there exists a set $A \in \binom{N_G(C) \cap S}{\leq 4} \subseteq \binom{|S|}{4}$ such that C can be resolved with respect to A and A is saturated by at most $|S| + 7$ components. Then, for each component C in F_3, we choose one such set A and associate A to C. Clearly there can be at most $|S| + 7$ components of F that cannot be resolved with respect to A, otherwise some set A would be saturated by $|S| + 8$ components. Hence the number of components in F_3 is at most $(|S| + 7) \sum_{i=1}^{4} \binom{|S|}{i} \leq (k + 7) \sum_{i=1}^{4} \binom{k}{i}$. □

4.3.4 Bounding $|H_2|$ and Putting Things together

Now, we proceed to get an upper bound on the number of vertices in H_2. We need a few more structural properties of pseudo-forests. As any component in F_2 has at least one vertex who has exactly one neighbor in F, the number of components in F_2, i.e. c_2 is at most $|H_1|$. Recall that the number of components in F_2 is c_2. So, we have the following lemma.

Lemma 17 $c_2 \leq |H_1|$.

Recall that in order to bound $|H_2|$, we also need an upper bound on the number of degree-2-paths in $\mathcal{P}$. The following structural property of a pseudo-forest is useful.

[2] A preliminary version [31] of this paper had a stronger lemma (Lemma 25 of [31]) which gave an upper bound of 3 for $|X'|$, and this resulted in an overall $\mathcal{O}(k^6)$ vertex kernel for this problem. However, the lemma had an error.

Observation 7 *Let $G = (V, E)$ be a pseudo-forest where every component has at least one vertex of degree 1. Let $V_1 = \{v \in V(G) | deg_G(v) \leq 1\}$ and $V_3 = \{v \in V(G) | deg_G(v) \geq 3\}$ and $\mathcal{P}$ be the set of maximal degree-2-paths in G. Then, $|\mathcal{P}| \leq |V_3| + |V_1|$.*

Proof Let $|V| = n$. Convert the given pseudo-forest $G = (V, E)$ into $G' = (V', E')$, where $V' = V_1 \cup V_3$ and E' be the set of edges that are constructed after short-circuiting (i.e. after applying Reduction Rule 3) all degree 2 vertices. Then, $|E'| \leq |V_3| + |V_1|$. Therefore, $|\mathcal{P}| \leq |E'| \leq |V_3| + |V_1|$. This completes the proof. □

Lemma 18 *If Reduction Rule 1, 2, 3, 4, 12, 13, 10, 14 are not applicable, then the number of vertices in H_2 is $\mathcal{O}(k^7)$.*

Proof Recall that by Lemma 17, we have that c_2 is $\mathcal{O}(k^3)$. Also by Lemma 16, we have that c_3 is $\mathcal{O}(k^5)$. So, $\hat{c} = c_2 + c_3$ which is $\mathcal{O}(k^3) + \mathcal{O}(k^5)$. Recall that M is a maximum matching in $\mathcal{P} \cup F_3$. As Reduction Rule 3 is not applicable, every vertex in H_2 has at least one neighbor in S. By Lemma 16, the number of 2-cycles in F is $\mathcal{O}(k^5)$. As Reduction Rule 10 is not applicable, for every $(u, v) \in M$ such that (u, v) is not a 2-cycle, we associate $z \in N_G(u) \cap N_G(v) \cap S$ when $N_G(u) \cap N_G(v) \cap S \neq \emptyset$ or (u, z) is a double-edge. Otherwise, we associate $x, y \in (N_G(u) \cup N_G(v)) \cap S$ such that (x, y) is not a double-edge. For any $z \in S$, define $Matched(z) = \{(u, v) \in M | (u, z)$ is a double-edge or $u, v \in N_G(z)\}$. If for some $z \in S$, $|Matched(z)| \geq |S| + \hat{c} + 1$, then there are at least $|S| + \hat{c} + 1$ cycles in $G[(\{x\} \cup F)]$ pairwise intersecting at z only. Then Reduction Rule 12 becomes applicable. This is a contradiction. So, $|Matched(z)| \leq |S| + \hat{c} + 1$. Similarly if for some $x, y \in \binom{S}{2}$ such that (x, y) is not a double-edge and either $N_G(x) \cup N_G(y)$ contains both end points of at least $|S| + \hat{c} + 2$ edges of M or $N_G(x) \cap N_G(y)$ contains one end point of at least $|S| + \hat{c} + 2$ edges of M, then there are at least $(|S| + \hat{c} + 2)$ internally vertex disjoint paths from x to y in $G[\{x\} \cup F]$. Then, Reduction Rule 13 is applicable. So, $|M| \leq |S|(|S| + \hat{c}) + \binom{|S|}{2} 2(|S| + \hat{c} + 1) \leq 2k^2 + 2k(k+7) \sum_{i=1}^{4} \binom{k}{i} + 2\binom{k}{2}(k + (k+7) \sum_{i=1}^{4} \binom{k}{i} + 1)$ which is $\mathcal{O}(k^7)$. A *maximal degree-2-path* can also have only one vertex which is not matched by M. Recall that $\mathcal{P}$ is the collection of all *maximal degree-2-paths* in $F_1 \cup F_2$. Using Observation 7, we get that $|\mathcal{P}| \leq |H_1| + |H_3|$ which is $\mathcal{O}(k^3)$. So, the number of vertices in H_2 that are not matched by M is at most $|\mathcal{P}| + |M|$ which is $\mathcal{O}(k^7)$. So, $|H_2|$ is $\mathcal{O}(k^7)$. □

The following is the main theorem of this section and this is an easy consequence of Observation 4, Lemma 13 and Lemma 18.

Theorem 10 FVS-PSEUDO-FOREST *has a kernel with $\mathcal{O}(k^7)$ vertices.*

4.4 Kernelization algorithm Parameterized by Deletion distance to bounded Mock Forest

Now we consider the FEEDBACK VERTEX SET problem parameterized by the size of a deletion set whose deletion results in a *mock-d-forest*. Recall that a graph is called mock-d-forest when every vertex is contained in at most one cycle and every connected component has at most d cycles. Formal definition of the problem is given below.

