An Approximate Kernel for Connected Feedback Vertex Set

M. S. Ramanujan
University of Warwick, UK
https://warwick.ac.uk/fac/sci/dcs/people/ramanujan_sridharan/
R.Maadapuzhi-Sridharan@warwick.ac.uk

Abstract

The Feedback Vertex Set problem is a fundamental computational problem which has been the subject of intensive study in various domains of algorithmics. In this problem, one is given an undirected graph G and an integer k as input. The objective is to determine whether at most k vertices can be deleted from G such that the resulting graph is acyclic. The study of preprocessing algorithms for this problem has a long and rich history, culminating in the quadratic kernelization of Thomasse [SODA 2010].

However, it is known that when the solution is required to induce a *connected* subgraph (such a set is called a connected feedback vertex set), a polynomial kernelization is unlikely to exist and the problem is NP-hard to approximate below a factor of 2 (assuming the Unique Games Conjecture).

In this paper, we show that if one is interested in only preserving *approximate* solutions (even of quality arbitrarily close to the optimum), then there is a drastic improvement in our ability to preprocess this problem. Specifically, we prove that for every fixed $0 < \varepsilon < 1$, graph G, and $k \in \mathbb{N}$, the following holds.

> There is a polynomial time computable graph G' of size $k^{\mathcal{O}(1)}$ such that for every $c \geq 1$, any c-approximate connected feedback vertex set of G' of size at most k is a $c \cdot (1+\varepsilon)$-approximate connected feedback vertex set of G.

Our result adds to the set of approximate kernelization algorithms introduced by Lokshtanov et al. [STOC 2017]. As a consequence of our main result, we show that Connected Feedback Vertex Set can be approximated within a factor $\min\{\mathrm{OPT}^{\mathcal{O}(1)}, n^{1-\delta}\}$ in polynomial time for some $\delta > 0$.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Mathematics of computing → Approximation algorithms

Keywords and phrases Parameterized Complexity, Kernelization, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2019.77

1 Introduction

Polynomial time preprocessing is one of the widely used methods to tackle NP-hardness in practice, and the area of *kernelization* has been extremely successful in laying down a mathematical framework for the design and rigorous analysis of preprocessing algorithms for decision problems. The central notion in kernelization is that of a *kernelization algorithm*, which is a preprocessing algorithm that runs in polynomial time and transforms a "large" instance of a decision problem into a significantly smaller, but equivalent instance (called a *kernel*). Over the last decade, the area of kernelization has seen the development of a wide range of tools to design preprocessing algorithms, as well as a rich theory of lower bounds based on assumptions from complexity theory [2, 6, 3, 15, 5, 9, 18, 7, 17]. We refer the reader to the survey articles by Kratsch [19] or Lokshtanov et al. [20] for relatively recent developments, or the textbooks [4, 8], for an introduction to the field.

An "efficient preprocessing algorithm" in this setting is referred to as a *polynomial kernelization* and is simply a kernelization whose output has size bounded polynomially in a parameter of the input. The central classification task in the area is to classify any problem as one which has a polynomial kernel, or as one that does not.

One fundamental class of problems for which polynomial kernels have been ruled out under certain complexity theoretic hypotheses, is the class of "subgraph hitting" problems *with a connectivity constraint*. It is well-known that placing connectivity constraints on certain subgraph hitting problems can have a dramatic effect on their amenability to preprocessing. A case in point is the classic Vertex Cover problem. This problem is known to admit a kernelization whose output has $\mathcal{O}(k)$ vertices [4]. However, the Connected Vertex Cover problem is amongst the earliest problems shown to exclude a polynomial kernel [7] (under a complexity theoretic hypothesis) and this lower bound immediately rules out the possibility of such a kernelization for numerous well-studied generalizations of it. Consequently, obtaining a finer understanding of the impact of connectivity constraints on the limits of preprocessing is an important objective in furthering the study of preprocessing techniques. This is even more relevant when one intends to run approximation algorithms or heuristics on the preprocessed instance.

Unfortunately, the existing notion of kernels, having been built around decision problems, does not combine well with approximation algorithms and heuristics. In particular, in order for kernels to be useful, one is required to solve the preprocessed instance *exactly*. However, this may not always be possible and the existing theory of kernelization says nothing about the inference of useful information from a good *approximate* solution for the preprocessed instance. In order to facilitate the rigorous analysis of preprocessing algorithms in conjunction with approximation algorithms, Lokshtanov et al. [22] introduced the notion of *α-approximate kernels*. Informally speaking, an α-approximate kernelization is a polynomial-time algorithm that, given an instance (I, k) of a *parameterized problem*, outputs an instance (I', k') such that $|I'| + k' \leq g(k)$ for some computable function g and any c-approximate solution to the instance (I', k') can be turned in polynomial time into a $(c \cdot \alpha)$-approximate solution to the original instance (I, k).

As earlier, the notion of "efficiency" in this context is captured by the function g being polynomially bounded, in which case we call this algorithm, an *α-approximate polynomial kernelization*. We refer the reader to Section 2 for a formal definition of all terms related to (approximate) kernelization.

In their work, Lokshtanov et al. considered several problems which are known to exclude polynomial kernels and presented an α-approximate polynomial kernelization for these problems for *every* fixed $\alpha > 1$. This implies that allowing for an arbitrarily small amount of error while preprocessing can drastically improve the extent to which the input instance can be reduced, even when dealing with problems for which polynomial kernels have been ruled out under the existing theory of lower bounds. In particular, they showed that the Connected Vertex Cover problem admits an α-approximate polynomial kernelization for every $\alpha > 1$. Their result provided a promising starting point towards obtaining a refined understanding of the role played by connectivity constraints in relation to preprocessing for covering problems on graphs.

In this paper, we consider one of the most natural generalizations of Connected Vertex Cover, the Connected Feedback Vertex Set (CFVS) problem defined as follows. In this problem, the input is a graph G and integer k (the parameter). The goal is to decide whether or not there is a set $S \subseteq V(G)$ of size at most k such that $G[S]$ is connected and $G - S$ is acyclic? The set S is called a connected feedback vertex set of G.

