Hitting forbidden minors: Approximation and Kernelization

Fedor V. Fomin[*]   Daniel Lokshtanov[†]   Neeldhara Misra[‡]   Geevarghese Philip[‡]

Saket Saurabh[‡]

Abstract

We study a general class of problems called $\mathcal{F}$-DELETION problems. In an $\mathcal{F}$-DELETION problem, we are asked whether a subset of at most k vertices can be deleted from a graph G such that the resulting graph does not contain as a minor any graph from the family $\mathcal{F}$ of forbidden minors. We obtain a number of algorithmic results on the $\mathcal{F}$-DELETION problem when $\mathcal{F}$ contains a planar graph. We give

- a linear vertex kernel on graphs excluding t-claw $K_{1,t}$, the star with t leves, as an induced subgraph, where t is a fixed integer.
- an approximation algorithm achieving an approximation ratio of $O(\log^{3/2} OPT)$, where OPT is the size of an optimal solution on general undirected graphs.

Finally, we obtain polynomial kernels for the case when $\mathcal{F}$ contains graph θ_c as a minor for a fixed integer c. The graph θ_c consists of two vertices connected by c parallel edges. Even though this may appear to be a very restricted class of problems it already encompasses well-studied problems such as VERTEX COVER, FEEDBACK VERTEX SET and DIAMOND HITTING SET. The generic kernelization algorithm is based on a non-trivial application of protrusion techniques, previously used only for problems on topological graph classes.

1 Introduction

Let $\mathcal{F}$ be a finite set of graphs. In an $\mathcal{F}$-DELETION problem[1], we are given an n-vertex graph G and an integer k as input, and asked whether at most k vertices can be deleted from G such that the resulting graph does not contain a graph from $\mathcal{F}$ as a minor. More precisely, the problem is defined as follows:

$\mathcal{F}$-DELETION
  Instance: A graph G and a non-negative integer k.
  Parameter: k
  Question: Does there exist $S \subseteq V(G)$, $|S| \leq k$,
            such that $G \setminus S$ contains no graph from $\mathcal{F}$ as a minor?

We refer to such a subset S as a $\mathcal{F}$-hitting set. The $\mathcal{F}$-DELETION problem is a generalization of several fundamental problems. For example, when $\mathcal{F} = \{K_2\}$, a complete graph on two vertices, this is the VERTEX COVER problem. When $\mathcal{F} = \{C_3\}$, a cycle on three vertices, this is the FEEDBACK VERTEX SET problem. Other famous cases are $\mathcal{F} = \{K_{2,3}, K_4\}$, $\mathcal{F} = \{K_{3,3}, K_5\}$ and $\mathcal{F} = \{K_3, T_2\}$, which correspond to removing vertices to obtain outerplanar graphs, planar graphs, and graphs of pathwidth one respectively. Here, $K_{i,j}$ denotes the complete bipartite graph with bipartitions of sizes i and j, and K_i denotes the complete graph on i vertices. Further, a T_2 is a star on three leaves, each of whose edges has been

---

[*]Department of Informatics, University of Bergen, N-5020 Bergen, Norway. `fomin@ii.uib.no`.

[†]University of California, San Diego, La Jolla, CA 92093-0404, USA, `dlokshtanov@cs.ucsd.edu`

[‡]The Institute of Mathematical Sciences, Chennai - 600113, India. `{neeldhara|gphilip|saket}@imsc.res.in`.