FVS-MOCK-d-FOREST FOR $d \geq 2$ **Parameter:** k
Input: An undirected graph G, $S \subseteq V(G)$ of size at most k such that $G[V(G) \setminus S]$ is a graph of which every vertex participates in at most most one cycle, every component has at most d cycles for some constant d and an integer ℓ.
Question: Does G have a feedback vertex set of size at most ℓ?

When d is not bounded, then there is no polynomial kernel unless NP ⊆ coNP/poly [26]. In this section, we first provide a polynomial kernel for this problem when d is a constant and $d \geq 2$. After that we provide a lower bound for this problem.

4.4.1 Polynomial Kernel for FVS-MOCK-d-FOREST *for $d \geq 2$*

Our kernelization algorithm follows along the line of the kernel for FVS-PSEUDO-FOREST in Section 4.3. Here, we need to use some special properties of mock-d-forest. We use $F = G \setminus S$ throughout the section. Let $\mathcal{P}_F$ be the collection of maximal acyclic degree-2-paths in F. Let $\mathcal{Q}$ is the set of vertices that are in $\mathcal{P}_F$ or in the induced cycles of F. Let M_F be a maximum matching in $\mathcal{Q}$. Let $\hat{c}$ be the total number of cycles in F. We partition $V(F)$ into three parts F_1, F_2, F_3 as follows. In addition, we denote the set of vertices of F that do not participate in any cycle by F_T. In particular, we use the following notations.

- $F_1 = \{u \in V(F) | deg_F(u) \leq 1\}$.
- $F_2 = \{u \in V(F) | deg_F(u) = 2\}$.
- $F_3 = \{u \in V(F) | deg_F(u) \geq 3\}$.
- $F_T = \{u \in V(F) | u$ does not participate in any cycle of $F\}$.

The following two observations are easy to see.

Observation 8 $F = F_1 \uplus F_2 \uplus F_3$.

Observation 9 $F_1 \subseteq F_T$.

Our first step is to bound the number of vertices in F_1. An upper bound on F_1 along with some properties of a mock-d-forest, we get an upper bound on the number of vertices of F_3. Then, we have to bound the number of edges in M_F and the number of maximal acyclic degree-2-paths in $\mathcal{P}_F$. Now, we are ready to state the Reduction Rules. Our Reduction Rules in this section are generalizations of the Reduction Rules in Section 4.3. In particular, the first two reduction rules (Reduction Rules 15 and 16) are quite similar to Reduction Rules 12 and 13. As, there are some notational differences, we state them separetely.

Reduction Rule 15 *(Flower Rule) Let $x \in S$. Then $G' \leftarrow G \setminus \{x\}, \ell' \leftarrow \ell - 1$ if one of the following conditions is satisfied.*

1. *$G[\{x\} \cup F_T]$ has $(|S| + 1)$-flower with core x.*
2. *$G[\{x\} \cup F]$ has $(|S| + \hat{c} + 1)$-flower with core x.*

Reduction Rule 16 *Let $(x, y) \in \binom{S}{2}$. Then make (x, y) into a double-edge if one of the following conditions is satisfied.*

1. *There are at least $|S| + 2$ internally vertex disjoint paths from x to y in the graph $G[\{x, y\} \cup F_T]$.*
2. *There are at least $|S| + \hat{c} + 2$ internally vertex disjoint paths from x to y in the graph $G[\{x, y\} \cup F]$.*

Note that the first part of both the Reduction Rules 15, 16 are the same as the first condition of the Reduction Rules 12 and 13 respectively. Vertices of F_T do not intersect any cycle in F. On the other hand, F intersects all cycles in F. So, safeness and polynomial running time of the above two reduction rules follow from the proof of Lemma 8 and 9. We apply Reduction Rule 10 after applying Reduction Rules 15, 16. But, the second part, we applied it in a slightly different subgraph for FVS-Pseudo-Forest.

We apply Reduction Rules 1, 2, 3, 4, 15, 16, 10 in this order (Recall that we did similar in Section 4.3).

Lemma 19 *When Reduction Rules 1 2, 3, 4, 15, 16 are not applicable, $|F_1|$ is $\mathcal{O}(k^3)$.*

Proof Recall that F_1 is the set of vertices with degree at most 1 in F. So, the proof is exactly the same as that of Lemma 13. ◻

To bound the number of vertices in F_2 and F_3, we bound the number of components in F. We need to do a little more work for that. The following lemma is a generalization of Lemma 14 and is useful to bound the number of components in F. It also proves Assumption 1 with values $t_{\mathcal{F}} = 3d$.

Lemma 20 *Let C be a connected component of F having exactly d cycles ($d \geq 2$) and let $X \subseteq N_G(C) \cap S$ such that C cannot be resolved with respect to X. Then, there exists $X' \subseteq X, |X'| \leq 3d$ such that C cannot be resolved with respect to X'.*

Proof This proof uses the idea of Lemma 4 in [26]. Let $C_1, \ldots, C_d$ be the cycles in C. Let $\mathcal{A} = \bigcup_{i=1}^{d} C_i$. So $\mathcal{A}$ is the set of vertices of C that participates in some cycle of C. Consider any vertex $x \in X$. And let u be a vertex of $\mathcal{A}$. We call u as an *attachment point* for x if the subgraph C contains a path possibly of length zero from u to a vertex of $N_G(x) \cap C$ that intersects $\mathcal{A}$ only at u. A vertex $u \in \mathcal{A}$ is called an *attachment point* if it is an attachment point for some $y \in X$. We use a case distinction to prove the existence of a set X' with the desired properties.