Misra et al. [23] were the first to study the CFVS problem from the point of view of parameterized complexity and obtained a single-exponential fixed-parameter algorithm, that is, an algorithm running in time $2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$, where n is the number of vertices in the input. They also observed that a straightforward reduction from Connected Vertex Cover implies that CFVS is also unlikely to admit a polynomial kernelization under standard complexity theoretic hypotheses. This raises the natural question of the existence of approximate kernelizations for this problem and the tradeoffs between preprocessing speed, output size and loss in the quality of the preserved solution.

Our results

A formal definition of α-approximate kernels can be found in Section 2.

▶ **Theorem 1.** *For every fixed $0 < \varepsilon < 1$,* Connected Feedback Vertex Set *has a $(1+\varepsilon)$-approximate kernelization of polynomial size.*

Note that the exponent in the size of our kernelization depends on ε.

The proof techniques we use in order to prove Theorem 1 also lead to a polynomial time approximation for this problem (Theorem 3) via the following *parameterized* approximation.

▶ **Theorem 2.** *There is an algorithm that given a graph G and $k \in \mathbb{N}$, runs in polynomial time and either correctly concludes that G has no connected feedback vertex set of size at most k or returns a connected feedback vertex set of G of size $k^{\mathcal{O}(1)}$.*

As a direct consequence of Theorem 2, we obtain the following result.

▶ **Theorem 3.** *There is a $0 < \delta < 1$ such that* Connected FVS *can be approximated within a factor $\min\{\mathrm{OPT}^{\mathcal{O}(1)}, n^{1-\delta}\}$ in polynomial time.*

Proof. By iteratively invoking Theorem 2 for $k = 1, \ldots, n$, one can find the least k for which the algorithm *does not* return a negative answer and returns a connected feedback vertex set of the input graph of size at most k^c for some fixed $c > 0$. Since $\mathrm{OPT} \geq k$, it follows that the returned solution has size at most OPT^c. Moreover, this algorithm always returns a solution whose size is bounded by n. The theorem follows from fact that the approximation ratio guaranteed by this algorithm is at most $\min\{\mathrm{OPT}^{c-1}, \frac{n}{\mathrm{OPT}}\} \leq n^{1-\frac{1}{c}}$. ◀

Our approximation result complements the classic result of Yannakakis [25] who showed that it is NP-hard to approximate within a factor-$\mathcal{O}(n^{1-\delta})$ (*for any* $\delta > 0$) the minimum number of vertices to delete from a graph such that the *resulting graph* is connected and has a property Π which is hereditary, non-trivial, "interesting" on connected graphs, and is determined by the blocks of the graph. In particular, this result holds if Π = acyclicity.

Related work on connected hitting set problems. Grigoriev and Sitters [16] studied the design of approximation algorithms for the Connected Feedback Vertex Set problem on planar graphs and obtained a Polynomial Time Approximation Scheme (PTAS), building upon the result of Escoffier et al. [13] for Connected Vertex Cover. Eiben et al. [11] obtained an approximate kernelization for the Connected $\mathcal{H}$-hitting set problem where $\mathcal{H}$ is a fixed set of graphs and the solution is a minimum set of vertices which induces a connected subgraph and hits all copies of graphs in $\mathcal{H}$, in G. Recently, Eiben et al. [12] obtained approximate kernelizations for the Connected Dominating Set problem on various sparse graph classes.

2 Preliminaries

A set $S \subseteq V(G)$ such that $G - S$ is a forest is called a *feedback vertex set* of G. For a path P, we denote by $V_{\mathsf{int}}(P)$ the set of internal vertices of the path P. Similarly, we denote by $V_{\mathsf{end}}(P)$ the set of endpoints of P. Two paths P_1 and P_2 are said to be *internally vertex disjoint* if $V_{\mathsf{int}}(P_1) \cap V_{\mathsf{int}}(P_2) = \emptyset$. For a graph G, we denote by $\mathcal{CC}(G)$ the set of connected components of G. For ease of presentation, we will abuse notation and interchangeably refer to $X \subseteq V(G)$ both as a vertex set and as a connected component of G if clear from the context.

▶ **Definition 4.** *Let G be a graph and $x, y \in V(G)$. Let $\mathcal{P}$ be a set of internally vertex-disjoint x-y paths in G. Then, we call $\mathcal{P}$ an x-y* flow. *The* value *of this flow is $|\mathcal{P}|$.*

Recall that Menger's theorem states that for distinct *non-adjacent* vertices x and y, the size of the smallest x-y separator is precisely the value of the maximum x-y flow in G.

Parameterized problems and (approximate) kernels. A parameterized problem Π is a subset of $\Gamma^* \times \mathbb{N}$ for some finite alphabet Γ. An instance of a parameterized problem consists of a pair (x, k), where k is called the parameter. We assume that k is *given in unary* and hence $k \leq |x|$.

▶ **Definition 5** (Kernelization)**.** *Let $\Pi \subseteq \Gamma^* \times \mathbb{N}$ be a parameterized problem and g be a computable function. We say that* Π admits a kernel of size g *if there exists an algorithm referred to as a* kernelization *(or a* kernel*) that, given $(x, k) \in \Gamma^* \times \mathbb{N}$, outputs in time polynomial in $|x| + k$, a pair $(x', k') \in \Gamma^* \times \mathbb{N}$ such that (a) $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$, and (b) $\max\{|x'|, k'\} \leq g(k)$. If $g(k) = k^{\mathcal{O}(1)}$ then we say that* Π admits a polynomial kernel.

▶ **Definition 6** ([22])**.** *A parameterized optimization (minimization or maximization) problem is a computable function $\Pi : \Sigma^* \times \mathbb{N} \times \Sigma^* \to \mathbb{R} \cup \{\pm\infty\}$.*

The *instances* of a parameterized optimization problem Π are pairs $(I, k) \in \Sigma^* \times \mathbb{N}$, and a *solution* to (I, k) is simply a string $s \in \Sigma^*$, such that $|s| \leq |I| + k$. The *value* of the solution s is $\Pi(I, k, s)$.

