



On Approximate Compressions for Connected Minor-Hitting Sets

M. S. Ramanujan   

University of Warwick, Coventry, UK

Abstract

In the CONNECTED \mathcal{F} -DELETION problem, \mathcal{F} is a fixed finite family of graphs and the objective is to compute a minimum set of vertices (or a vertex set of size at most k for some given k) such that (a) this set induces a connected subgraph of the given graph and (b) deleting this set results in a graph which excludes every $F \in \mathcal{F}$ as a minor. In the area of kernelization, this problem is well known to exclude a polynomial kernel subject to standard complexity hypotheses even in very special cases such as $\mathcal{F} = \{K_2\}$, i.e., CONNECTED VERTEX COVER.

In this work, we give a $(2 + \epsilon)$ -approximate polynomial compression for the CONNECTED \mathcal{F} -DELETION problem when \mathcal{F} contains at least one planar graph. This is the first approximate polynomial compression result for this generic problem. As a corollary, we obtain the first approximate polynomial compression result for the special case of CONNECTED η -TREEWIDTH DELETION.

2012 ACM Subject Classification Mathematics of computing \rightarrow Graph algorithms; Mathematics of computing \rightarrow Approximation algorithms

Keywords and phrases Parameterized Complexity, Kernelization, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.78

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 this area is that of a *kernelization* (also called a *kernel*), 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. Over the last decade, the area of kernelization has seen the development of a wide range of tools to design preprocessing algorithms and lower bounds techniques. The reader may find an introduction to the field in [24, 25, 3, 7].

An “efficient preprocessing algorithm” in this setting is referred to as a *polynomial kernel* and is simply a kernel whose output has size bounded polynomially in a parameter of the input. The central classification task in the area of kernelization is to identify NP-hard problems and associated parameters for which polynomial kernels exist and one of the main success stories in the area is the development of a rich theory of lower bounds based on complexity theoretic assumptions [1, 5, 2, 18, 4, 9, 21, 6, 20] allowing one to rule out the existence of polynomial kernels completely or lower bound the degree of the polynomial.

One fundamental class of problems for which polynomial kernels have been ruled out in this way is the class of subgraph hitting problems *with connectivity constraints*. It is well-known that placing connectivity constraints on subgraph hitting problems can have a dramatic effect on their amenability to efficient preprocessing. A case in point is the classic VERTEX COVER problem. This problem is known to admit a kernel whose output has $\mathcal{O}(k)$ vertices [3], where the parameter k is the solution size. However, the CONNECTED VERTEX COVER problem is amongst the earliest problems shown to exclude a polynomial kernel [6] and this lower bound immediately rules out the possibility of such a kernel for numerous well-studied generalizations of this problem. Consequently, one cannot hope to design approximation



© M. S. Ramanujan;
licensed under Creative Commons License CC-BY 4.0
29th Annual European Symposium on Algorithms (ESA 2021).

Editors: Petra Mutzel, Rasmus Pagh, and Grzegorz Herman; Article No. 78; pp. 78:1–78:16



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

algorithms for such problems via polynomial kernels in the traditional sense and so, 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 in general and in the design of approximation algorithms for connectivity constrained subgraph hitting problems.

One of the most frequently investigated subgraph hitting problems in the literature is the \mathcal{F} -DELETION problem which generalizes numerous well-studied NP-complete problems. In this problem, \mathcal{F} is a fixed finite family of graphs and one is given a graph G and an integer k as input. The objective is to determine whether at most k vertices can be deleted from G so that the resulting graph is \mathcal{F} -minor free (does not contain a minor isomorphic to a graph in \mathcal{F}). The optimization version of this problem asks for a minimum set of vertices whose deletion leaves a graph which is \mathcal{F} -minor free. Well-studied special cases of this problem include VERTEX COVER ($\mathcal{F} = \{K_2\}$), FEEDBACK VERTEX SET ($\mathcal{F} = \{K_3\}$), PLANARIZATION ($\mathcal{F} = \{K_{3,3}, K_5\}$) [27], DIAMOND HITTING SET ($\mathcal{F} = \{\theta_3\}$) [14], PATHWIDTH ONE VERTEX DELETION ($\mathcal{F} = \{K_3, T_2\}$) [29], and θ_c -DELETION [22, 15]. A common feature shared by many such well explored special cases of this problem is that \mathcal{F} contains at least one planar graph. Motivated by this, Fomin et al. [17] investigated this restricted variant of the problem (when the family \mathcal{F} contains at least one planar graph) and obtained a polynomial kernel for every such \mathcal{F} . This particular variant of \mathcal{F} -DELETION is known in the literature as the PLANAR \mathcal{F} -DELETION problem.

Motivated by the prevalence of the PLANAR \mathcal{F} -DELETION problem in existing work on subgraph hitting problems, we initiate the study of the PLANAR \mathcal{F} -DELETION problem when there is a connectivity constraint on the solution. This problem, which we call the CONNECTED PLANAR \mathcal{F} -DELETION problem, is formally defined as follows. The input is a graph G , and integer k (the parameter) and the goal is to determine whether there is a set $S \subseteq V(G)$ of size at most k such that $G[S]$ is connected and $G - S$ is \mathcal{F} -minor free? The set S is called a connected \mathcal{F} -deletion set.

As already discussed, CONNECTED PLANAR \mathcal{F} -DELETION displays stark differences to the version without connectivity constraints when considering the approximability as well as amenability to efficient preprocessing even when \mathcal{F} is a very simple family such as $\{K_2\}$. To be specific, since this problem is a clear generalization of CONNECTED VERTEX COVER, it cannot have a $(2 - \varepsilon)$ -approximation in polynomial time for any fixed $\varepsilon > 0$ under UGC [23] and moreover, it is unlikely to have a polynomial kernel [6].

Since using the existing notion of polynomial kernels and associated reduction rules in order to design approximation algorithms for such connectivity constrained problems appears to be difficult, we rely on the recently developed notion of α -approximate kernels which was introduced by Lokshtanov et al. [26] in order to facilitate the rigorous analysis of preprocessing algorithms in conjunction with approximation algorithms.

Informally speaking, an α -approximate kernel is a polynomial-time algorithm that given as input a pair (I, k) where I is the problem instance and k is the parameter, outputs an instance (I', k') of the same problem such that $|I'| + k' \leq g(k)$ for some computable function g and any c -approximate solution for the instance I' can be turned in polynomial time into a $(c \cdot \alpha)$ -approximate solution for the original instance I . When the output is an instance of a different problem (with the remaining conditions holding), one obtains an α -approximate compression.

As earlier, the notion of efficiency in this context is captured by the function g being required to be 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, Lokshitanov et al. considered several problems which are known to exclude polynomial kernels and presented an α -approximate polynomial kernel 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 CONNECTED VERTEX COVER admits an α -approximate polynomial kernel for every $\alpha > 1$. Their result provided a promising starting point towards a refined understanding of the role played by connectivity constraints in relation to preprocessing for covering problems on graphs. Subsequently, Eiben et al. [11] extended this result to the CONNECTED \mathcal{H} -HITTING SET problem where \mathcal{H} is a fixed collection of finite subgraphs and the solution is a minimum set of vertices in the given graph G which induce a connected subgraph and hit all copies of graphs in \mathcal{H} which are present in G . Recently, Ramanujan [30] obtained the first α -approximate polynomial kernel (for every $\alpha > 1$) for the CONNECTED FEEDBACK VERTEX SET problem, demonstrating the power of approximate preprocessing for cases where the goal is to hit obstructions of unbounded size while maintaining connectivity of the hitting set.