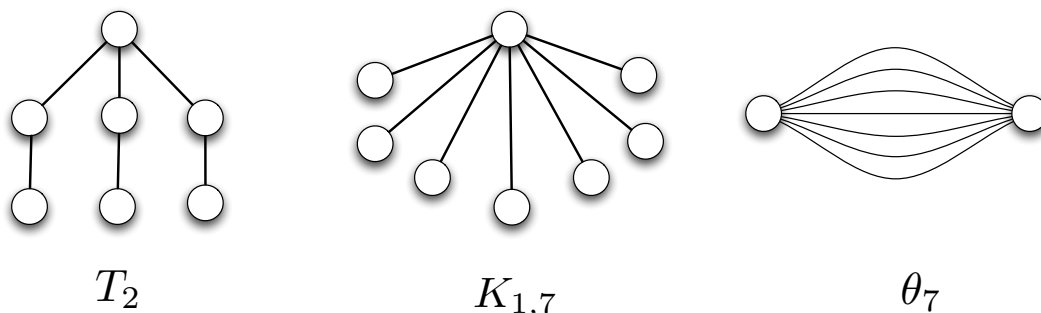[1]We use prefix p to distinguish the parameterized version of the problem.

Figure 1: Graphs T_2, t-claw $K_{1,t}$ with $t = 7$, and θ_c with $c = 7$

subdivided exactly once. A T_2 structure is depicted in the leftmost graph of Figure 1. In the literature, these problems are known as p-OUTERPLANAR DELETION SET, p-PLANAR DELETION SET and p-PATHWIDTH ONE DELETION SET respectively.

Our interest in the $\mathcal{F}$-DELETION problem is motivated by its generality and the recent developments in kernelization or polynomial time preprocessing. The parameterized complexity of this general problem is well understood. By a celebrated result of Robertson and Seymour, every $\mathcal{F}$-DELETION problem is fixed-parameter tractable (FPT). That is, there is an algorithm solving the problem in time $O(f(k) \cdot n^3)$ [44]. In this paper we study this problem from the view point of polynomial time preprocessing and approximation, when the obstruction set $\mathcal{F}$ satisfies certain properties.

Preprocessing as a strategy for coping with hard problems is universally applied in practice and the notion of *kernelization* provides a mathematical framework for analyzing the quality of preprocessing strategies. We consider parameterized problems, where every instance I comes with a *parameter k*. Such a problem is said to admit a *polynomial kernel* if every instance (I, k) can be reduced in polynomial time to an equivalent instance with both size and parameter value bounded by a polynomial in k. The study of kernelization is a major research frontier of Parameterized Complexity and many important recent advances in the area are on kernelization. These include general results showing that certain classes of parameterized problems have polynomial kernels [3, 12, 31, 38]. The recent development of a framework for ruling out polynomial kernels under certain complexity-theoretic assumptions [11, 25, 32] has added a new dimension to the field and strengthened its connections to classical complexity. For overviews of kernelization we refer to surveys [10, 33] and to the corresponding chapters in books on Parameterized Complexity [30, 42].

While the initial interest in kernelization was driven mainly by practical applications, the notion of kernelization turned out to be very important in theory as well. It is well known, see e.g. [26], that a parameterized problem is fixed parameter tractable, or belongs to the class FPT, if and only if it has a (perhaps exponential) kernel. Kernelization enables us to classify problems within the class FPT further, based on the sizes of the problem kernels. So far, most of the work done in the field of kernelization is still specific to particular problems and powerful unified techniques to identify classes of problems with polynomial kernels are still in their nascent stage. One of the fundamental challenges in the area is the possibility of characterising general classes of parameterized problems possessing kernels of polynomial sizes. From this perspective, the class of the $\mathcal{F}$-DELETION problems is very interesting because it contains as special cases the p-VERTEX COVER and p-FEEDBACK VERTEX SET problems which are the most intensively studied problems from the kernelization perspective.

Our contribution and key ideas. One of the main conceptual contributions of this work is the extension of protrusion techniques, initially developed in [12, 31] for obtaining meta-kernelization theorems for problems on sparse graphs like planar and H-minor-free graphs, to general graphs. We demonstrate

2

this by obtaining a number of kernelization results for the $\mathcal{F}$-Deletion problem when $\mathcal{F}$ contains a planar graph. Our first result is the following theorem for graphs that do not contain $K_{1,t}$ (a star on t leaves, see Figure 1) as an induced subgraph.