Since we only deal with a minimization problem in this work, we state some of the definitions only in terms of minimization problems when the definition for maximization problems is analogous. The parameterized optimization version of Connected Feedback Vertex Set is a minimization problem with the optimization function $\mathrm{CFVS} : \Sigma^* \times \mathbb{N} \times \Sigma^* \to \mathbb{R} \cup \{\infty\}$ defined as follows.

$$\mathrm{CFVS}(G, k, S) = \begin{cases} \infty & \text{if } S \text{ is not a connected feedback vertex set of } G, \\ \min\{|S|, k+1\} & \text{otherwise.} \end{cases}$$

▶ **Definition 7** ([22])**.** *For a parameterized minimization problem Π, the* optimum value *of an instance $(I, k) \in \Sigma^* \times \mathbb{N}$ is $OPT_\Pi(I, k) = \min_{\substack{s \in \Sigma^* \\ |s| \leq |I|+k}} \Pi(I, k, s)$.*

Consequently, in the case of Connected Feedback Vertex Set, we define

$$\mathrm{OPT}(G, k) = \min_{S \subseteq V(G)} \mathrm{CFVS}(G, k, S).$$

▶ Remark 8 (Restricting our interest to solutions of size at most k). A reader encountering this particular definition of parameterized optimization problems for the first time might find the choice of $k+1$ as a threshold a bit counter-intuitive because when one combines it with the natural notion of approximate solutions in the most intuitive way, the size of the solution would appear to always exceed $k+1$, thus being normalized by this explicit threshold.

However, this definition is in fact equivalent (upto constant factors) to the more seemingly natural definition and in addition allows us to define the problem we are tackling independently of the (approximation factor of) algorithms for the problem. An additional point which we encourage the reader to keep in mind is the following. We consider k as a threshold; for solutions of size at most k we care about what their size is, while all solutions of size larger than k are *equally bad* in our eyes, and are consequently assigned value $k+1$. We point the interested reader to Section 2.1, [22] and Section 3.2, [21] for an in-depth discussion of these definitions and their motivations.

We now recall the other relevant definitions from [22] regarding *approximate kernels*.

▶ **Definition 9** ([22]). *Let $\alpha \geq 1$ be a real number and Π be a parameterized minimization problem. An* α-approximate polynomial time preprocessing algorithm $\mathcal{A}$ *for Π is a pair of polynomial-time algorithms. The first one is called the* reduction algorithm*, and computes a map $\mathcal{R}_{\mathcal{A}} : \Sigma^* \times \mathbb{N} \to \Sigma^* \times \mathbb{N}$. Given as input an instance (I,k) of Π the reduction algorithm outputs another instance $(I',k') = \mathcal{R}_{\mathcal{A}}(I,k)$.*

The second algorithm is called the solution lifting algorithm*. This algorithm takes as input an instance $(I,k) \in \Sigma^* \times \mathbb{N}$ of Π, the output instance (I',k') of the reduction algorithm, and a solution s' to the instance (I',k'). The solution lifting algorithm works in time polynomial in $|I|$,k,$|I'|$,k' and s', and outputs a solution s to (I,k) such that $\frac{\Pi(I,k,s)}{OPT(I,k)} \leq \alpha \cdot \frac{\Pi(I',k',s')}{OPT(I',k')}$.*

The size *of a polynomial time preprocessing algorithm $\mathcal{A}$ is a function $size_{\mathcal{A}} : \mathbb{N} \to \mathbb{N}$ defined as $size_{\mathcal{A}}(k) = \sup\{|I'| + k' : (I',k') = \mathcal{R}_{\mathcal{A}}(I,k), I \in \Sigma^*\}$.*

▶ **Definition 10** ([22], α-approximate kernelization). *An* α-approximate kernelization *(or* α-approximate kernel*) for a parameterized optimization problem Π, and real $\alpha \geq 1$, is an α-approximate polynomial time preprocessing algorithm $\mathcal{A}$ for Π such that $size_{\mathcal{A}}$ is upper bounded by a computable function $g : \mathbb{N} \to \mathbb{N}$. We say that $\mathcal{A}$ is an α-approximate polynomial kernelization if g is a polynomial function.*

▶ **Definition 11** ([22]). *A* polynomial size approximate kernelization scheme *(PSAKS) for a parameterized optimization problem Π is a family of α-approximate polynomial kernelization algorithms, with one such algorithm for every $\alpha > 1$.*

Least Common Ancestor-Closure of sets in trees. For a rooted tree T and vertex set $M \subseteq V(T)$ the least common ancestor-closure (*LCA-closure*) LCA-closure(M) is obtained by the following process. Initially, set $M' = M$. Then, as long as there are vertices x and y in M' whose least common ancestor w is not in M', add w to M'. When the process terminates, output M' as the LCA-closure of M. The following folklore lemma summarizes the properties of LCA-closures which we will use in this paper.

▶ **Lemma 12.** *Let T be a rooted tree, $M \subseteq V(T)$ and $M' =$ LCA-closure(M). Then $|M'| \leq 2|M|$ and for every connected component C of $T - M'$, $|N(C)| \leq 2$. Moreover, the number of connected components of $T - M'$ which have exactly 2 neighbors in M' is at most $|M'| - 1$.*

▶ **Observation 2.1.** *Let T be a rooted tree and T' a subtree of T with a unique neighbor in $V(T) \setminus V(T')$. If $M \subseteq V(T) \setminus V(T')$, then* LCA-closure$(M)$ *is disjoint from $V(T')$.*

3 Overview of our Techniques

This section is devoted to an overview of the proof techniques we use to obtain our results. Fix $0 < \varepsilon < 1$ and let (G, k) be the input. Our initial objective is to identify a partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ of $V(G)$, where $|\mathcal{B}| = k^{\mathcal{O}(1)}$, $G - \mathcal{B}$ is acyclic and there are no edges between $\mathcal{A}$ and $\mathcal{C}$. In other words, $\mathcal{B}$ separates $\mathcal{A}$ and $\mathcal{C}$. Moreover, we will be able to prove that the vertices in $\mathcal{C}$ only play the role of "connectors" and removing them from a connected feedback vertex set S of G may disconnect $G[S]$, but will still leave a subset of S which hits all cycles in G. On the other hand, the interaction of vertices in $\mathcal{A}$ with the solution S could be much more complex. However, the number of connected components of $G[\mathcal{A}]$ will be shown to be $k^{\mathcal{O}(1)}$ and these can be shown to have a highly structured neighborhood in $\mathcal{B}$. For instance, we will be able to ensure (after a small modification to G) that the neighborhood of any connected component of $G[\mathcal{A}]$ can be partitioned into 2 sets T and J such that T is part of *every* feedback vertex set of G and $|J| = 2$. For every such component, the sets T and J can be efficiently computed from $\mathcal{A}$.