1. Suppose that there exists an $i \in [d]$ such that C_i has 3 distinct attachment points $u_1, u_2, u_3 \in C_i$. Choose $x_1, x_2, x_3 \in X$ such that u_i is an attachment point for x_i for $i \in [3]$. Then, we claim that C cannot be resolved with respect to $X' = \{x_1, x_2, x_3\}$. By Definition 20, we know that for any $\{v_1, \ldots, v_d\}$ such that $C \setminus \{v_1, \ldots, v_d\}$ is acyclic, there exists a component C' of $C \setminus \{v_1, \ldots, v_d\}$ such that either $|N_G(C') \cap X| \geq 2$ or $|N_G(X) \cap C'| \geq 2$. Now, at most one from $\{u_1, u_2, u_3\}$ can intersect $\{v_1, \ldots, v_d\}$. Therefore, there are at least 2 attachment points from X' that was not chosen to be deleted. Deleting one from $\{u_1, u_2, u_3\}$ still keeps the remaining two vertices of $\{u_1, u_2, u_3\}$ into the same component. As each such attachment point has a path to a vertex in X' that avoids the vertices of $\mathcal{A}$, removing any d vertices to make C acyclic still leaves a component which is adjacent to at least 2 vertices of X' and hence C cannot be resolved with respect to X'. Hence, $|X'| \leq 3$ in this case.
2. Suppose that there exists a component C' of $C \setminus V(\mathcal{A})$ such that either $|N_G(C') \cap X| \geq 2$ or $|N_G(X) \cap C'| \geq 2$. If $|N_G(C') \cap X| \geq 2$, then pick any two vertices $X' = \{x_1, x_2\}$ from $N_G(C') \cap X$ and it is clear that C cannot be resolved with respect to X'. Similarly if $|N_G(X) \cap C'| \geq 2$, then either there are two distinct points $x, y \in X$ such that both x and y have distinct neighbors to C' in which case $X' = \{x, y\}$. Otherwise there exists a $x \in X$ such that $N_G(x)$ contains two neighbors in C'. In such a case $X' = \{x\}$.
3. If there exists at most one attachment point in every cycle, let $u_1, \ldots, u_d$ be the attachment points in $C_1, \ldots, C_d$ respectively. Then, we claim that C can be resolved with respect to X'. We choose $\{u_1, \ldots, u_d\}$ to delete from C. As none of the previous cases hold, for any component C' of $C \setminus V(\mathcal{A})$, $|N_G(C') \cap X| \leq 1$ and $|N_G(X) \cap C'| \leq 1$. Now suppose that there exists C'' of $C \setminus \{u_1, \ldots, u_d\}$ such that $|N_G(C'') \cap X| \geq 2$ or $|N_G(X) \cap C''| \geq 2$. Suppose $|N_G(C'') \cap X| \geq 2$. Consider $V(C'') \cap V(\mathcal{A})$. By the choice of deleted vertices, there is no attachment point in $V(C'') \cap V(\mathcal{A})$. Therefore, $V(C'') \cap V(\mathcal{A})$ has no neighbor in X. If $V(C'') \setminus V(\mathcal{A})$ has a neighbor in X, then $V(C'') \cap V(\mathcal{A})$ would be empty as otherwise some cycle will have one more attachment point which is a contradiction. When some cycle has no attachment point, then pick an arbitrary vertex from those cycles (that has no attachment point) and the proof follows using similar arguments.
4. If none of the above cases apply, then every cycle has at most two attachment points and there is a cycle which has exactly two attachment points. Now some of the attachment points in $\mathcal{A}$ may be attachment points for at least two vertices in X. Let the number of such attachment points be q.
   - If $q \geq d+1$, let $v_1, v_2, \ldots, v_{d+1}$ be those vertices of $\mathcal{A}$ and v_i is an attachment point for $x_{i,1}, x_{i,2} \in X$. Now, we construct $X' = \bigcup_{i=1}^{d+1} \{x_{i,1}, x_{i,2}\}$ and claim that C cannot be resolved with respect to X'. Certainly whichever d vertices are deleted from C, there is a component consisting of at least one vertex w from $\mathcal{A}$ and there are two vertices from X' from whom there is a path to w avoiding $\mathcal{A}$. Therefore, C cannot be resolved with respect to X'. Therefore, $|X'| = 2d+2$ in this case.
   - If $q \leq d$, then suppose $v_1, v_2, \ldots, v_q$ be those vertices. Now, if for some $i, j \in [q]$, v_i, v_j are part of the same cycle, then C cannot be resolved with respect to $\{x_{i,1}, x_{i,2}, x_{j,1}, x_{j,2}\}$ since at most one of v_i and v_j can be in D. Therefore, all $v_1, \ldots, v_q$ appear in different cycles. Then consider any $D \subseteq C, |D| = d$ such that $C \setminus D$ is acyclic. If $v_i \notin D$ for some $i \in [q]$, then certainly D cannot resolve C as there is a path to v_i from $x_{i,1}, x_{i,2}$ whose internal vertices are in $C \setminus D$ and intersects $\mathcal{A}$ only at v_i. We pick $X_i = \{x_{i,1}, x_{i,2}\}$ for all such $i \in [q]$. Now, there are at most $2d - q$ other attachment points $u_1, \ldots, u_{2d-q}$ since every cycle can have at most 2 attachment points. Now, all those $2d - q$ other attachment points are attachment point for exactly one vertex of X. Let $w_1, \ldots, w_{2d-q}$ be those attachment points and they are attachment points for vertices $a_1, \ldots, a_{2d-q} \in X$ Let $X' = (\bigcup_{i=1}^{q} \{x_{i,1}, x_{i,2}\}) \cup \{a_1, \ldots, a_{2d-q}\}$. We claim that C cannot be resolved with respect to X'. Suppose not. Then there exists $D \subseteq C$ where $|D| = d$ and for every component C' of $C \setminus D$, $|N_G(C') \cap X'| \leq 1$ and $|N_G(X') \cap C'| \leq 1$ and there is no cycle in the graph induced on the vertices of $(C \setminus D) \cup X'$. By assumption, D must contain $v_1, \ldots, v_q$. Now, D can contain at most $d - q$ other vertices from the other $2d - q$ attachment points. So, there are still d attachment points that are in $C \setminus D$. Now, as C can be resolved with respect to X' but not with respect to X, for some $i \in [q]$, v_i has another neighbor in $a \in X \setminus X'$ which is again an attachment point for some u_j where $j \in [2d - q]$. But then u_j is also an attachment point for two vertices in X. Then, there are $q + 1$ attachment points in C that are attachment point for two vertices in X which contradicts our assumption that the

number of such vertices can be at most q. So, C cannot be resolved with respect to X'. Now we have that, $|X'| \leq \sum_{i=1}^{q} |X_i| + 2d - q = 2q + 2d - q = 2d + q$. Now, since $q \leq d$ $|X'| \leq 3d$.