Our results

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

► **Theorem 1.** *For every fixed $0 < \varepsilon < 1$, CONNECTED PLANAR \mathcal{F} -DELETION has a $(2 + \varepsilon)$ -approximate compression of polynomial size.*

As an immediate corollary of Theorem 1, we obtain a factor- $(2 + \varepsilon)$ parameterized approximation algorithm for CONNECTED PLANAR \mathcal{F} -DELETION running in time $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ for every fixed $0 < \varepsilon < 1$ ¹. In fact, the proof techniques we use in order to prove Theorem 1 also enable us to obtain a polynomial-time $\text{poly}(\text{OPT})$ approximation for this problem.

► **Theorem 2.** *For every fixed \mathcal{F} containing a planar graph, 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 \mathcal{F} -deletion set of size at most k or returns a connected \mathcal{F} -deletion set of G of size $k^{\mathcal{O}(1)}$.*

Using Theorem 2 and adopting an approach similar to that in [30], we obtain the following.

► **Theorem 3.** *There is a $0 < \delta < 1$ such that CONNECTED PLANAR \mathcal{F} -DELETION can be approximated within a factor $\min\{\text{OPT}^{\mathcal{O}(1)}, n^{1-\delta}\}$ in polynomial time.*

Related work on approximation for connected hitting set problems

Grigoriev and Sitters [19] 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. Fiorini et al. [14] studied the DIAMOND HITTING SET problem (where $\mathcal{F} = \{\theta_3\}$) and obtained the first constant-factor approximation.

¹ We note that this problem can be easily seen to be fixed-parameter tractable parameterized by k since the problem is MSO-expressible and the treewidth of yes-instances must be bounded by $\mathcal{O}(k)$, allowing for an invocation of Courcelle's theorem.

2 Preliminaries

For a graph G , we denote by $\mathcal{CC}(G)$ the set of connected components of G . 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 . For a set $X \subseteq V(G)$, the graph obtained from G by *identifying* the vertices in X is the graph G' defined as follows. The vertex set of G' is $(V(G) \setminus X) \cup \{x\}$ where $x \notin V(G)$. For every edge of G which is not incident on X , G' has the same edge. For every edge $(u, v) \in E(G)$ where $u \in X, v \notin X$, we add a new edge (x, v) . Note that we ignore all edges which have both endpoints in X .

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 (x, k) , where k is called the parameter. We assume that k is *given in unary* and hence $k \leq |x|$. The notion of *kernelization* is formally defined as follows.

► **Definition 4** (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 5** ([26]). *A parameterized optimization (minimization or maximization) problem is a computable function $\Pi : \Sigma^* \times \mathbb{N} \times \Sigma^* \rightarrow \mathbb{R} \cup \{\pm\infty\}$.*

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 the problems we come across in this paper are minimization problems, we state some of the definitions only in terms of minimization problems when the definition for maximization problems is analogous. For instance, the parameterized optimization version of CONNECTED PLANAR \mathcal{F} -DELETION is defined as follows (using the convention from [26]). This is a minimization problem with the optimization function $\text{CPFD} : \Sigma^* \times \mathbb{N} \times \Sigma^* \rightarrow \mathbb{R} \cup \{\infty\}$ defined as follows.

$$\text{CPFD}(G, k, S) = \begin{cases} \infty & \text{if } S \text{ is not a feasible solution,} \\ \min\{|S|, k + 1\} & \text{otherwise.} \end{cases}$$

► **Definition 6** ([26]). *For a parameterized minimization problem Π , the optimum value of an instance $(I, k) \in \Sigma^* \times \mathbb{N}$ is $\text{OPT}_\Pi(I, k) = \min_{\substack{s \in \Sigma^* \\ |s| \leq |I| + k}} \Pi(I, k, s)$.*

We now recall the other relevant definitions from [26] regarding *approximate kernels*.

► **Definition 7** ([26]). *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 $\mathcal{R}_\mathcal{A} : \Sigma^* \times \mathbb{N} \rightarrow \Sigma^* \times \mathbb{N}$. 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 $\frac{\Pi(I, k, s)}{\text{OPT}(I, k)} \leq \alpha \cdot \frac{\Pi(I', k', s')}{\text{OPT}(I', k')}$.

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

► **Definition 8** ([26]). Let $\alpha \geq 1$ be a real number. Let Π and Π' be two parameterized minimization problems. An α -approximate polynomial parameter transformation (α -appt for short) \mathcal{A} from Π to Π' is a pair of polynomial-time algorithms, called reduction algorithm $\mathcal{R}_{\mathcal{A}}$ and solution-lifting algorithm. Given as input an instance (I, k) of Π the reduction algorithm outputs an instance (I', k') of Π' such that $k' = k^{O(1)}$. The solution-lifting algorithm takes as input an instance (I, k) of Π , the output instance $(I', k') = \mathcal{R}_{\mathcal{A}}(I, k)$ of Π' , and a solution s' to the instance I' and outputs a solution s to (I, k) such that

$$\frac{\Pi(I, k, s)}{\text{OPT}_{\Pi}(I, k)} \leq \alpha \cdot \frac{\Pi'(I', k', s')}{\text{OPT}_{\Pi'}(I', k')}.$$

► **Definition 9** ([26], α -approximate compression). Let $\alpha \geq 1$ be a real number. Let Π and Π' be two parameterized minimization problems. An α -approximate compression from Π to Π' is an α -appt \mathcal{A} from Π to Π' such that $\text{size}_{\mathcal{A}}(k) = \sup\{|I'| + k' : (I', k') = \mathcal{R}_{\mathcal{A}}(I, k), I \in \Sigma^*\}$, is upper bounded by a computable function $g : \mathbb{N} \rightarrow \mathbb{N}$, where $\mathcal{R}_{\mathcal{A}}$ is the reduction algorithm in \mathcal{A} . We say that \mathcal{A} is an α -approximate polynomial compression if g is a polynomial function.

Treewidth, t -Boundaried graphs and minors

We now recall standard definitions regarding treewidth and minor models. The notation is based on [17]. Let G be a graph. A *tree decomposition* of G is a pair $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ where T is a tree and \mathcal{X} is a collection of subsets of $V(G)$ such that (a) $\forall e = uv \in E(G), \exists t \in V(T) : \{u, v\} \subseteq X_t$ and (b) $\forall v \in V(G), T[\{t \mid v \in X_t\}]$ is a non-empty connected subtree of T . We call the vertices of T *nodes* and the sets in \mathcal{X} *bags* of the tree decomposition (T, \mathcal{X}) . The *width* of (T, \mathcal{X}) is denoted by $\text{width}(T, \mathcal{X})$ is defined as $\max\{|X_t| - 1 \mid t \in V(T)\}$ and the *treewidth* of G is the minimum width over all tree decompositions of G . A t -boundaried graph is a graph G and a set $B \subset V(G)$ of size at most t with each vertex $v \in B$ having a label $\ell_G(v) \in \{1, \dots, t\}$. Each vertex in B has a unique label. We refer to B as the *boundary* of G . We use the notation (G, B) to refer to the t -boundaried graph G with boundary B .