Once we have this partition in hand, we focus on each connected component of $G[\mathcal{A}]$ separately and from each component we identify a set of $k^{\mathcal{O}(f(1/\varepsilon))}$ vertices which, together with $\mathcal{B}$ and a $k^{\mathcal{O}(f(1/\varepsilon))}$ sized subset of $\mathcal{C}$ cover a $(1+\varepsilon)$-approximate solution. Finally, the remaining vertices are discarded by either deleting or contracting edges as appropriate. We note that the high level approach of trying to identify "hitters" and "connectors" among the vertices is quite natural and has been used in other work [22, 11, 12]. However, the structure is much more complex in our case due to the highly non-local nature of the forbidden structures (cycles) and the fact that there is no clear way of completely separating the two tasks of hitting cycles and connectivity. In fact, the main difficulty arises from the need to detect and control subsets of vertices which are critical with respect to both objectives simultaneously.

We now proceed to a slightly more detailed overview of the steps in our algorithm. Let δ be a positive constant such that $(1+\delta)^3 \leq 1+\varepsilon$. We also fix a constant $\rho = 2^{\mathcal{O}(1/\delta)^2}$ satisfying certain appropriate inequalities. As mentioned, our PSAKS for Connected Feedback Vertex Set has two main parts: a structural decomposition and a reduction using marking rules.

3.1 Structural Decomposition

The first part of our PSAKS is the decomposition given by Lemma 13 below(also see Figure 1). To get the required partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ we set $\mathcal{B} = H \cup X \cup Z$ where H, X, Z are as defined in the statement of Lemma 13. We then set $\mathcal{C}$ to be the set of vertices in connected components of $G - (H \cup X \cup Z)$ which are adjacent to at most one vertex of Z and $\mathcal{A}$ to be the rest of the vertices in $G - (H \cup X \cup Z)$.

▶ **Lemma 13.** *There is a polynomial-time algorithm that given a pair (G, k), either correctly concludes that G has no connected feedback vertex set of size at most k or outputs pairwise vertex disjoint subsets $H, X, Z \subseteq V(G)$ satisfying the following properties.*

1. $|H| \leq (1 + \frac{\delta}{2})k, |X| = \mathcal{O}(k)$, $|Z| = \mathcal{O}(k^6)$.
2. *$H \cup X$ is a feedback vertex set of G and if G has a connected feedback vertex set S of size at most k, then there is one of size at most $(1 + \frac{\delta}{2})|S|$ that contains H.*
3. *Every connected component of the graph $\widetilde{G_Z} = G - (H \cup X \cup Z)$ is adjacent to at most 2 vertices of Z and there are $\mathcal{O}(k^6)$ connected components in $\widetilde{G_Z}$ which are adjacent to exactly 2 vertices of Z.*

4. *No vertex of X has a neighbor in a connected component of $\widetilde{G_Z}$ which is adjacent to 2 vertices of Z and moreover, every vertex in any such component is adjacent to vertices in $\mathcal{O}(1/\delta)$ connected components of $G[H]$.*
5. *For every connected component D in the graph $\widetilde{G_Z}$ with at most 1 neighbor in Z and for every minimal feedback vertex set S of $G-H$ of size at most $2k$, $|N(D) \setminus (H \cup S)| \leq 1$.*

Proof. (Sketch for the construction of H, X, Z). We only sketch the construction of these sets here.

Step 1: This step is inspired by Fomin et al. [14]. However, since we need to handle connectivity constraints, we need to enhance the output of this step further with several problem specific features. Let $H = \{v_1, \ldots, v_r\}$ be a maximal set such that for every $i \in [r]$, there is a v_i-flower of order $2k+1$ ($2k+1$ cycles which are pairwise disjoint except for intersecting at v_i) in $G - H_{i-1}$, where $H_i = \{v_1, \ldots, v_i\}$ and $H_0 = \emptyset$. It is known that there is a polynomial time algorithm that, given G, v, ℓ (assuming v is not incident on a self-loop) either outputs a v-flower of order ℓ or a set $X \subseteq V(G) \setminus \{v\}$ of $\mathcal{O}(\ell)$ vertices intersecting every cycle containing v [24]. Hence H can be computed in polynomial time. Furthermore, it can be observed that every vertex in H is part of every feedback vertex set of size at most $2k$. Therefore, if $|H| > k$, then we may correctly conclude that G has no feedback vertex set of size at most k.

Let Q be a feedback vertex set of G computed using the 2-approximation algorithm of Bafna et al. [1]. If $|Q| > 2k$, then we may correctly conclude that G has no feedback vertex set of size at most k. Otherwise, $H \subseteq Q$ and we define $X = Q \setminus H$. Due to the maximality of H, it follows that for every $x \in X$, there is a set P_x which is disjoint from x, has size $\mathcal{O}(k)$ and intersects every cycle containing x in $G - H$.

For every pair $x, y \in X$ such that there is no x-y flow of value $2k+3$ in the graph $G_{xy} = G - (H \cup (X \setminus \{x, y\}))$, we denote by Z_{xy} an arbitrarily chosen minimum x-y separator in the graph G'_{xy}, where $G'_{xy} = G_{xy}$ if $(x, y) \notin E(G)$ and $G'_{xy} = G_{xy} - (x, y)$ otherwise. By Menger's Theorem, for every such pair x, y, the size of the set Z_{xy} is at most $2k+2$. We define $J = (\bigcup_{x \in X} P_x \cup \bigcup_{x,y \in X} Z_{xy}) \setminus (X \cup H)$, where for every pair $x, y \in X$ such that there *is* an x-y flow of value at least $2k+3$ in the graph $G_{xy} = G - (H \cup (X \setminus \{x, y\}))$, Z_{xy} is defined to be $\emptyset$. We now define $Y = \mathsf{LCA\text{-}closure}(J)$ with the LCA-closure taken in the graph $G - (H \cup X)$ where each tree is arbitrarily rooted. Since $|H| \leq k$ and $|X| \leq 2k$, it follows that $|J| = \mathcal{O}(k^3)$ and Lemma 12 implies that $|Y| = |\mathsf{LCA\text{-}closure}(J)| = \mathcal{O}(k^3)$, every connected component of the graph $G - (H \cup X \cup Y)$ is adjacent to at most 2 vertices of Y and there are $\mathcal{O}(|Y|)$ of these components which are adjacent to exactly 2 vertices of Y.