As the case distinction is exhaustive, this concludes the proof. □

We again recall the Definition 11 in Section 4.2. If some set $A \subseteq S$ of at most $3d$ vertices is saturated by a large number of components in F, then any minimum feedback vertex set must intersect A.

The following lemma is obtained from Lemma 11 by plugging in $t_{\mathcal{F}} = 3d$.

Lemma 21 *Let (G, S, ℓ) be an instance of* FVS-Mock-d-Forest *and $A \subseteq S, |A| \leq 3d$ and A is saturated by $|S| + \binom{3d}{2} + 1$ components in F, then any minimum feedback vertex set of G must intersect A.*

Now, we have just one more reduction rule to get an upper bound on the number of components in F. And Lemma 22 is a consequence of inapplicability of Reduction Rule 17. We get the following reduction rule from Reduction Rule 11 by putting $t_{\mathcal{F}} = 3d$. Safeness is clear from the proof of safeness of Reduction Rule 11 (or equivalently Lemma 12).

Reduction Rule 17 *Let C be a connected component in F that contains some cycle. If for each $A \in \binom{N_G(C) \cap X}{\leq 3d}$, C can be resolved with respect to A or A is saturated by $|S| + \binom{3d}{2} + 2$ components, then remove C and reduce ℓ by the number of cycles in C.*

Lemma 22 *Let (G, S, ℓ) be an irreducible instance with respect to Reduction Rule 17, then the number of components in F is at most $\mathcal{O}(|S|^{3d+1})$.*

Proof Consider any component $C \in F$. Reduction Rule 17 is not applicable. Hence, there exists $A \subseteq S, |A| \leq 3d$ such that C cannot be resolved with respect to A. Also, for the same reason, A can be saturated by at most $|S| + \binom{3d}{2} + 1$ components. Therefore, the number of components is at most $(|S| + \binom{3d}{2})\binom{|S|}{3d} \leq 9d^2.|S|^{3d+1}$ which is $\mathcal{O}(d^2.|S|^{3d+1})$. □

We have bounded the number of components in F. We already have bounded $|F_1|$. We are left to bound $|F_3 \cup F_2|$. We need a graph theoretic properties of a mock-d-forest to get an upper bound on $|F_3|$. Recall that in Section 4.3, we used observations about pseudo-forest. Similarly, in this section, we use observations about mock-forest when there are at most d cycles in each component of a mock-forest.

Observation 10 *Let $G = (V, E)$ be a mock forest with c components where every component has at most d cycles. $V_1 = \{v \in V(G) | deg_G(v) \leq 1\}, V_2 = \{v \in V(G) | deg_G(v) = 2\}, V_3 = \{v \in V(G) | deg_G(v) \geq 3\}$. Then $|V_3| \leq |V_1| + 2cd - 2c$.*

Proof Let us first assume that G has no isolated vertex and c be the number of components in G. Removing at most cd edges makes G into a forest. Therefore $|E(G)| = m \leq n - c + cd$. We know that $2m = \sum_{v \in V(G)} deg_G(v)$. As G has no isolated vertices, we have that

$$2m \geq |V_1| + 2|V_2| + 3|V_3|$$

$$2n + 2cd - 2c \geq |V_1| + 2|V_2| + 3|V_3|$$

$$2|V_1| + 2|V_2| + 2|V_3| + 2cd - 2c \geq |V_1| + 2|V_2| + 3|V_3|$$

$$|V_1| + 2cd - 2c \geq |V_3|$$

When G has no isolated vertex, we have that $|V_3| \leq |V_1| + 2c(d-1)$. Now, when G has c' isolated vertices, they contribute an additional value of c' to $|V_1|$ and $2c$ and none to $|V_3|$. Hence, the claim follows. □

Using Lemma 19 and Observation 10, we have the following Lemma.

Lemma 23 $|F_3|$ *is* $\mathcal{O}(k^{3d+1})$.

Proof By Lemma 19, we know that $|F_1|$ is $\mathcal{O}(k^3)$. Now, by Observation 10, we know that $|F_3| \leq |F_1| + 2c(d-1)$. Now, $c(d-1) \leq \hat{c}$ which is $\mathcal{O}(k^{3d+1})$. So, $|F_3|$ is $\mathcal{O}(k^{3d+1})$. □

Now, what remains is to bound the number of vertices in F_2. For that, we need to bound M_F and also the number of maximal acyclic degree-2-paths in $\mathcal{P}_F$. Using structural properties of mock-d-forest, we have the following lemma that bounds the number of maximal acyclic degree-2-paths in F, i.e. $\mathcal{P}_F$.

Lemma 24 *$|\mathcal{P}_F|$ is $\mathcal{O}(k^{3d+1})$ where c' is the number of components in F that have at least two cycles.*

Proof Short-circuit every vertex of a *maximal acyclic degree-2-path* of F by applying Reduction Rule 3 and we construct F'. So, the resulting graph F' is still remains a mock-d-forest. So, F' has at most $|V(F')| + dc' \leq |F_1| + |F_3| + dc'$ edges. Now, every degree-2-path corresponds to some edge in F'. So, $|\mathcal{P}_F| \leq |F_1| + |F_3| + dc'$. As, c' is $\mathcal{O}(k^{3d+1})$ and $|F_3|$ is $\mathcal{O}(k^{3d+1})$, we have that $|\mathcal{P}_F|$ is $\mathcal{O}(k^{3d+1})$. □

Using the above observations and lemmas, we have the following lemma.