Theorem 1. *Let $\mathcal{F}$ be an obstruction set containing a planar graph. Then $\mathcal{F}$-Deletion admits a linear vertex kernel on graphs excluding $K_{1,t}$ as an induced subgraph, where t is a fixed integer.*

Several well studied graph classes do not contain graphs with induced $K_{1,t}$. Of course, every graph with maximum vertex degree at most $(t-1)$ is $K_{1,t}$-free. The class of $K_{1,3}$-free graphs, also known as claw-free graphs, contains line graphs and de Bruijn graphs. Unit disc graphs are known to be $K_{1,7}$-free [19]. We remark that the number of vertices in the kernels of Theorem 1 is (up to a multiplicative constant) optimal, unless P=NP.

Our kernelization is a divide and conquer algorithm which finds large protrusions. A *protrusion* is a subgraph of constant treewidth separated from the remaining part of the graph by a constant number of vertices. Having found protrusions of substantial size, the kernelization algorithm replaces them with smaller, "equivalent" protrusions. Here we use the results from the work by Bodlaender et al. [12] that enable this step whenever the parameterized problem in question "behaves like a regular language". To prove that $\mathcal{F}$-Deletion has the desired properties for this step, we formulate the problem in monadic second order logic and show that it exhibits certain monotonicity properties. As a corollary we obtain that p-Feedback Vertex Set, p-Diamond Hitting Set, p-Pathwidth One Deletion Set, and p-Outerplanar Deletion Set admit a linear vertex kernel on graphs excluding $K_{1,t}$ as an induced subgraph. With the same methodology we also obtain a $O(k \log k)$ vertex kernel for p-Disjoint Cycle Packing on graphs excluding $K_{1,t}$ as an induced subgraph. We note that p-Disjoint Cycle Packing does not admit a polynomial kernel on general graphs [13] unless $coNP \subseteq NP/poly$.

Let θ_c be a graph with two vertices and $c \geq 1$ parallel edges (see Figure 1). Our second result is the following theorem on general graphs.

Theorem 2. *Let $\mathcal{F}$ be an obstruction set containing θ_c. Then $\mathcal{F}$-Deletion admits a kernel of size $O(k^2 \log^{3/2} k)$.*

A number of well-studied NP-hard combinatorial problems are special cases of p-θ_c-Free-Deletion. When $c = 1$, this is the classical Vertex Cover problem [41]. For $c = 2$, this is another well studied problem, the Feedback Vertex Set problem [5, 7, 18, 35]. When $c = 3$, this is the Diamond Hitting Set problem [29]. Let us note that the size of the best known kernel for $c = 2$ is $O(k^2)$, which is very close to the size of the kernel in Theorem 2. Also, Dell and van Melkebeek proved that no NP-hard vertex deletion problem based on a graph property that is inherited by subgraphs can have kernels of size $O(k^{2-\varepsilon})$ unless $coNP \subseteq NP/poly$ [25] and thus the sizes of the kernels in Theorem 2 are tight up to a polylogarithmic factor.

The proof of Theorem 2 is obtained in a series of non-trivial steps. The very high level idea is to reduce the general case to problem on graphs of bounded degree, which allows us to use the protrusion techniques as in the proof of Theorem 1. However, vertex degree reduction is not straightforward and requires several new ideas. One of the new tools is a generic $O(\log^{3/2} OPT)$-approximation algorithm for the $\mathcal{F}$-Deletion problem when the class of excluded minors for $\mathcal{F}$ contains at least one planar graph. More precisely, we obtain the following result, which is of independent interest.