Step 2: Since Step 1 guarantees that there are no cycles in $G - (H \cup Y \cup (X \setminus \{x\}))$ for any $x \in X$ we conclude that no vertex of X can have 2 neighbors in any connected component of $\widetilde{G_Y} = G - (H \cup X \cup Y)$. Let $\mathcal{R}$ denote the set of all connected components of $\widetilde{G_Y}$ which have no neighbors in Y, let $\mathcal{Q}$ denote the set of all connected components of $\widetilde{G_Y}$ which have exactly 1 neighbor in Y, and let $\mathcal{W}$ denote the set of all connected components of $\widetilde{G_Y}$ which have exactly 2 neighbors in Y.

For every $x \in X$ and $y \in Y$, we define the set $\mathcal{J}_{xy}$ as the set of components of $\widetilde{G_Y}$ which are adjacent to x and whose neighborhood in Y is exactly $\{y\}$. We say that the set $\mathcal{J}_{x,y}$ is *rich* if $|\mathcal{J}_{xy}| \geq 2k+3$ and *poor* otherwise. We call a connected component in $\mathcal{Q}$ poor if it appears in *at least one* poor set and rich otherwise. Let $\mathcal{Q}_{\mathsf{poor}}$ denote the set of poor components in $\mathcal{Q}$. By definition, the size of the set $\mathcal{Q}_{\mathsf{poor}}$ is bounded by $|X| \cdot |Y| \cdot (2k+2) = \mathcal{O}(k^5)$. Let P_1 denote the neighborhood of X in the set of components

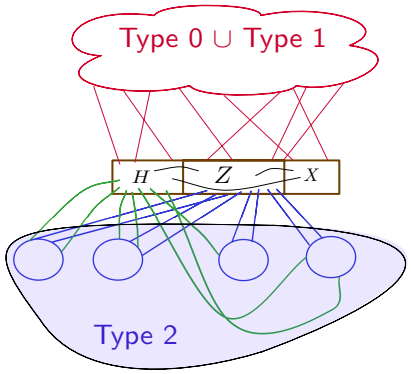

Figure 1 An illustration of the decomposition guaranteed by Lemma 13. The (blue) circles below the set $(H \cup Z \cup X)$ represent components of $G - (H \cup Z \cup X)$ with 2 neighbors in Z. Note that there are no edges between X and such a component.

in $\mathcal{Q}_{\mathsf{poor}}$. Since every vertex of X has at most 1 neighbor in each of these components, the size of P_1 is at most $|X| \cdot |\mathcal{Q}_{\mathsf{poor}}| = \mathcal{O}(k^6)$. Let P_2 denote the neighborhood of X in the set of components of $\widetilde{G_Y}$ with exactly 2 neighbors in Y. Since there are only $\mathcal{O}(k^3)$ such components (from Step 1), the size of P_2 is $\mathcal{O}(k^4)$.

Finally, we define $Z = \mathsf{LCA\text{-}closure}(Y \cup P_1 \cup P_2)$ where the LCA-closure is taken in the graph $G - (H \cup X)$ with an arbitrary rooting of each tree. From Lemma 12, it follows that $|Z| = \mathcal{O}(k^6)$, every connected component of the graph $\widetilde{G_Z} = G - (H \cup X \cup Z)$ is adjacent to at most 2 vertices of Z and there are $\mathcal{O}(k^6)$ connected components in the graph $\widetilde{G_Z}$ which are adjacent to exactly 2 vertices of Z. Moreover, we will be able to argue using flow arguments and the definition of rich/poor components that (i) no vertex of X has a neighbor in a connected component of $\widetilde{G_Z}$ which is adjacent to 2 vertices of Z and (ii) for every connected component D in the graph $\widetilde{G_Z}$ with at most 1 neighbor in Z and for any minimal feedback vertex set S of $G - H$ of size at most $2k$, S contains all but at most one vertex of $N(D) \setminus H$.

Step 3: We will finally augment the set H by adding some more vertices. Specifically, we will grow the set H by "buying a cheap set of vertices" as follows. As long as there is a vertex v which is adjacent to at least $\lceil \frac{2}{\delta} \rceil + 1$ connected components of $G[H]$ and contained in a connected component of $\widetilde{G_Z}$ adjacent to 2 vertices of Z, we set $H := H \cup \{v\}$. When this process terminates, it must be the case that every vertex which is in a connected component of $\widetilde{G_Z}$ adjacent to 2 vertices of Z, has at most $\lceil \frac{2}{\delta} \rceil$ neighboring components of $G[H]$. A simple counting argument based on the fact that we repeatedly decrease the number of connected components in $G[H]$ shows that the blow-up in the size of H is at most a $\delta/2$ fraction of the original value of $|H|$. ◀

We call a component D of $\widetilde{G_Z}$ a Type 0 component if $|N(D) \cap Z| = 0$, a Type 1 component if $|N(D) \cap Z| = 1$ and a Type 2 component if $|N(D) \cap Z| = 2$. Lemma 13 (5) implies that the Type 0 and Type 1 components (which comprise the set $\mathcal{C}$) only play the role of connectors and Lemma 13 (3) guarantees that the number of Type 2 components (which comprise the set $\mathcal{A}$) is $\mathcal{O}(k^6)$.

3.2 Marking Rules and Reduction Strategy via Steiner Trees

Once the partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ is computed, the second part of the PSAKS relies on an extension of the following result of Du et al. [10] to a special case of the Group Steiner Tree problem where at most one group can have size greater than 1.

▶ **Proposition 14** ([10]). *For every $p \geq 1$, graph G, $R \subseteq V(G)$, cost function $w : E(G) \to \mathbb{N} \cup \{0\}$ and R-Steiner tree T, there is a p-restricted R-Steiner tree in G of cost at most $(1 + \frac{1}{\lfloor \log_2 p \rfloor}) \cdot w(T)$.*

In the above proposition, an R-Steiner tree is a subtree of G containing R and a p-restricted R-Steiner tree is a connected subgraph of G containing R and whose edge set can be written as the union of the edge sets of Steiner trees for some subsets of R of size at most p where these subsets appear as the leaves of the respective Steiner trees In a similar spirit to Proposition 14, we show that for every $p \geq 1$, graph G, $R \subseteq V(G)$, cost function $w : E(G) \to \mathbb{N} \cup \{0\}$ and R-Steiner tree T intersecting a set $\mathcal{Z}$ of arbitrary size disjoint from R, there is a $2p + 1$-restricted R-Steiner tree in G of cost at most $(1 + \frac{1}{\lfloor \log_2 p \rfloor}) \cdot w(T)$ which also intersects $\mathcal{Z}$. We believe that this extension and close variants thereof are likely to have have future applications in dealing with connectivity constraints. We design a set of marking rules that achieve the following.