Lemma 25 *$|F_2|$ is $\mathcal{O}(|S|^{3d+3})$.*

Proof By Lemma 19 and 23, we have that $|F_1|$ is $\mathcal{O}(k^3)$ and $|F_3|$ is $\mathcal{O}(k^{3d+1})$ respectively. As Reduction Rule 10 is not applicable, for every edge $(u, v) \in M_F$, we associate either $x \in S$ when $x \in N_G(u) \cap N_G(v)$ or (x, u) is a double-edge. Otherwise, when $N_G(u) \cap N_G(v) \cap S = \emptyset$, then we associate $(x, y) \in S$ such that $x, y \in (N_G(u) \cup N_G(v)) \cap S$ such that (x, y) is not a double-edge. Let $Matched(x) = \{(u, v) \in M_F | (u, x)$ is a double-edge or $u, v \in N_G(x)\}$. If $|Matched(x)| \geq |S| + \hat{c} + 1$, then Reduction Rule 16 is applicable which is a contradiction. So, $|Matched(x)| \leq |S| + \hat{c}$. If $N_G(x) \cup N_G(y)$ contains both end points of at least $|S| + \hat{c} + 2$ edges of M_F, then there are at least $|S| + \hat{c} + 2$ vertex disjoint path from x to y in $G[F \cup \{x, y\}]$. So, Reduction Rule 16 is applicable which is also a contradiction. Similarly, when $N_G(x) \cap N_G(y)$ has one end-point of at least $|S| + \hat{c} + 2$ edges of M_F, then there are at least $|S| + \hat{c} + 2$ internally vertex disjoint paths from x to y in $G[\{x, y\} \cup F]$. So, Reduction Rule 16 is applicable which is a contradiction. Now using Lemma 22, we get $|M_F| \leq 2(|S| + \hat{c} + 1)(|S| + \binom{|S|}{2})$ which is $\mathcal{O}(k^{3d+3})$. Therefore, $|M_F|$ is $\mathcal{O}(|S|^{3d+3})$. Also, by using Lemma 19, 23, 24 we get that $|\mathcal{P}_F|$ is $\mathcal{O}(k^{3d+1})$. The number of vertices in $\mathcal{P}_F$ and in induced cycles of F that are not matched by M_F is at most $|M_F| + |\mathcal{P}_F|$ as a maximal acyclic degree-2-path can be a single vertex which is not matched by M_F. Hence the number of vertices in F_2 is at most $\mathcal{O}(|S|^{3d+3})$ and the theorem follows. □

Combining Lemma 19, 23, 25, we get the following theorem.

Theorem 11 FVS-Mock-d-Forest *has a kernel with $\mathcal{O}(k^{3d+3})$ vertices.*

4.4.2 Kernel Lower Bound for FVS-Mock-d-Forest *for $d \geq 2$*

We provide a *parameter preserving transformation* from $(d+2)$-CNF-SAT parameterized by the number of variables to Feedback Vertex Set parameterized by deletion distance to Mock-d-Forest where $d \geq 2$. A parameter preserving transformation from CNF-SAT to FVS-Mock-Forest when the length of every clause is a power of 2 is already known [26] (See Section 4.1). We modify the construction for a polynomial parameter transformation from $(d+2)$-CNF-SAT to FVS-Mock-d-Forest where d is not necessarily a power of 2.
Let the clause C_i have $d_i \leq d+2$ literals. We provide a clause gadget of height j_i where $2^{j_i - 1} < d_i \leq 2^{j_i}$. We create d^2 many copies for this gadget. In this gadget, the terminal vertices are the corresponding vertices of literals (see Figure 4). For clause C_q with its r'th copy, we name literals as $y_{q,r,1}, \ldots, y_{q,r,d_i}$. And we create a variable gadget for variable x_i as a cycle of 3 vertices. Let $\{t_i, f_i, e_i\}$ are those vertices. We define $S = \bigcup_{i=1}^{n} \{t_i, f_i, e_i\}$. Let $y_{q,r,j}$ be the j'th literal of clause C_q. Let the variable corresponding to that variable is x_i. Then, if the literal $y_{q,r,j}$ is $\bar{x_i}$, then we connect $y_{q,r,j}$ with f_i. Otherwise we connect $y_{q,r,j}$ with t_i. We do the same for every $r \in [d^2]$. We set $\ell = d^2 \sum_{i=1}^{m} (d_i - 2)$.

For any triangle T other than triangle at the topmost level, we define $\pi(T) = T'$ where T' is the triangle which is connected to T by an edge. If T is a triangle at the topmost level, then we define $\pi(T) = \perp$. For any terminal vertex v which is contained in triangle T and also not in the highest level, we define $\pi(v)$ as the unique vertex u that is adjacent to the top vertex of T. Note that $u = \pi(v)$ means that u is a bottom vertex of the triangle T' such that $\pi(T) = T'$. Similarly if v is a terminal vertex which belongs to a triangle at the topmost level, then $\pi(v) = \perp$. Now we show the following.

Lemma 26 *Let ϕ be a $(d+2)$-CNF formula. Let G_ϕ be the graph constructed from ϕ using the construction above. Then ϕ is satisfiable if and only if (G_ϕ, S, ℓ) is* Yes-Instance*. Thus there is a parameter preserving transformation from $(d+2)$-CNF-SAT to* FVS-Mock-d-Forest.

Proof The proof goes along the line of the proof of Lemma 7. But there are some differences. We mention the differences here. The following observation (also available in [26]) is a property of the clause gadget.