Least Common Ancestor-Closure of Sets in Graphs of Bounded Treewidth

For a graph G with a nice tree decomposition (T, χ) rooted at $r \in V(T)$ and vertex set $X \subseteq V(G)$ the least common ancestor-closure $\text{LCA-closure}(T, \chi, X)$ is defined as follows. We let $M \subseteq V(T)$ denote a minimal set of nodes in T such that for every $x \in X$, there is a bag $u \in M$ such that $x \in \chi(u)$ and moreover, u is the closest such vertex to r . Finally, we define $\text{LCA-closure}(T, \chi, X)$ as the set $\chi(\text{LCA-closure}(M))$, where $\text{LCA-closure}(M)$ denotes the LCA-closure of M in the rooted tree T . We ignore the explicit reference to the root r in the notation $\text{LCA-closure}(T, \chi, X)$ because we will be using an arbitrary vertex as the root when invoking this definition.

The following lemma is a direct consequence of the definition of $\text{LCA-closure}(T, \chi, X)$ and the application of LCA-closure on trees (see, for example, [17]).

► **Lemma 10.** *Let G be a graph with a nice tree decomposition (T, χ) and let $X \subseteq V(G)$. Let X' denote the set $\text{LCA-closure}(T, \chi, X)$. Then, $|X'| \leq 2|X| \cdot \text{width}(T, \chi)$ and for every connected component C of $G - X'$, $|N(C)| \leq 2 \cdot \text{width}(T, \chi)$.*

► **Definition 11.** *Let G be a graph and H be a minor of G with $V(H) = \{h_1, \dots, h_\ell\}$ and suppose that H has no self-loops. A set $\mathcal{P}_H = \{P_1, \dots, P_\ell\}$ of pairwise vertex-disjoint subsets of $V(G)$ is said to be a minor model of H in G if*

- $G[P_i]$ is connected for every $i \in [\ell]$ and
- there is an injective mapping $\phi : E(H) \rightarrow E(G)$ such that for every $e = (i, j) \in E(H)$, $\phi(e)$ is an edge in G with one endpoint in P_i and one in P_j .

Note that if H has parallel edges, then the minor model needs to have a unique edge corresponding to each parallel edge. We say that H is a minimal minor model if there is no strict subgraph of $G[\bigcup_{i \in [\ell]} P_i]$ which also contains H as a minor.

► **Definition 12.** *Let G_1 and G_2 be two graphs, and let t be a fixed positive integer. For $i \in \{1, 2\}$, let f_{G_i} be a function that associates with every vertex of $V(G_i)$ some subset of $[t]$. The image of a vertex $v \in G_i$ under f_{G_i} is called the label of that vertex. We say that G_1 is label-wise isomorphic to G_2 , and denote it by $G_1 \cong_t G_2$, if there is a map $h : V(G_1) \rightarrow V(G_2)$ such that (a) h is one to one and onto; (b) $(u, v) \in E(G_1)$ if and only if $(h(u), h(v)) \in E(G_2)$ and (c) $f_{G_1}(v) = f_{G_2}(h(v))$. We call h a label-preserving isomorphism.*

Notice that the first two conditions of Definition 12 simply indicate that G_1 and G_2 are isomorphic. Now, let G be a t -boundaried graph, that is, G has t distinguished vertices, uniquely labeled from 1 to t . Given a t -boundaried graph G , we define a canonical labeling function $\mu_G : V(G) \rightarrow 2^{[t]}$. The function μ_G maps every distinguished vertex v with label $\ell \in [t]$ to the set $\{\ell\}$, that is $\mu_G(v) = \{\ell\}$, and for all other vertices we have that $\mu_G(v) = \emptyset$.

Next we define a notion of labeled edge contraction. Let H be a graph together with a function $f_H : V(H) \rightarrow 2^{[t]}$ and $(u, v) \in E(H)$. Furthermore, let H' be the graph obtained from H by identifying the vertices u and v into w_{uv} and removing all loops. Then by *labeled edge contraction* of an edge (u, v) of a graph H , we mean obtaining a graph H' with the label function $f_{H'} : V(H') \rightarrow 2^{[t]}$ defined as follows. For $x \in V(H') \cap V(H)$ we have that $f_{H'}(x) = f_H(x)$ and for w_{uv} we define $f_{H'}(w_{uv}) = f_H(u) \cup f_H(v)$. Now we introduce a notion of labeled minors of a t -boundaried graph.

► **Definition 13.** *Let H be a graph together with a function $f : V(H) \rightarrow 2^{[t]}$ and (G, Z) be a t -boundaried graph with canonical labeling function μ_G . A graph H is called a labeled minor of G , if we can obtain a labeled isomorphic copy of H from G by performing edge deletions and labeled edge contractions. The h -folio of a t -boundaried graph (G, Z) is the set $\mathcal{M}_h(G, Z)$ of all labeled minors of G (starting with the canonical labeling on G) on at most h vertices.*

We also need the following well-known result bounding the treewidth of any graph which excludes a fixed planar graph as a minor.

► **Proposition 14** ([17]). *For every fixed planar graph H , there is a constant λ_H such that any graph G with $\text{tw}(G) \geq \lambda_H$ contains H as a minor.*

Steiner Trees

For a graph G , a set $R \subseteq V(G)$ called *terminals* and a cost function $w : E(G) \rightarrow \mathbb{N} \cup \{0\}$, a *Steiner tree* is a subtree T of G such that $R \subseteq V(T)$, and the *cost* of a tree T is defined as $w(T) = \sum_{e \in E(T)} w(e)$. In the STEINER TREE problem we are given as input the graph G ,

the set R and the cost function $w : E(G) \rightarrow \mathbb{N} \cup \{0\}$. The task is to find an *optimal Steiner tree*, which is a Steiner tree of minimum cost. However, in this paper we will only work with edges of unit or zero cost and we denote by w_1 the function that assigns a cost of 1 to every edge of the graph under consideration. When T is a Steiner tree for the set of terminals R , we say that T is an *R -Steiner tree*. An R -Steiner tree T is said to be *minimal* if there is no $e \in E(T)$ such that $T - e$ also contains an R -Steiner tree. It is well-known that there is an algorithm for STEINER TREE with a single exponential dependence on the number of terminals.

► **Proposition 15** ([8, 28]). *The STEINER TREE problem can be solved in time $2^{\mathcal{O}(|R|)} n^{\mathcal{O}(1)}$.*

Let R continue to denote the set of terminals. For a $k \in \mathbb{N}$, a *k -component* is a tree with at most k leaves, all of which are in R . A *k -restricted Steiner tree* \mathcal{T} is a collection of k -components, such that the union of these components is a Steiner tree T . The cost of \mathcal{T} is the sum of the costs of all the k -components in \mathcal{T} . It may be the case that multiple k -components which are part of a k -restricted Steiner tree, share edges. As a result, some edges may contribute multiple times to the cost of \mathcal{T} . To keep the presentation simple, if the k -components of a k -restricted Steiner tree are clear from the context, then we also refer to the Steiner tree composed of their union as a k -restricted Steiner tree. Recall that according to its original definition, a k -restricted Steiner tree is a *set* of k -components.

3 Overview of our Algorithms

Fix $0 < \varepsilon < 1$. We identify a partition (A, B, C) of the vertex set of G , where $|B| = k^{\mathcal{O}(f(1/\varepsilon))}$ and there are no edges between A and C . In other words, B separates A and C . We then prove that the vertices in C only play the role of “connectors” and removing them from a hypothetical solution S may disconnect $G[S]$, but will still leave a minimal hitting set for all \mathcal{F} -minor models. On the other hand, while the interaction of vertices in A with the solution S could be much more complex, we show that the number of connected components of $G[A]$ is $k^{\mathcal{O}(f(1/\varepsilon))}$ and these can be shown to have a well-structured neighborhood in B . Once we have this partition in hand, we focus on each connected component of $G[A]$ separately and from each component we identify a set of $k^{\mathcal{O}(f(1/\varepsilon))}$ vertices which, together with B and a $k^{\mathcal{O}(f(1/\varepsilon))}$ sized subset of C cover a $(2 + \varepsilon)$ -approximate solution. Finally, the remaining vertices are discarded in an appropriate manner once the relevant information they hold is compiled into a polynomially bounded data structure. This high level approach of identifying “hitters” and “connectors” among the vertices is a natural first step for problems with this flavor [11, 12, 30] and the more involved problem-specific part in each case resides in (a) computing such a partition and (b) setting up a procedure to identify and remove redundant parts of the input, leaving only $k^{\mathcal{O}(f(1/\varepsilon))}$ vertices or edges.