▶ **Lemma 15.** *There is an algorithm that, given G, k and the partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$, runs in time $k^{\mathcal{O}(\rho)} n^{\mathcal{O}(1)}$ and marks a set $\mathcal{A}' \subseteq \mathcal{A}$ of $k^{\mathcal{O}(\rho)}$ vertices such that for any S which is a connected feedback vertex set of G of size at most k, there is a connected feedback vertex set of G of size at most $(1 + \delta)^2 |S|$ whose intersection with $\mathcal{A}$ is contained in $\mathcal{A}'$.*

The main technical difficulty in the above lemma lies in identifying and marking vertices such that for any subset of $S \cap \mathcal{A}$ which may be performing the dual job of "hit" and "connect", the set $\mathcal{A}'$ of marked vertices contains a subset which would do the same job with only a small increase in size. Moreover, since we deal with each connected component of $G[\mathcal{A}]$ separately, we cannot simply use Proposition 14 or its extension we propose. This is where the upper bound on the degree of vertices of $\mathcal{A}$ into the set H will be crucial (Lemma 13 (4)).

Using Lemma 15, we will show that there is a way to reduce the graph $G[\mathcal{A}]$ by deleting or contracting all but $k^{\mathcal{O}(\rho)}$ of the rest of the edges so that the only unbounded set following this step is the set $\mathcal{C}$. As we know that the vertices in $\mathcal{C}$ only perform the job of "connectors" and are not necessary to hit cycles in G, we will mark the optimal Steiner tree (if it is small enough) in G for every choice of a sufficiently small subset of $\mathcal{A}' \cup \mathcal{B}$ as the set of terminals and use Proposition 14 to prove the following.

▶ **Lemma 16.** *There is an algorithm that, given G, k and the partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$, runs in time $k^{\mathcal{O}(\rho^2)} n^{\mathcal{O}(1)}$ and marks a set $\mathcal{C}' \subseteq \mathcal{C}$ of $k^{\mathcal{O}(\rho^2)}$ vertices such that for any S which is a connected feedback vertex set of G of size at most k, there is a connected feedback vertex set of G of size at most $(1 + \delta)^3 |S|$ whose intersection with $\mathcal{C}$ is contained in $\mathcal{C}'$.*

Finally, we show that we can delete the unmarked vertices in $\mathcal{C}$ after marking an additional $k^{\mathcal{O}(1)}$ new vertices and edges simply to remember the cycles passing through $\mathcal{C}$. Since we chose δ such that $(1 + \delta)^3 \leq 1 + \varepsilon$, we will have obtained the required graph. Specifically, we prove the following lemma.

▶ **Lemma 17.** *There is an algorithm that given G, k and the outputs of Lemma 13, Lemma 15 and Lemma 16 runs in time $k^{\mathcal{O}(\rho^2)} n^{\mathcal{O}(1)}$ and either correctly concludes that G has no connected feedback vertex set of size at most k or returns a graph G' such that:*

1. $|V(G')| = k^{\mathcal{O}(\rho^2)}$.
2. *Every minimal connected feedback vertex set of G' of size at most $(1+\varepsilon)k$ is contained in $V(G') \cap V(G)$ and is also a connected feedback vertex set of G.*
3. *For every S which is a minimal connected feedback vertex set of G of size at most k, G' has a connected feedback vertex set of size at most $(1+\varepsilon)|S|$.*

We summarize the steps of our algorithm below.

1. (Lemma 13) In polynomial time, identify a partition $(\mathcal{A}, \mathcal{B}, \mathcal{C})$ of $V(G)$ such that:
   - $|\mathcal{B}| = k^{\mathcal{O}(1)}$, $G - \mathcal{B}$ is acyclic, and $\mathcal{B}$ separates $\mathcal{A}$ and $\mathcal{C}$.
   - For every connected feedback vertex set S of G, $G - (S \setminus \mathcal{C})$ is acyclic.
   - Every connected component of $G[\mathcal{A}]$ has exactly 2 neighbors in $\mathcal{B}$ and there are $k^{\mathcal{O}(1)}$ connected components in $G[\mathcal{A}]$.
2. (Lemma 15 + Lemma 16) In time $k^{f(1/\varepsilon)}n^{\mathcal{O}(1)}$, mark sets $\mathcal{A}' \subseteq \mathcal{A}$ and $\mathcal{C}' \subseteq \mathcal{C}$ of $k^{\mathcal{O}(f(1/\varepsilon))}$ vertices such that for any S which is a connected feedback vertex set of G of size at most k, there is a connected feedback vertex set of G of size at most $(1+\varepsilon)|S|$ whose intersection with $\mathcal{A}$ ($\mathcal{C}$) is contained in $\mathcal{A}'$ ($\mathcal{C}'$).
3. (Lemma 17) In time $k^{f(1/\varepsilon)}n^{\mathcal{O}(1)}$, compute a graph G' with $k^{\mathcal{O}(f(1/\varepsilon))}$ vertices where:
   - Every minimal connected feedback vertex set of G' of size $\leq (1+\varepsilon)k$ is contained in $V(G') \cap V(G)$ and is also a connected feedback vertex set of G.
   - For every S which is a minimal connected feedback vertex set of G of size at most k, G' has a connected feedback vertex set of size at most $(1+\varepsilon)|S|$.

3.3 The PSAKS and Factor-OPT$^{\mathcal{O}(1)}$ Approximation

We are now ready to prove Theorem 1 by translating Lemma 17 into a PSAKS for Connected Feedback Vertex Set under the framework from [22]. Recall the definition of the parameterized optimization version of Connected Feedback Vertex Set has the optimization function $\text{CFVS} : \Sigma^* \times \mathbb{N} \times \Sigma^* \to \mathbb{R} \cup \{\infty\}$ defined as follows.

$$\text{CFVS}(G, k, S) = \begin{cases} \infty & \text{if } S \text{ is not a connected feedback vertex set of } G \\ \min\{|S|, k+1\} & \text{otherwise} \end{cases}$$

We define $\text{OPT}(G, k) = \min_{S \subseteq V(G)} \text{CFVS}(G, k, S)$. We point out that ∞ can be replaced in our setting with a sufficiently large value depending on $|V(G)|$ while serving the same purpose and avoiding potentially undefined operations.