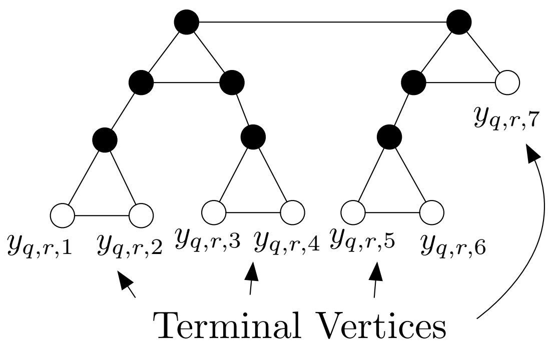

Fig. 4: Illustration of Clause Gadget Construction for 7 literals

Observation 11 *Let t_1, t_2 be two terminals (not necessarily distinct) of a clause gadget $\mathcal{G}_i$ for $i \geq 2$. Then either $\pi(t_1) = \pi(t_2)$ or any path between t_1 and t_2 contains both $\pi(t_1)$ and $\pi(t_2)$.* □

Observation 12 *Let t_1, t_2 be two terminals in distinct triangles T_1 and T_2 respectively of a clause gadget of $\mathcal{G}_i$ for $i \geq 2$. Then all paths from t_1 to t_2 contain the top vertex of both T_1 and T_2.*

We need the following lemma to prove the correctness of the reduction. Proof of this lemma goes closely along the line of Lemma 3 in [26].

Lemma 27 *Let $i \geq 2$ be an integer and consider the clause gadget $\mathcal{G}_i$ whose height is i with $r \geq 4$ literals where $2^{i-1} < r \leq 2^i$. Then the following are satisfied.*

1. *S is a feedback vertex set in $\mathcal{G}_i$ if and only if S intersects every triangle in $\mathcal{G}_i$.*
2. *Any feedback vertex set S has size at least $r - 2$.*
3. *For any feedback vertex set S in $\mathcal{G}_i$ of size at most $r - 2$, there exists two distinct terminals that are connected by a path.*
4. *For any pair of distinct terminals $\{t, t'\}$ of $\mathcal{G}_i$, there is a feedback vertex set $S \subset V(\mathcal{G}_i)$ of size $r - 2$ such that $\{t, t'\}$ is the only pair of terminals that are connected by a path in $\mathcal{G}_i \setminus S$.*

Proof of Lemma 26: We prove that ϕ is satisfiable if and only if G_ϕ has a feedback vertex set of size at most ℓ.
($\Rightarrow$) Let ϕ be satisfiable. Then, there exists a satisfying assignment to variables $x_1, \ldots, x_n$. Now, we construct a feedback vertex set D as follows. If x_i is set to true by the assignment and that satisfies the clause C_j, then we pick t_i into D otherwise we pick f_i into D. Now, corresponding to x_i, there is a literal in every copy of the clause gadget C_j. Let the terminal vertex corresponding to literal x_i be $y_{j,r,p}$. Then we fix $(y_{j,r,p}, y_{j,r,q})$ as the pair of literals where $y_{j,r,q}$ is a different literal of C_j. And we use Lemma 27 to pick exactly $d_j - 2$ vertices from the clause gadget of C_j such that those vertices make the corresponding clause gadget acyclic and $(y_{j,r,p}, y_{j,r,q})$ is the only pair of literals which are connected by a path in this clause gadget. We pick vertices in this way for every such clause gadgets. Similarly we pick vertices from the other clause gadgets in a similar way. Hence we construct D with ℓ vertices. Now, we claim that D is a feedback vertex set of G_ϕ. Suppose not. Then there exists a cycle in $G_\phi \setminus D$. Now, that cycle must use vertices from a variable gadget and also at least two vertices from some of the clause gadgets. By Lemma 27, we know that there exists only one pair of terminal vertices that are connected by a path in $G_\phi \setminus D$. Now, by construction of D, if for a pair of terminal vertices, the path survives after deletion of D, then one of its literals corresponding to that pair of terminal vertices is picked in D. Therefore, such a cycle cannot exist. Therefore, D is a feedback vertex set of G_ϕ.
($\Leftarrow$) Let D be a feedback vertex set of G_ϕ of size ℓ. Note that D picks exactly one vertex from every variable gadget and exactly one vertex from every triangle of clause gadget (all triangles of clause gadgets are pairwise disjoint anyway). Now, for every variable x_i exactly one of t_i and f_i is present in D. Note that in case D picks e_i, then we substitute e_i by t_i or f_i arbitrarily. If D picks t_i then we assign x_i to *true*, otherwise D picks f_i and we assign x_i to *false* to construct an assignment in $\{0,1\}^n$. Now we claim that ϕ is satisfiable. Suppose not. Then there exists a clause C_j which is not satisfied

by this assignment. Then, all literals of C_j are assigned false. Now by the property of a minimum feedback vertex set D, there exists a pair of terminal vertex in every pair of clause gadgets who are connected by a path in $G_\phi \setminus D$. Since there are d^2 copies of every clause gadget, there are two clause gadgets where same pair of terminal vertices are connected by a path in both the copies. Now, since the corresponding literal vertices are not in D, a cycle containing vertices from variable gadgets and clause gadgets are created. It contradicts the fact that D is a feedback vertex set of G_ϕ. Therefore, this assignment satisfies ϕ. □

Now to use this transformation to get a lower bound on the kernel size, we have to define *Oracle Communication Protocol.*

Definition 12 (Oracle Communication Protocol) (See [11]) Let $L \subseteq \Sigma^*$ be a language. An oracle communication protocol for language L is a communication protocol with two players Alice and Bob. Alice is given an input $x \in \Sigma^*$ and can only use $poly(|x|)$ time for her computations. Player Bob is computationally unbounded, but not given any part of x. At the end of the protocol, Alice should be able to decide $x \in L$ using help (communication) from Bob. The cost of the protocol is the number of bits communicated between Alice and Bob.