Fix $\rho = 2^{\mathcal{O}(1/\varepsilon)}$ to be a constant such that $\frac{1}{\lceil \log_2 \rho \rceil} \leq \varepsilon$. Our starting point is a lemma of Fomin et al. [17] showing that there is a polynomial-time algorithm that takes as input the pair (G, k) and either correctly concludes that G has no \mathcal{F} -deletion set (a set of vertices whose deletion leaves an \mathcal{F} -minor free graph) of size at most k or outputs disjoint sets $X, Z \subseteq V(G)$ such that (a) $|X \cup Z| = k^{\mathcal{O}(1)}$, (b) X is an \mathcal{F} -deletion set of G , (c) for every connected component C of $G - (X \cup Z)$, $|N(C) \cap Z| \leq 2(\eta + 1)$, and (d) for every $x, y \in X$, either Z intersects every x, y path in $G - (X \setminus \{x, y\})$ or there is an x - y flow of value at least $3k + \eta + 3$ in the graph $G - (X \setminus \{x, y\})$. Let G_τ denote the graph $G - (X \cup Z)$. In the proposed partition (A, B, C) , our intention is to set $B = X \cup Z$.

Given the sets X and Z described above we will compute a subset of vertices covering a $(2 + \varepsilon)$ -approximate connected \mathcal{F} -deletion set (if one of size at most k exists) in two stages. In the first stage, we mark a set of $k^{\mathcal{O}(\rho)}$ vertices of G_τ such that for every subset U of $X \cup Z$, if

there is a set T_U of vertices in G_τ using which the vertices in U can be connected, then there is a set of roughly $(1 + \varepsilon)|U|$ *marked* vertices which can do the same connecting task as T_U with respect to the vertices in U . In the second stage, we ignore the connectivity requirement on the subset of the solution in $X \cup Z$ and for each connected component of G_τ , mark $k^{\mathcal{O}(\rho)}$ vertices such that if there is a solution which intersects a component of G_τ then there is a way to select a sufficiently small subset of the *marked* vertices in the component which can be connected to the vertices of the solution in $X \cup Z$ and achieve the same “ \mathcal{F} -minor hitting” behaviour as the original set of vertices.

A major obstacle here is the fact that the number of connected components of G_τ may be unbounded, which might mean that we mark a set of vertices whose size is not bounded by a function of the parameter k at all. It is to overcome this obstacle that we distribute the set $V(G) \setminus B$ among the remaining partitions A and C . In other words, we will be able to partition the components of G_τ into two sets that we call **Type 1** components (corresponding to C) and **Type 2** components (corresponding to A) and then show that the **Type 1** components although unbounded in number, only provide connectivity to a minimal \mathcal{F} -deletion set contained in the solution while, on the other hand, the **Type 2** components are more complex but bounded polynomially (in k) in number. We then argue that there is a way to achieve the objective of our initial marking strategy by marking only a *polynomial* number of vertices of G_τ in total, and these vertices cover a factor- $(2 + \varepsilon)$ approximate solution and moreover, a factor- $\text{OPT}^{\mathcal{O}(1)}$ approximate solution can be recovered by a straightforward examination of the connected components of the subgraph induced by the marked vertices.

To extend this to our approximate compression, we prove the following lemma.

► **Lemma 16.** *There is an algorithm that given G, k, X, Z as described above, runs in time $k^{\mathcal{O}(\rho)}n^{\mathcal{O}(1)}$ and either correctly concludes that G has no connected \mathcal{F} -deletion set of size at most k or returns a set $L \subseteq V(G)$ such that the following statements hold.*

1. $L \supseteq (X \cup Z)$, $|L| = k^{\mathcal{O}(\rho)}$.
2. For every S which is a minimal connected \mathcal{F} -deletion set of G of size at most k , G has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ contained in L .
3. For every connected component C of $G - L$, $|N(C) \cap (L \setminus X)| \leq 2(\eta + 1)$.
4. For every \mathcal{F} -deletion set S of G of size at most $3k$ and every $C \in \mathcal{CC}(G - L)$, $|(N(C) \cap X) \setminus S| \leq \eta + 1$.

The above lemma is obtained following the previously discussed marking steps that led to the factor- $\text{OPT}^{\mathcal{O}(1)}$ approximation and can be seen as an “approximate-solution-capturing” variant of the lemma of Fomin et al. [17] in the context of our problem. Once we have this lemma in hand, we define an annotated version of **CONNECTED PLANAR \mathcal{F} -DELETION** and encode the output of the lemma above as an instance of **ANNOTATED CONNECTED PLANAR \mathcal{F} -DELETION** whose size is bounded polynomially in k , which completes the compression, giving us Theorem 1.

4 The $(2 + \varepsilon)$ -Approximate Compression for **CONNECTED PLANAR \mathcal{F} -DELETION**

Recall that we have chosen $\rho = 2^{\mathcal{O}(1/\varepsilon)}$ to be a constant such that $\frac{1}{\lfloor \log_2 \rho \rfloor} \leq \varepsilon$. Throughout this section, we suppress the dependence of the running time and compression size on \mathcal{F} in the $\mathcal{O}(\cdot)$ notation. However, we make the dependencies on ε explicit. Our first aim in this section is to prove Lemma 17, which is then invoked in the proof of Lemma 16. This lemma says that there is a polynomial-time algorithm that identifies a set of $k^{\mathcal{O}(\rho)}$ vertices that cover a $(2 + \varepsilon)$ -approximate solution (if one of size at most k exists).

► **Lemma 17.** *There is an algorithm that, given G and k , runs in time $k^{\mathcal{O}(\rho)}n^{\mathcal{O}(1)}$ and either correctly concludes that G has no connected \mathcal{F} -deletion set of size at most k or returns a set $V^\infty \subseteq V(G)$ such that the following statements hold.*

1. $|V(G) \setminus V^\infty| = k^{\mathcal{O}(\rho)}$.
2. For every S which is a minimal connected \mathcal{F} -deletion set of G of size at most k , G has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ that is disjoint from V^∞ .

Let η be a constant depending only on \mathcal{F} such that every graph which is \mathcal{F} -minor free has treewidth at most η . By Proposition 14, we know that such a constant exists. Let $h = 10\eta + \max_{F \in \mathcal{F}} |F|$. Note that h is an upper bound on the number of vertices and the number of edges in any graph in \mathcal{F} .

We begin building towards the proof of Lemma 17 by recalling the following lemma from [17] and some associated observations. The numbering of the lemmas in our citations is derived from the full version of [16]. A set $S \subseteq V(G)$ is called an \mathcal{F} -deletion set of G if $G - S$ is an \mathcal{F} -minor free graph.