▶ **Theorem 1.** *For every fixed $0 < \varepsilon < 1$,* Connected Feedback Vertex Set *has a $(1+\varepsilon)$-approximate kernelization of polynomial size.*

Proof. We begin by describing the reduction algorithm. If G is a forest, then return $(G', 0)$ where G' is the empty graph. Henceforth we ignore this corner case and assume that G contains a cycle. Given G, k and ε, we execute the algorithm of Lemma 17. If this algorithm concludes that G does not contain a connected feedback vertex set of size at most k, then we return the instance (G', k) where G' is a connected graph with $k+1$ vertex disjoint cycles and having $\mathcal{O}(k)$ vertices. Clearly such a graph exists. For instance, take the disjoint union of $k+1$ triangles and connect them in the form of a path by selecting exactly one vertex from each triangle. Otherwise, suppose that this algorithm returns a graph G'. Then the reduction algorithm returns the instance (G', k') where $k' = (1+\varepsilon)k$. At this point, we may assume that G is connected and so $V(G)$ is a connected feedback vertex set of G. If G' has multiple connected components containing cycles, then it cannot have a connected feedback

vertex set of *any* size, implying (by Lemma 17 (3)) that G cannot have a connected feedback vertex set of size at most k. Therefore we fall back into the previous case and so we may assume going forward that G' is also connected (if there are multiple components but only one contains cycles, then discard the rest) and so $V(G')$ is a connected feedback vertex set of G'. The polynomial bound on the size of the output instance and the time required to obtain G' follow from Lemma 17.

We now describe the solution lifting algorithm as follows. Let S' be the given solution for (G', k'). If S' is not a connected feedback vertex set of G', then the algorithm outputs $\emptyset$ as the solution for (G, k). If S' is a connected feedback vertex set of G' and $|S'| > k'$, then the algorithm outputs $V(G)$ as the solution for (G, k). Finally, if S' is a connected feedback vertex set of G' and $|S'| \leq k'$, then the algorithm returns S' as the solution for (G, k). We denote by S the output of the solution lifting algorithm. Clearly, the solution lifting algorithm runs in polynomial time.

We now prove that this reduction algorithm and the solution lifting algorithm together constitute a $(1+\varepsilon)$-approximate kernelization for Connected Feedback Vertex Set. We show that if $\mathrm{CFVS}(G', k', S') \leq c \cdot \mathrm{OPT}(G', k')$, then $\mathrm{CFVS}(G, k, S) \leq (1+\varepsilon)c \cdot \mathrm{OPT}(G, k)$. In the case where we concluded that G has no connected feedback vertex set of size at most k and returned (G', k), notice that $\mathrm{CFVS}(G', k, S') = \mathrm{CFVS}(G, k, S)$ and $\mathrm{OPT}(G, k) = \mathrm{OPT}(G', k) = k{+}1$. Hence, we are left with the following cases which arise when the invocation to Lemma 17 returned a graph G'. We will use the fact that $\mathrm{OPT}(G', k') \leq (1+\varepsilon)\mathrm{OPT}(G, k)$.

Case 1: *S' is a connected feedback vertex set of G' and $|S'| > k'$.* In this case, $S = V(G)$. We now consider two subcases: G has a connected feedback vertex set of size at most k or it does not. If G has no connected feedback vertex set of size at most k, it follows that $\mathrm{OPT}(G, k) = \mathrm{CFVS}(G, k, V(G)) = k + 1$. This is because G is connected and so $V(G)$ is a connected feedback vertex set. Therefore,

$$\frac{\mathrm{CFVS}(G, k, V(G))}{\mathrm{OPT}(G, k)} = 1 \leq (1+\varepsilon) \cdot \frac{\mathrm{CFVS}(G', k', S')}{\mathrm{OPT}(G', k')}. \tag{1}$$

Hence, we may assume that G has a connected feedback vertex set of size at most k. That is, $\mathrm{OPT}(G, k) \leq k$. But this implies the following.

$$\frac{\mathrm{CFVS}(G, k, V(G))}{\mathrm{OPT}(G, k)} = \frac{k+1}{\mathrm{OPT}(G, k)} \leq (1{+}\varepsilon) \cdot \frac{k'+1}{\mathrm{OPT}(G', k')} = (1{+}\varepsilon)\frac{\mathrm{CFVS}(G', k', S')}{\mathrm{OPT}(G', k')}. \tag{2}$$

Case 2: *S' is a connected feedback vertex set of G' and $|S'| \leq k'$.* In this case, $\mathrm{OPT}(G', k') \leq k'$. We consider two subcases: $|S'| \leq k$ or $k+1 \leq |S'| \leq k'$. In the former subcase we have the following.

$$\frac{\mathrm{CFVS}(G, k, S)}{\mathrm{OPT}(G, k)} = \frac{\mathrm{CFVS}(G, k, S')}{\mathrm{OPT}(G, k)} = \frac{\mathrm{CFVS}(G', k', S')}{\mathrm{OPT}(G, k)} \leq (1+\varepsilon)\frac{\mathrm{CFVS}(G', k', S')}{\mathrm{OPT}(G', k')}. \tag{3}$$

And if $k+1 \leq |S'| \leq k'$, then we have the following.

$$\begin{aligned}\frac{\mathrm{CFVS}(G, k, S)}{\mathrm{OPT}(G, k)} = \frac{\mathrm{CFVS}(G, k, S')}{\mathrm{OPT}(G, k)} = \frac{k+1}{\mathrm{OPT}(G, k)} &\leq (1+\varepsilon)\frac{k+1}{\mathrm{OPT}(G', k')} \\ &\leq (1+\varepsilon)\frac{\mathrm{CFVS}(G', k', S')}{\mathrm{OPT}(G', k')}.\end{aligned} \tag{4}$$

Case 3: *S' is not a connected feedback vertex set of G'.* Then, the set returned by the solution lifting algorithm, $\emptyset$, is also not a connected feedback vertex set of G and so, $\text{CFVS}(G', k', S') = \text{CFVS}(G, k, \emptyset)$. As a result, the required inequality follows and this completes the proof of the theorem. ◀

Lemma 17 also implies the following $\text{OPT}^{\mathcal{O}(1)}$-approximation for Connected Feedback Vertex Set.