Theorem 12 *([11]) d-CNF-SAT has no oracle communication protocol of cost $\mathcal{O}(n^{d-\epsilon})$ for any $d \geq 3, \epsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$ where n is the number of variables of the input formula.*

Using Lemma 26 and Theorem 12, we have the following lemma.

Lemma 28 FVS-Mock-d-Forest *has no oracle communication protocol of cost $\mathcal{O}(k^{d+2-\epsilon})$ for any $\epsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.*

Proof Suppose FVS-Mock-d-Forest has an oracle communication protocol $\mathcal{P}_1$ of cost $\mathcal{O}(k^{d_1})$ where $d_1 = d + 2 - \epsilon$ for some $\epsilon > 0$. Then we get an oracle communication protocol for $(d+2)$-CNF-SAT as follows. Given ϕ an instance of $(d+2)$-CNF-SAT with n variables, Alice (polynomially bounded player) runs the parameter preserving transformation (as described above) and get (G_ϕ, S, ℓ) such that $|S| = 2n$, $G \setminus S$ is a mock-forest where every component has at most d cycles. Now, Alice and Bob use the protocol $\mathcal{P}_1$ for FVS-Mock-d-Forest and get the answer (Yes/No) for the $(d+2)$-CNF-SAT. So, we get an oracle communication protocol of cost $\mathcal{O}(n^{d+2-\epsilon})$ for $(d+2)$-CNF-SAT. This implies $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. So, FVS-Mock-$d$-Forest has no oracle communication protocol of cost $\mathcal{O}(k^{d+2-\epsilon})$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. □

Theorem 13 FVS-d-Mock-Forest *has no kernel of $\mathcal{O}(k^{d+2-\epsilon})$ size for every $d \geq 2, \epsilon > 0$ unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.*

Proof Suppose FVS-Mock-d-Forest has a kernel consisting of $\mathcal{O}(k^{d+2-\epsilon})$ size (or edges). We need $\log_2(\mathcal{O}(k^{d+2-\epsilon}))$, i.e. $\mathcal{O}(k^{d+2-\epsilon} \cdot d \cdot \log_2 k)$ bits to represent this kernel. So, FVS-Mock-d-Forest has a kernel with $\mathcal{O}(k^{d+2-\epsilon'})$ bits for some $\epsilon' > 0$. Then in the oracle communication protocol, Alice who is allowed to do polynomial time computation first computes this kernel with $\mathcal{O}(k^{d+2-\epsilon'})$ bits and sends this entire kernel to Bob. Now Bob is a computationally unbounded player. He can compute and return Yes or No answer correctly. This protocol has cost $\mathcal{O}(k^{d+2-\epsilon'})$ bits. Then by Lemma 28 we have $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. This proves the theorem. □

5 Conclusion

Continuing the line of research on structural parameterization of Feedback Vertex Set initiated in [26], we have given substantially improved kernel bounds and considered several other structural parameterization of Feedback Vertex Set where the parameter is the deletion distance to a graph class where FVS is polynomially solvable. See Table 1 for the state of the art on some of the structural parameterizations of FVS, including some results in this paper. A clear open problem is to improve the runtime of the FPT algorithms and the sizes of the kernel we considered in this paper. Improving the FPT runtime of FVS parameterized by solution size remains as an open problem by itself. The existence of an FPT algorithm for Feedback Vertex Set parameterized by deletion distance to a (sub)-cubic graph remains open, and we do not even know an XP algorithm for the problem. We have shown that the related edge version is fixed-parameter tractable, though we do not know whether it admits a polynomial kernel.

Table 1: Summary of Results: Results marked $\star$ indicate our results

Parameterization Considered	FPT Algorithm	Polynomial Kernel
FVS-Solution-Size	$\mathcal{O}^*(3.619^k)$ [28]	$\mathcal{O}(k^2)$ vertices and edges [36]
FVS-High-Degree	$\mathcal{O}^*(2^k)$ $\star$	No polynomial kernel $\star$
FVS-Vertex-Deletion-to-Sub-Cubic	Open	No polynomial kernel $\star$
FVS-Edge-Deletion-to-Sub-Cubic	$\mathcal{O}^*(4^k)$ $\star$	Open
FVS-Vertex-Clique-Cover	W[1]-hard [26]	No kernel
FVS-deletion to $(c,1)$-Graph	$\mathcal{O}(3.148^k \cdot n^{\mathcal{O}(c)})$ $\star$	No polynomial kernel [4]
FVS-SVD	$\mathcal{O}^*(3.148^k)$ $\star$	No polynomial kernel [4]
FVS-CVD	$\mathcal{O}^*(5^k)$ $\star$	No polynomial kernel [4]
FVS-Pseudo-Forest	$\mathcal{O}^*((8+\epsilon)^k)$ [18]	$\mathcal{O}(k^{10})$ vertices and edges [26]
FVS-Pseudo-Forest	$\mathcal{O}^*((8+\epsilon)^k)$ [18]	$\mathcal{O}(k^7)$ vertices$\star$
FVS-Mock-Forest	$\mathcal{O}^*((8+\epsilon)^k)$ [18]	No polynomial kernel [26]
FVS-Mock-d-Forest	$\mathcal{O}^*((8+\epsilon)^k)$ [18]	$\mathcal{O}(k^{3d+3})$ vertices$\star$