► **Lemma 18** (Lemma 25, [16]). *There is a randomized polynomial-time algorithm that given (G, k) , either concludes that G has no \mathcal{F} -deletion set of size at most k or outputs disjoint sets $X, Z \subseteq V(G)$ satisfying the following properties.*

1. $|X \cup Z| = k^{\mathcal{O}(1)}$.
2. X is an \mathcal{F} -deletion set of G .
3. For every connected component C of $G - (X \cup Z)$, $|N(C) \cap Z| \leq 2(\eta + 1)$.
4. For every $x, y \in X$, either Z intersects every x - y path in $G - (X \setminus \{x, y\})$ or there is an x - y flow of value at least $3k + \eta + 3$ in the graph $G - (X \setminus \{x, y\})$.

If G has an \mathcal{F} -DELETION set of size k , then this algorithm outputs the pair (X, Z) with probability $(1 - \frac{1}{2^n})$. Otherwise, the algorithm always correctly concludes that no such set exists.

When the pair $\tau = (X, Z)$ is clear from the context, we use G_τ to refer to the graph $G - (X \cup Z)$.

The randomization in Fomin et al.'s proof of this lemma arises due to the execution of a randomized linear-time constant-factor approximation algorithm for (unconnected) PLANAR \mathcal{F} -DELETION, i.e., the subroutine used to compute the set X . However, this step can be replaced with any deterministic factor-OPT ^{$\mathcal{O}(1)$} approximation for the same problem with, again, only a constant-factor change in the degree of the polynomial in the first property of the lemma. In particular, we can use the factor- $O(\log^{3/2} \text{OPT})$ approximation from [15] and avoid the randomization.

We now recall the following useful properties of non-solution vertices in the neighborhood of any connected component of G_τ .

► **Proposition 19** (Lemma 26, [16]). *Let P be a set of vertices such that for every distinct $u, v \in P$, there is a u - v flow of value $3k + \eta + 3$, let S be an \mathcal{F} -deletion set of G of size at most $3k$ which is disjoint from P and let (T, χ) be a tree decomposition of $G - S$ of width at most η . Then, for every pair of vertices $u, v \in P$, there is a vertex $x \in V(T)$ such that $u, v \in \chi(x)$ and moreover, there is a vertex $y \in V(T)$ such that $P \subseteq \chi(y)$.*

The above proposition captures the fact that if P is disjoint from S and is a set of vertices with pairwise-high flow, then these vertices must appear together in some bag of the tree decomposition (T, χ) . Indeed, if this were not the case, then a pair of vertices in this set can be separated by deleting a small separator in the graph, contradicting the high flow between them. The next statement follows as a consequence of Proposition 19.

► **Proposition 20** (Lemma 26, [16]). *Let G, k and $\tau = (X, Z)$ be as in Lemma 18. For every \mathcal{F} -deletion set S of G of size at most $3k$ and every connected component C of G_τ , $|N(C) \cap X \setminus S| \leq \eta + 1$.*

Indeed, the fourth property ensured by Lemma 18 guarantees that for every connected component C of G_τ , $N(C) \cap X$ is a set of vertices with a pairwise flow of value at least $3k + \eta + 3$ in G . If it were the case that $|N(C) \cap X \setminus S| > \eta + 1$, then Proposition 19 would imply that some bag of a tree decomposition of $G - S$ of width η contains the set $(N(C) \cap X) \setminus S$, which has size greater than $\eta + 1$, a contradiction.

Proposition 20 guarantees that for every connected component C of G_τ , all but at most $\eta + 1$ vertices in $N(C) \cap X$ must be deleted (equivalently, contained in S). Combining this with the third property ensured by Lemma 18, we may conclude that all but at most $3(\eta + 1)$ vertices in $N(C)$ must be contained in S .

Our marking rules rely on the following result of Du et al. [10] which has found applications in other algorithms for connected deletion problems [11, 30].

► **Proposition 21** ([10]). *For every $p \geq 1$, graph G , $R \subseteq V(G)$, cost function $w : E(G) \rightarrow \mathbb{N} \cup \{0\}$ and R -Steiner tree T , there is a p -restricted R -Steiner tree in G of cost at most $(1 + \frac{1}{\lfloor \log_2 p \rfloor}) \cdot w(T)$.*

4.1 The $\text{OPT}^{\mathcal{O}(1)}$ -approximation for CONNECTED PLANAR \mathcal{F} -DELETION

In what follows, we fix G, k , and $\tau^* = (X, Z)$ as given by Lemma 18.

► **Definition 22.** *For a vertex set C disjoint from X , a set $J \subseteq N(C)$ is called a set compatible with C if $|J \cap X| \leq \eta + 1$. We denote by \mathcal{B}_C the set of all sets compatible with C . For a set C and set J compatible with C , the $|J|$ -boundaried graph $(G[C \cup J], J)$ is denoted by $G_{C,J}$.*

Recall that $\forall C \in \mathcal{CC}(G_{\tau^*})$, $|N(C) \cap Z| \leq 2(\eta + 1)$ (argued using the third property of Lemma 18). Consequently, for every $C \in \mathcal{CC}(G_{\tau^*})$, any set $J \subseteq N(C)$ compatible with C has size at most $3\eta + 3$, which is a constant in our setting. Moreover, due to Proposition 20, we have that for any solution S , the set $N(C) \setminus S$ must be a set compatible with C .

► **Lemma 23.** *There is an algorithm that given G, k , and $\tau^* = (X, Z)$, runs in polynomial time and returns a set $P^\infty \subseteq V(G) \setminus (X \cup Z)$ and a partition of $\mathcal{CC}(G_{\tau^*})$ into sets \mathcal{P} and \mathcal{Q} such that the following statements hold.*

1. $|\mathcal{P}| = k^{\mathcal{O}(1)}$ and $P^\infty \subseteq V(\mathcal{P})$.
2. Every \mathcal{F} -deletion set of $G - V(\mathcal{Q}) - P^\infty$ of size at most $3k$ is an \mathcal{F} -deletion set of G .

► **Definition 24.** *Let $G, k, \tau^* = (X, Z), \mathcal{P}, \mathcal{Q}$ be as in Lemma 23. We call every component in \mathcal{Q} a Type 1 component of G_τ and every component in \mathcal{P} a Type 2 component of G_{τ^*} .*

Lemma 23 implies that the vertices contained in Type 1 components are only required for providing connectivity between those vertices of a minimal \mathcal{F} -deletion set of size at most $3k$ that are contained in $X \cup Z$ and the Type 2 components are bounded polynomially in k . The proof of this lemma closely follows the proof of Fomin et al. (see Lemma 36, [16]) with the following difference: instead of discarding irrelevant vertices, we identify them and use them to define the sets \mathcal{P} and \mathcal{Q} .

We are now ready to state our marking rules. We fix $G, k, \tau^* = (X, Z), P^\infty, \mathcal{P}, \mathcal{Q}$ as given by Lemma 23.

► **Marking Rule 1.** For every $R \subseteq X \cup Z$ of size at most ρ , we compute an optimal R -Steiner tree T_R in G with the unitary cost function (denoted w_1). If $|V(T_R)| \leq (2 + \varepsilon)k$, then we mark the vertices of T_R .

Before we describe our next marking rule, we need the following definition.

► **Definition 25.** For a graph G and vertex $p \in V(G)$, we say that a set $S \subseteq V(G)$ is a p -connected set in G if $p \notin S$ and $G[S \cup \{p\}]$ is connected. For a t -boundaried graph (G, L) , $P \subseteq V(G) \setminus L$, $k \in \mathbb{N}$ and $p \in V(G) \setminus L$, a set $S \subseteq V(G)$ is called a (p, P, k) -representative set for (G, L) if the following holds:

For every p -connected set $S \subseteq V(G) \setminus L$ of size at most k in G , S contains a p -connected set $\tilde{S} \subseteq V(G) \setminus L$ of size at most $|S|$ such that $\mathcal{M}_h(G - p - (S \cup P), L) \supseteq \mathcal{M}_h(G - p - (\tilde{S} \cup P), L)$.