▶ **Lemma 18.** *There is a polynomial time algorithm that given G, k, either correctly concludes that G has no connected feedback vertex set of size at most k or returns a connected feedback vertex set of G of size $k^{\mathcal{O}(1)}$.*

Proof. Given G, k, we set ε to be an arbitrary constant between 0 and 1, say $\frac{1}{2}$. We then invoke Lemma 17 to either correctly conclude that G has no connected feedback vertex set of size at most k or compute the graph G' guaranteed by the lemma. In the former case we return the same. Otherwise, we check for each connected component of $G[V(G') \cap V(G)]$ whether it is a connected feedback vertex set of G. If such a component exists, then we return the vertex set of this component and otherwise we conclude that G has no connected feedback vertex set of size at most k. The correctness follows from the fact that if G contains at least one connected feedback vertex set of size at most k then at least one connected component of $G[V(G') \cap V(G)]$ is guaranteed to contain a connected feedback vertex set of G according to Lemma 17 and such a connected component by itself must also be a connected feedback vertex set of G. This completes the proof of the lemma. ◀

4 Conclusions and Open Problems

Our result on approximate kernelization for Connected Feedback Vertex Set provides another useful data point in improving our understanding of the extent to which (approximate) preprocessing can be performed in the presence of connectivity constraints. Moreover, we believe that our techniques could have further applications in the design of approximate kernels for covering problems with connectivity constraints. Finally, this line of investigation offers several interesting opportunities for further research.

For instance, is there a *space efficient* PSAKS for Connected Feedback Vertex Set? In a space efficient PSAKS, we require the size of the output to be bounded by $f(\frac{1}{\varepsilon}) \cdot k^c$, where f is a computable function and c is a constant independent of the error parameter ε. Essentially, a PSAKS is an apt analogue of a PTAS in the approximate kernelization world and an Efficient PSAKS is a natural analogues of an Efficient PTAS in this setting. We end by pointing out that the existence of a space efficient PSAKS is open even in the case of the Connected Vertex Cover problem.

References

1 Vineet Bafna, Piotr Berman, and Toshihiro Fujito. A 2-Approximation Algorithm for the Undirected Feedback Vertex Set Problem. *SIAM J. Discrete Math.*, 12(3):289–297, 1999. doi:10.1137/S0895480196305124.

2 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.

3 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Cross-Composition: A New Technique for Kernelization Lower Bounds. In *28th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 165–176, 2011.

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

5 Holger Dell and Dániel Marx. Kernelization of packing problems. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 68–81, 2012.

6 Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 251–260, 2010.

7 Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization Lower Bounds Through Colors and IDs. *ACM Transactions on Algorithms*, 11(2):13:1–13:20, 2014.

8 Rodney G Downey and Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.

9 Andrew Drucker. New Limits to Classical and Quantum Instance Compression. *SIAM J. Comput.*, 44(5):1443–1479, 2015.

10 Ding-Zhu Du, Yanjun Zhang, and Qing Feng. On Better Heuristic for Euclidean Steiner Minimum Trees (Extended Abstract). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 431–439, 1991. doi:10.1109/SFCS.1991.185402.

11 Eduard Eiben, Danny Hermelin, and M. S. Ramanujan. Lossy Kernels for Hitting Subgraphs. In *42nd International Symposium on Mathematical Foundations of Computer Science, MFCS 2017, August 21-25, 2017 - Aalborg, Denmark*, pages 67:1–67:14, 2017. doi:10.4230/LIPIcs.MFCS.2017.67.

12 Eduard Eiben, Mithilesh Kumar, Amer E. Mouawad, Fahad Panolan, and Sebastian Siebertz. Lossy Kernels for Connected Dominating Set on Sparse Graphs. In *35th Symposium on Theoretical Aspects of Computer Science, STACS 2018, February 28 to March 3, 2018, Caen, France*, pages 29:1–29:15, 2018. doi:10.4230/LIPIcs.STACS.2018.29.

13 Bruno Escoffier, Laurent Gourvès, and Jérôme Monnot. Complexity and approximation results for the connected vertex cover problem in graphs and hypergraphs. *J. Discrete Algorithms*, 8(1):36–49, 2010. doi:10.1016/j.jda.2009.01.005.

14 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar $\mathcal{F}$-Deletion: Approximation, Kernelization and Optimal FPT Algorithms. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 470–479, 2012. doi:10.1109/FOCS.2012.62.

15 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.

16 Alexander Grigoriev and René Sitters. Connected Feedback Vertex Set in Planar Graphs. In *Graph-Theoretic Concepts in Computer Science, 35th International Workshop, WG 2009, Montpellier, France, June 24-26, 2009. Revised Papers*, pages 143–153, 2009. doi:10.1007/978-3-642-11409-0_13.

17 Danny Hermelin, Stefan Kratsch, Karolina Soltys, Magnus Wahlström, and Xi Wu. A Completeness Theory for Polynomial (Turing) Kernelization. *Algorithmica*, 71(3):702–730, 2015.

18 Danny Hermelin and Xi Wu. Weak compositions and their applications to polynomial lower bounds for kernelization. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 104–113, 2012.

19 Stefan Kratsch. Recent developments in kernelization: A survey. *Bulletin of the EATCS*, 113, 2014.

20 Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Kernelization–preprocessing with a guarantee. In *The Multivariate Algorithmic Revolution and Beyond*, pages 129–161. Springer, 2012.

21 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy Kernelization. *CoRR*, abs/1604.04111, 2016. arXiv:1604.04111.

22 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 224–237, 2017. doi:10.1145/3055399.3055456.

23 Neeldhara Misra, Geevarghese Philip, Venkatesh Raman, Saket Saurabh, and Somnath Sikdar. FPT algorithms for connected feedback vertex set. *J. Comb. Optim.*, 24(2):131–146, 2012. doi:10.1007/s10878-011-9394-2.

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

25 Mihalis Yannakakis. The Effect of a Connectivity Requirement on the Complexity of Maximum Subgraph Problems. *J. ACM*, 26(4):618–630, 1979. doi:10.1145/322154.322157.