Theorem 3. *Let $\mathcal{F}$ be an obstruction set containing a planar graph, and let OPT be the size of the smallest F-hitting set. Given a graph G, in polynomial time we can find a subset $S \subseteq V(G)$ such that $G[V \setminus S]$ contains no element of $\mathcal{F}$ as a minor and $|S| = O(OPT \cdot \log^{3/2} OPT)$.*

While several generic approximation algorithms are known for problems of minimum vertex deletion to obtain subgraphs with property P, like when P is a hereditary property with a finite number of minimal forbidden subgraphs [40], or can be expressed as a universal first order sentence over subsets of edges of

the graph [37], we are not aware of any generic approximation algorithm for the case when a property P is characterized by a finite set of forbidden minors.

We then use the approximation algorithm as a subroutine in a polynomial time algorithm that transforms the input instance (G, k) into an equivalent instance (G', k') such that $k' \leq k$ and the maximum degree of G' is bounded by $O(k \log^{3/2} k)$. An important combinatorial tool used in designing this algorithm is the q–Expansion Lemma. For $q = 1$ this lemma is Hall's theorem and its usage is equivalent to the application of the Crown Decomposition technique [1, 17]. After obtaining an equivalent instance with bounded degree, we apply protrusion techniques and prove Theorem 2.

Related work. All non-trivial $\mathcal{F}$-DELETION problems are NP-hard [39]. By one of the most well-known consequences of the celebrated Graph Minor theory of Robertson and Seymour, the $\mathcal{F}$-DELETION problem is fixed parameter tractable for every finite set of forbidden minors. A special case of that problem, when the set $\mathcal{F}$ contains θ_c, has been studied from approximation and parameterized perspectives. In particular, the case of p-θ_1-DELETION or, equivalently, p-VERTEX COVER, is the most well-studied problem in Parameterized Complexity. Different kernelization techniques were applied on the problem, eventually resulting in a $2k$-sized vertex kernel [1, 16, 24, 34]. For the kernelization of p-FEEDBACK VERTEX SET, or p-θ_2-DELETION, there has been a sequence of dramatic improvements starting from an $O(k^{11})$ vertex kernel by Buragge et al. [15], improved to $O(k^3)$ by Bodlaender [9], and then finally to $O(k^2)$ by Thomassé [46]. Recently Philip et al. [43] and Cygan et al. [23] obtained polynomial kernels for p-PATHWIDTH ONE DELETION SET. Constant factor approximation algorithms are known for VERTEX COVER and FEEDBACK VERTEX SET [5, 6]. Very recently, a constant factor approximation algorithm for the DIAMOND HITTING SET problem, or p-θ_3-DELETION, was obtained in [29]. Prior to our work, no polynomial kernels were known for p-DIAMOND HITTING SET or more general families of $\mathcal{F}$-DELETION problems.

The remaining part of the paper is organised as follows. In Section 2 we provide preliminaries on basic notions from Graph Theory and Logic used in the paper. Section 3 is devoted to the proof of Theorem 1. In Section 4 we give an approximation algorithms proving Theorem 3. The proof of Theorem 2 is given in Section 5. We conclude with open questions in Section 6.

2 Preliminaries

In this section we give various definitions which we use in the paper. For $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, \dots, n\}$. We use $V(G)$ to denote the vertex set of a graph G, and $E(G)$ to denote the edge set. The degree of a vertex v in G is the number of edges incident on v, and is denoted by $d(v)$. We use $\Delta(G)$ to denote the maximum degree of G. A graph G' is a *subgraph* of G if $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. The subgraph G' is called an *induced subgraph* of G if $E(G') = \{\{u, v\} \in E(G) \mid u, v \in V(G')\}$. Given a subset $S \subseteq V(G)$ the subgraph induced by S is denoted by $G[S]$. The subgraph induced by $V(G) \setminus S$ is denoted by $G \setminus S$. We denote by $N(S)$ the open neighborhood of S, i.e. the set of vertices in $V(G) \setminus S$ adjacent to S. Let $\mathcal{F}$ be a finite set of graphs. A vertex subset $S \subseteq V(G)$ of a graph G is said to be a $\mathcal{F}$-*hitting set* if $G \setminus S$ does not contain any graphs in the family $\mathcal{F}$ as a minor.