We extend the notion of p -connected sets in a natural way to the additional case when p is not a vertex, but $p = \emptyset$. In this case, S is p -connected if $G[S]$ is connected. The definition of (p, P, k) -representative sets for $p = \emptyset$ is analogous.

The notion of p -connected sets is motivated by the following observation.

► **Observation 26.** Let S be a connected \mathcal{F} -deletion set of G of size at most $3k$ and let $C \in \mathcal{CC}(G_\tau)$. Let $S_C = S \cap C$ and suppose that S_C is non-empty. Let $J = N(C) \setminus S$ and consider the boundaried graph (G_C^J, J) , where G_C^J is defined as the graph obtained from $G[N[C]]$ as follows. If $T_C = N(C) \cap S$ is non-empty, then we identify the vertices in T_C into a single vertex v_J^* to obtain G_C^J . Otherwise, $G_C^J = G[N[C]]$ and set $v_J^* = \emptyset$. Then, S_C is a v_J^* -connected set of size at most $3k$ in the graph G_C^J .

The definition of the (p, P, k) -representative set S (Definition 25) guarantees that for every p -connected set $S \subseteq V(G) \setminus L$ of size at most k in G , there is a p -connected set in S that provides the “same” connectivity as S with respect to p , costs at most $|S|$, and hits every minor model hit by S in $G - p - P$ (and possibly more). The case when $p = \emptyset$ covers the case when the entire solution is contained in a single component of G_{τ^*} .

► **Lemma 27.** Let (G, L) be a t -boundaried graph for some $t \leq 3\eta + 3$, $k \in \mathbb{N}$, $p \in V(G) \setminus L$ and $P \subseteq V(G) \setminus L$. Then, there is a (p, P, k) -representative set for (G, L) of size $\mathcal{O}(k)$. Moreover, if $\text{tw}(G) = \mathcal{O}(\eta)$, then such a set can be computed in polynomial time.

We are now ready to describe our second and final marking rule. Recall that \mathcal{P} , \mathcal{Q} and P^∞ are as given by Lemma 23. Let C be a Type 2 component of $\mathcal{CC}(G_{\tau^*})$. Then, $C \in \mathcal{P}$. Let P_C^∞ denote the set $P^\infty \cap C$.

► **Marking Rule 2.** For each compatible set $J \in \mathcal{B}_C$, we construct a $|J|$ -boundaried graph (G_C^J, J) defined as the graph obtained from $(G[N[C]], J)$ by identifying all vertices of $N(C) \setminus J$ with the resulting vertex denoted by v_J^* . If $J = N(C)$, then we simply define $v_J^* := \emptyset$. We then use Lemma 27 to compute a $(v_J^*, P_C^\infty, 3k)$ -representative set Q_C^J of size $\mathcal{O}(k)$ for (G_C^J, J) and mark the vertices in it.

In the rule above, J is intended to represent the subset of $N(C)$ that is *not* deleted by a hypothetical connected \mathcal{F} -deletion set of size at most $3k$ that intersects $N[C]$. If $N(C) = J$, then any hypothetical connected \mathcal{F} -deletion set that intersects $N[C]$ and has an empty intersection with $N(C)$ must also be contained entirely in C . Consequently, we simulate this case by setting $v_J^* = \emptyset$. Note that (G_C^J, J) is a t -boundaried graph for some $t \leq 3\eta + 3$ because $|J \cap X| \leq \eta + 1$ and $|J \cap Z| \leq 2(\eta + 1)$. Moreover, $\text{tw}(G_C^J) \leq 2\eta + 2$. As the treewidth of $G[C \cup (N(C) \cap Z)]$ is at most η , it follows that adding v_J^* and a subset of X of size at most $\eta + 1$ can increase the treewidth of $G[C \cup (N(C) \cap Z)]$ to at most $2\eta + 2$.

We are now ready to prove Lemma 17.

78:12 On Approximate Compressions for Connected Minor-Hitting Sets

Proof of Lemma 17. We first invoke Lemma 18 to either correctly conclude that G has no \mathcal{F} -deletion set of size at most k or compute the pair $\tau^* = (X, Z)$. In the former case we return the same. Otherwise, we invoke Lemma 23 to compute the partition $\mathcal{P} \uplus \mathcal{Q}$ of $\mathcal{CC}(G_{\tau^*})$ and the set $P^\infty \subseteq V(\mathcal{P})$. Recall that the components contained in \mathcal{P} are called Type 2 components and those contained in \mathcal{Q} are called Type 1 components. We now execute Marking Rule 1.

For each $C \in \mathcal{CC}(G_{\tau^*})$, let R_C denote the set of marked vertices in the connected component C . Note that the *total* number of vertices marked across all components of G_{τ^*} in this step is at most $3k \cdot |X \cup Z|^\rho = k^{\mathcal{O}(\rho)}$.

Now, for each Type 2 component $C \in \mathcal{CC}(G_{\tau^*})$ we execute Marking Rule 2 with Q_C denoting the subset of its vertices marked in this execution. For every Type 1 component C , $Q_C = \emptyset$. Note that the number of vertices marked in each Type 2 component C in this step is bounded by $\mathcal{O}(k \cdot |\mathcal{B}_C|) = k^{\mathcal{O}(1)}$. Since the number of Type 2 components is bounded by $k^{\mathcal{O}(1)}$, we conclude that Marking Rule 2 marks a total of $k^{\mathcal{O}(1)}$ vertices across all connected components of G_{τ^*} .

We now argue that we have preserved a $(2 + \varepsilon)$ -approximation in the union of the marked set of vertices and $X \cup Z$.

▷ **Claim 28.** If G has a connected \mathcal{F} -deletion set S of size at most k , then it has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ contained in $X \cup Z \cup \bigcup_{C \in \mathcal{CC}(G_{\tau^*})} (Q_C \cup R_C)$.

Proof sketch. Let $S'' = S \cap (X \cup Z)$ and $A = S \setminus S''$. Suppose that $G[S'']$ is not connected. Consider the terminal set S'' with a weight function on the edges that assigns 0 to every edge with both endpoints in S'' and assigns 1 to every other edge. Then, we observe that $G[S]$ contains an S'' -Steiner tree with weight at most $|A| + \beta - 1$ where β is the number of connected components in $G[S'']$ and Proposition 21 guarantees that the set of vertices marked by Marking Rule 1 contains a set A'' such that $G[A'' \cup S'']$ is connected, $|A''| + \beta - 1 \leq (1 + \varepsilon)(|A| + \beta - 1)$, implying that $|A'' \cup S''| \leq (1 + \varepsilon)|S|$.

In order to prove the claim, we will show that once we have added A'' to S'' , we can (a) ignore all other vertices of Type 1 components originally contained in S and (b) consider each Type 2 component $C \in \mathcal{CC}(G_{\tau^*})$ independently and replace the set $S \cap C$ with a subset of Q_C of size at most $|S \cap C|$ in such a way that the resulting set is eventually still connected and hits all \mathcal{F} -minor models in G . Observe that if $S'' = \emptyset$, then $A'' = \emptyset$. Moreover, if $G[S'']$ is connected, then we set $A'' = \emptyset$. ◁

Finally, we define V^∞ as follows. $V^\infty = V(G) \setminus (X \cup Z \cup \bigcup_{C \in \mathcal{CC}(G_{\tau^*})} (Q_C \cup R_C))$. ◀

Theorem 2 can now be proved using Lemma 17.