References

1. Balas, E., Yu, C.S.: On graphs with polynomially solvable maximum-weight clique problem. Networks **19**(2), 247–253 (1989)
2. Bodlaender, H.L., Cygan, M., Kratsch, S., Nederlof, J.: Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. Inf. Comput. **243**, 86–111 (2015)
3. Bodlaender, H.L., van Dijk, T.C.: A Cubic Kernel for Feedback Vertex Set and Loop Cutset. Theory Comput. Syst. **46**(3), 566–597 (2010)
4. Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernelization Lower Bounds by Cross-Composition. SIAM J. Discrete Math. **28**(1), 277–305 (2014)
5. Bodlaender, H.L., Thomassé, S., Yeo, A.: Kernel bounds for disjoint cycles and disjoint paths. Theor. Comput. Sci. **412**(35), 4570–4578 (2011)
6. Boral, A., Cygan, M., Kociumaka, T., Pilipczuk, M.: A Fast Branching Algorithm for Cluster Vertex Deletion. Theory Comput. Syst. **58**(2), 357–376 (2016)
7. Cao, Y., Chen, J., Liu, Y.: On Feedback Vertex Set: New Measure and New Structures. Algorithmica **73**(1), 63–86 (2015)
8. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015)
9. Cygan, M., Nederlof, J., Pilipczuk, M., Pilipczuk, M., van Rooij, J.M.M., Wojtaszczyk, J.O.: Solving Connectivity Problems Parameterized by treewidth in Single Exponential Time. CoRR **abs/1103.0534** (2011)
10. Cygan, M., Pilipczuk, M.: Split Vertex Deletion meets Vertex Cover: New fixed-parameter and exact exponential-time algorithms. Inf. Process. Lett. **113**(5-6), 179–182 (2013)
11. Dell, H., van Melkebeek, D.: Satisfiability Allows No Nontrivial Sparsification unless the Polynomial-Time Hierarchy Collapses. J. ACM **61**(4), 23:1–23:27 (2014)
12. Diestel, R.: Graph Theory, 4th Edition, *Graduate texts in mathematics*, vol. 173. Springer (2012)
13. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer (2013)
14. Fellows, M.R., Jansen, B.M.P., Rosamond, F.A.: Towards fully multivariate algorithmics: Parameter ecology and the deconstruction of computational complexity. Eur. J. Comb. **34**(3), 541–566 (2013)
15. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series. Springer (2006)
16. Fomin, F., Strømme, T.: Vertex cover structural parameterization revisited. In: Graph-Theoretic Concepts in Computer Science - 42nd International Workshop, WG 2016, Istanbul, Turkey, June 22-24, 2016, Revised Selected Papers, pp. 171–182 (2016)
17. Fomin, F.V., Lokshtanov, D., Misra, N., Saurabh, S.: Planar F-Deletion: Approximation, Kernelization and Optimal FPT Algorithms. In: IEEE Symposium of Foundations of Computer Science FOCS, pp. 470–479 (2012)
18. Fomin, F.V., Lokshtanov, D., Panolan, F., Saurabh, S.: Efficient Computation of Representative Families with Applications in Parameterized and Exact Algorithms. J. ACM **63**(4), 29:1–29:60 (2016)
19. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. J. Comput. Syst. Sci. **77**(1), 91–106 (2011)
20. Fujito, T.: A Unified Approximation Algorithm for Node-deletion Problems. Discrete Applied Mathematics **86**(2-3), 213–231 (1998)
21. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
22. Gutin, G., Kim, E.J., Lampis, M., Mitsou, V.: Vertex Cover Problem Parameterized Above and Below Tight Bounds. Theory Comput. Syst. **48**(2), 402–410 (2011)

23. Impagliazzo, R., Paturi, R., Zane, F.: Which problems have strongly exponential complexity? J. Comput. Syst. Sci. **63**(4), 512–530 (2001)
24. Jansen, B.M.P., Bodlaender, H.L.: Vertex cover kernelization revisited - upper and lower bounds for a refined parameter. Theory Comput. Syst. **53**(2), 263–299 (2013)
25. Jansen, B.M.P., Kratsch, S.: Data reduction for graph coloring problems. Inf. Comput. **231**, 70–88 (2013)
26. Jansen, B.M.P., Raman, V., Vatshelle, M.: Parameter Ecology for Feedback Vertex Set. Tsinghua Science and Technology **19**(4), 387–409 (2014)
27. Kloks, T., Liu, C., Poon, S.: Feedback Vertex Set on Chordal Bipartite Graphs. CoRR **abs/1104.3915v2** (2011)
28. Kociumaka, T., Pilipczuk, M.: Faster deterministic Feedback Vertex Set. Inf. Process. Lett. **114**(10), 556–560 (2014)
29. Kolay, S., Panolan, F.: Parameterized Algorithms for Deletion to (r, l)-Graphs. In: Proceedings of Foundation of Software Technology and Theoretical Computer Science FSTTCS, pp. 420–433 (2015)
30. Kratsch, D., Müller, H., Todinca, I.: Feedback Vertex Set on AT-free graphs. Discrete Applied Mathematics **156**(10), 1936–1947 (2008)
31. Majumdar, D.: Structural Parameterizations of Feedback Vertex Set. In: 11th International Symposium of Parameterized and Exact Computation IPEC, pp. 21:1–21:16 (2016)
32. Majumdar, D., Raman, V.: FPT algorithms for FVS parameterized by split and cluster vertex deletion sets and other parameters. In: International Frontiers of Algorithmics Workshop FAW, pp. 209–220 (2017)
33. Majumdar, D., Raman, V., Saurabh, S.: Kernels for Structural Parameterizations of Vertex Cover - case of Small Degree Modulators. In: 10th International Symposium of Parameterized and Exact Computation IPEC, pp. 331–342 (2015)
34. M.Cygan, Nederlof, J., M.Pilipczuk, M.Pilipczuk, Rooij, J., J.O.Wojtaszczyk: Solving Connectivity Problems Parameterized by Treewidth in Singly Exponential Time. In: IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011, pp. 150–159 (2011)
35. Rizzi, R.: Minimum Weakly Fundamental Cycle Bases Are Hard To Find. Algorithmica **53**(3), 402–424 (2009)
36. Thomassé, S.: A $4k^2$ kernel for Feedback Vertex Set. ACM Trans. Algorithms **6**(2) (2010)
37. Ueno, S., Kajitani, Y., Gotoh, S.: On the nonseparating independent set problem and feedback set problem for graphs with no vertex degree exceeding three. Discrete Mathematics **72**(1-3), 355–360 (1988)