4.2 The approximate compression for CONNECTED PLANAR \mathcal{F} -DELETION

The crux of our approximate compression is the following stronger version of Lemma 18, where we have now also ensured the presence of a $(2 + \varepsilon)$ -approximate solution within a polynomially bounded set L (which plays the role that the set $X \cup Z$ plays in Lemma 18).

► **Lemma 29.** *There is an algorithm that given $G, k, \tau^* = (X, Z)$ from Lemma 18, runs in time $k^{\mathcal{O}(\rho)} n^{\mathcal{O}(1)}$ and either correctly concludes that G has no connected \mathcal{F} -deletion set of size at most k or returns a set $L \subseteq V(G)$ such that the following statements hold.*

1. $L \supseteq (X \cup Z)$, $|L| = k^{\mathcal{O}(\rho)}$.
2. For every S which is a minimal connected \mathcal{F} -deletion set of G of size at most k , G has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ contained in L .

3. For every connected component C of $G - L$, $|N(C) \cap (L \setminus X)| \leq 2(\eta + 1)$.
4. For every \mathcal{F} -deletion set S of G of size at most $3k$ and any $C \in \mathcal{CC}(G - L)$, $|(N(C) \cap X) \setminus S| \leq \eta + 1$.

Proof. We invoke Lemma 17 to either conclude that G has no \mathcal{F} -deletion set of size at most k or compute the set V^∞ such that $|V(G) \setminus V^\infty| = k^{\mathcal{O}(\rho)}$ and for every S which is a minimal connected \mathcal{F} -deletion set of G of size at most k , G has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ disjoint from V^∞ . In the former case we return the same. Otherwise, let $Y = V(G) \setminus V^\infty$. Although it does not follow from the statement of Lemma 17, an inspection of the definition of V^∞ in the proof shows that $V^\infty \cap (X \cup Z) = \emptyset$ and so we may assume without loss of generality that $Y \supseteq X \cup Z$.

Consider the graph $G - X$ which is known to be \mathcal{F} -minor free and hence has treewidth at most η . Let (T, χ) be an arbitrary tree decomposition of $G - X$ of minimum width and suppose that it is arbitrarily rooted. We now define L to be the set $X \cup \text{LCA-closure}(T, \chi, Y \setminus X)$. The fact that $L \supseteq X \cup Z$ and $|L| = k^{\mathcal{O}(\rho)}$ follows from the fact that $X \cup Z \subseteq Y \subseteq L$, $|Y| = k^{\mathcal{O}(\rho)}$, and Lemma 10. Moreover, Lemma 10 also implies that the third statement holds. Similarly, the second statement follows from the fact that for every S which is a minimal connected \mathcal{F} -deletion set of G of size at most k , G has a connected \mathcal{F} -deletion set of size at most $(2 + \varepsilon)|S|$ contained in Y which is contained in L .

For the final statement, we know from Lemma 18 that for every $x, y \in X$, either $Z \subseteq Y \subseteq L$ intersects every x, y path in $G - (X \setminus \{x, y\})$ or there is an x - y flow of value at least $3k + \eta + 3$ in the graph $G - (X \setminus \{x, y\})$. Consequently, we can invoke Proposition 20 with the pair $(X, L \setminus X)$ instead of (X, Z) to conclude that for every \mathcal{F} -deletion set S of G of size at most $3k$ and any $C \in \mathcal{CC}(G - L)$, $|(N(C) \cap X) \setminus S| \leq \eta + 1$. This completes the proof of the lemma. \blacktriangleleft

Before proceeding to the description of the compression (Theorem 1), we need to define an annotated version of our problem. Recall that $h \geq \max\{10\eta, \max_{F \in \mathcal{F}} |F|\}$. In the ANNOTATED CONNECTED \mathcal{F} -DELETION problem, the input is a graph G , integer k , and a set $\mathcal{R} = \{(P_i, Q_i, \mathcal{T}_i)\}_{i \in [\ell]}$ where $\forall i \in [\ell]$, $P_i \cap Q_i = \emptyset$, $P_i \cup Q_i \subseteq V(G)$, $|Q_i| \leq 3(\eta + 1)$, and \mathcal{T}_i is a set of $|Q_i|$ -labeled graphs of size at most h . The goal is to decide whether there is a set $S \subseteq V(G)$ of size at most k such that $G[S]$ is connected and G_S has no minor isomorphic to a graph in \mathcal{F} , where G_S is defined as the graph obtained from G by going over all $i \in [\ell]$ and gluing a graph (H_i, Q_i) on to (G, Q_i) using the canonical labeling, whenever $P_i \subseteq S$ and $Q_i \cap S = \emptyset$, and where $\mathcal{M}_h(H_i, Q_i) = \mathcal{T}_i$.

A set S (not necessarily of size at most k) satisfying the properties above is said to be a feasible solution. Following the convention from [26], the parameterized optimization version of ANNOTATED CONNECTED \mathcal{F} -DELETION is defined to be a minimization problem with the optimization function $ACFD : \Sigma^* \times \mathbb{N} \times \Sigma^* \rightarrow \mathbb{R} \cup \{\infty\}$ defined as follows.

$$ACFD(G, \mathcal{R}, k, S) = \begin{cases} \infty & \text{if } S \text{ is not a feasible solution,} \\ \min\{|S|, k + 1\} & \text{otherwise.} \end{cases}$$

We next describe a compression algorithm that takes as input an instance of CONNECTED \mathcal{F} -DELETION and outputs an instance of ANNOTATED CONNECTED \mathcal{F} -DELETION with size bounded polynomially in k and in which a $(2 + \varepsilon)$ -approximate solution for the original instance is preserved.

► Lemma 30. *There is an algorithm that given G, k , runs in time $k^{\mathcal{O}(\rho)} n^{\mathcal{O}(1)}$ and either correctly concludes that G has no connected \mathcal{F} -deletion set of size at most k or returns an instance $\Gamma = (\tilde{G}, \tilde{k}, \tilde{\mathcal{R}} = \{(\tilde{P}_i, \tilde{Q}_i, \tilde{\mathcal{T}}_i)\}_{i \in [\ell]})$ of ANNOTATED CONNECTED \mathcal{F} -DELETION such that the following statements hold.*

1. $|V(\tilde{G})| = k^{\mathcal{O}(\rho)}$, $\ell = k^{\mathcal{O}(\rho)}$.
2. If G has a connected \mathcal{F} -deletion set S of size at most k , then Γ has a feasible solution of size at most $(2 + \varepsilon)|S|$.
3. Every feasible solution of Γ is a connected \mathcal{F} -deletion set of G .

Proof sketch. We begin by executing Lemma 18 and then Lemma 29 to either conclude that G has no connected \mathcal{F} -deletion set of size at most k or compute $\tau = (X, Z)$ and the set $L \supseteq X \cup Z$. We set $\tilde{G} = G[L]$ and $\tilde{k} = (2 + \varepsilon)k$. Recall that $|L| = k^{\mathcal{O}(\rho)}$.

If the number of connected components of $G - L$ is not already bounded by $k^{\mathcal{O}(\rho)}$, then we use the arguments from Lemma 23 (with L used in place of $X \cup Z$) to partition them into \mathcal{P} and \mathcal{Q} and compute the set $P^\infty \subseteq V(\mathcal{P})$ such that every \mathcal{F} -deletion set of size at most $3k$ for $G - V(\mathcal{Q}) - P^\infty$ is an \mathcal{F} -deletion set of G and $|\mathcal{P}| = k^{\mathcal{O}(\rho)}$. Observe that because we are now using L in place of $X \cup Z$, the size of the set $|\mathcal{P}|$ is now bounded by $k^{\mathcal{O}(\rho)}$ (because $|L| = k^{\mathcal{O}(\rho)}$) as opposed to $k^{\mathcal{O}(1)}$ where the degree of the polynomial only depends on \mathcal{F} .

We now define $\tilde{\mathcal{R}}$ as follows. Let $\hat{G} = G - V(\mathcal{Q}) - P^\infty$. For every $C \in \mathcal{CC}(\hat{G})$ and compatible set $J \in \mathcal{B}_C$ in the graph \hat{G} , we add to $\tilde{\mathcal{R}}$ the tuple $(N(C) \setminus J, J, \mathcal{M}_h(\hat{G}_C^J, J))$. This completes the definition of $\tilde{\mathcal{R}}$ and consequently, the definition of Γ . We now show that the three properties specified in the lemma hold.

Since we have $k^{\mathcal{O}(\rho)}$ connected components in $\mathcal{CC}(\hat{G})$ and each connected component has $k^{\mathcal{O}(1)}$ compatible sets, we conclude that $\ell = k^{\mathcal{O}(\rho)}$. It remains to prove the second and third statements of the lemma. For the second statement, Lemma 29 guarantees that if G has a connected \mathcal{F} -deletion set S of size at most k then it has a connected \mathcal{F} -deletion set S' of size at most $(2 + \varepsilon)|S|$ which is contained in L . We claim that S' is also a feasible solution for the constructed instance of ANNOTATED CONNECTED \mathcal{F} -DELETION. If this were not the case then there is an \mathcal{F} -minor model in the graph $\tilde{G}_{S'}$ as defined in the definition of the problem. However, the definition of the tuples in $\tilde{\mathcal{P}}$ implies that every \mathcal{F} -minor model present in $\tilde{G}_{S'}$ is also present in $G - S'$, a contradiction to S' being a connected \mathcal{F} -deletion set S' of G . Therefore, we conclude that the second statement holds.

For the third statement, let S be a feasible solution for the constructed instance of ANNOTATED CONNECTED \mathcal{F} -DELETION and suppose that it is not a connected \mathcal{F} -deletion set of G . Then, there is an \mathcal{F} -minor model in $G - S$. In fact, due to Lemma 23 and the definition of the sets P^∞ and \mathcal{Q} , we conclude that there is an \mathcal{F} -minor model in $G - V(\mathcal{Q}) - P^\infty - S = \hat{G} - S$ because otherwise S must be an \mathcal{F} -deletion set of G as well.

However, for every \mathcal{F} -minor model in $\hat{G} - S$, one can invoke “cut and paste” arguments similar to those used in the proof of Lemma 23 to construct an alternate minor model in \tilde{G}_S for the same graph in \mathcal{F} , a contradiction to our choice of S as a feasible solution of Γ . This completes the proof of the lemma. \blacktriangleleft

Theorem 1 now follows from Lemma 30. The reduction algorithm uses the algorithm of this lemma and the solution-lifting algorithm simply returns the approximate solution computed for the instance of ANNOTATED CONNECTED \mathcal{F} -DELETION.

5 Conclusion and Open Problems

Our result on the approximate compressibility of the connectivity constrained variant of PLANAR \mathcal{F} -DELETION demonstrates that preprocessing lower bounds can be side-stepped even for very general versions of problems such as Vertex Cover and Feedback Vertex Set as long as one allows a small loss in accuracy. While a step forward in our understanding of preprocessing under connectivity constraints, our work leaves some natural questions for follow-up work.

1. Is there a $(1 + \varepsilon)$ -approximate kernel for this problem (for every $0 < \varepsilon < 1$)?
2. What is the best approximation factor one can achieve for this problem in polynomial time?
3. Is there a fixed-parameter algorithm for CONNECTED PLANAR \mathcal{F} -DELETION with a single-exponential dependence on k ? To the best of our knowledge, such an algorithm is not known in the literature even when $\mathcal{F} = \{\theta_c\}$ for $c \geq 3$. Here, θ_c is the graph on 2 vertices with c parallel edges between them.

References

- 1 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.
- 2 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM J. Discrete Math.*, 28(1):277–305, 2014. doi:10.1137/120880240.
- 3 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
- 4 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.
- 5 Holger Dell and Dieter van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *J. ACM*, 61(4):23:1–23:27, 2014. doi:10.1145/2629620.
- 6 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.
- 7 Rodney G Downey and Michael Ralph Fellows. *Parameterized complexity*. Springer Science & Business Media, 2012.
- 8 S. E. Dreyfus and R. A. Wagner. The steiner problem in graphs. *Networks*, 1(3):195–207, 1971.
- 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 Samuel Fiorini, Gwenaél Joret, and Ugo Pietropaoli. Hitting diamonds and growing cacti. In *Integer Programming and Combinatorial Optimization, 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9-11, 2010. Proceedings*, pages 191–204, 2010. doi:10.1007/978-3-642-13036-6_15.
- 15 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM J. Discrete Math.*, 30(1):383–410, 2016. doi:10.1137/140997889.

- 16 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar \mathcal{F} -deletion: Approximation, kernelization and optimal FPT algorithms.
- 17 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.
- 18 Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct pcps for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.
- 19 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.
- 20 Danny Hermelin, Stefan Kratsch, Karolina Soltys, Magnus Wahlström, and Xi Wu. A completeness theory for polynomial (Turing) kernelization. *Algorithmica*, 71(3):702–730, 2015.
- 21 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.
- 22 Gwenaël Joret, Christophe Paul, Ignasi Sau, Saket Saurabh, and Stéphan Thomassé. Hitting and harvesting pumpkins. *SIAM J. Discrete Math.*, 28(3):1363–1390, 2014. doi:10.1137/120883736.
- 23 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within $2-\epsilon$. *Journal of Computer and System Sciences*, 74(3):335–349, 2008.
- 24 Stefan Kratsch. Recent developments in kernelization: A survey. *Bulletin of the EATCS*, 113, 2014.
- 25 Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Kernelization–preprocessing with a guarantee. In *The Multivariate Algorithmic Revolution and Beyond*, pages 129–161. Springer, 2012.
- 26 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.
- 27 Dániel Marx and Ildikó Schlotter. Obtaining a planar graph by vertex deletion. *Algorithmica*, 62(3-4):807–822, 2012. doi:10.1007/s00453-010-9484-z.
- 28 Jesper Nederlof. Fast polynomial-space algorithms using möbius inversion: Improving on steiner tree and related problems. In *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, pages 713–725, 2009. doi:10.1007/978-3-642-02927-1_59.
- 29 Geevarghese Philip, Venkatesh Raman, and Yngve Villanger. A quartic kernel for pathwidth-one vertex deletion. In *Graph Theoretic Concepts in Computer Science - 36th International Workshop, WG 2010, Zarós, Crete, Greece, June 28-30, 2010 Revised Papers*, pages 196–207, 2010. doi:10.1007/978-3-642-16926-7_19.
- 30 M. S. Ramanujan. An approximate kernel for connected feedback vertex set. In *27th Annual European Symposium on Algorithms, ESA 2019, September 9-11, 2019, Munich/Garching, Germany*, volume 144 of *LIPICs*, pages 77:1–77:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPICs.ESA.2019.77.