By *contracting* an edge (u, v) of a graph G, we mean identifying the vertices u and v, keeping all the parallel edges and removing all the loops. A *minor* of a graph G is a graph H that can be obtained from a subgraph of G by contracting edges. We keep parallel edges after contraction since the graph θ_c which we want to exclude as a minor itself contains parallel edges.

Let G, H be two graphs. A subgraph G' of G is said to be a *minor-model* of H in G if G' contains H as a minor. The subgraph G' is a *minimal minor-model* of H in G if no proper subgraph of G' is a minor-model of H in G.

A graph class C is *minor closed* if any minor of any graph in C is also an element of C. A minor closed graph class C is H-*minor-free* or simply H-*free* if $H \notin C$.

2.1 Monadic Second Order Logic (MSO)

The syntax of MSO on graphs includes the logical connectives $\lor$, $\land$, $\lnot$, $\Leftrightarrow$, $\Rightarrow$, variables for vertices, edges, sets of vertices and sets of edges, the quantifiers $\forall$, $\exists$ that can be applied to these variables, and the following five binary relations:

1. $u \in U$ where u is a vertex variable and U is a vertex set variable;

2. $d \in D$ where d is an edge variable and D is an edge set variable;

3. $\mathbf{inc}(d, u)$, where d is an edge variable, u is a vertex variable, and the interpretation is that the edge d is incident on the vertex u;

4. $\mathbf{adj}(u, v)$, where u and v are vertex variables u, and the interpretation is that u and v are adjacent;

5. equality of variables representing vertices, edges, set of vertices and set of edges.

Many common graph-theoretic notions such as vertex degree, connectivity, planarity, being acyclic, and so on, can be expressed in MSO, as can be seen from introductory expositions [14, 21]. Of particular interest to us are p-MIN-MSO problems. In a p-MIN-MSO graph problem Π, we are given a graph G and an integer k as input. The objective is to decide whether there is a vertex/edge set S of size at most k such that the MSO-expressible predicate $P_\Pi(G, S)$ is satisfied.

2.2 Parameterized algorithms and 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. A central notion in parameterized complexity is *fixed parameter tractability (FPT)* which means, for a given instance (x, k), solvability in time $f(k) \cdot p(|x|)$, where f is an arbitrary function of k and p is a polynomial in the input size. The notion of *kernelization* is formally defined as follows.

Definition 1. [**Kernelization, Kernel**] [30] A kernelization algorithm for a parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$ is an algorithm that, given $(x, k) \in \Sigma^* \times \mathbb{N}$, outputs, in time polynomial in $(|x| + k)$, a pair $(x', k') \in \Sigma^* \times \mathbb{N}$ such that: (a) $(x, k) \in \Pi$ if and only if $(x', k') \in \Pi$ and (b) $|x'|, k' \leq g(k)$, where g is some computable function. The output instance x' is called the kernel, and the function g is referred to as the size of the kernel. If $g(k) = k^{O(1)}$, then we say that Π admits a polynomial kernel.

2.3 Tree-width and protrusions

Let G be a graph. A *tree decomposition* of a graph G is a pair $(T, X = \{X_t\}_{t \in V(T)})$ such that

- $\cup_{t \in V(T)} X_t = V(G)$,
- for every edge $\{x, y\} \in E(G)$ there is a $t \in V(T)$ such that $\{x, y\} \subseteq X_t$, and
- for every vertex $v \in V(G)$ the subgraph of T induced by the set $\{t \mid v \in X_t\}$ is connected.

The *width* of a tree decomposition is $(\max_{t \in V(T)} |X_t|) - 1$ and the *treewidth* of G is the minimum width over all tree decompositions of G. A tree decomposition (T, X) is called a *nice tree decomposition* if T is a tree rooted at some node r where $X_r = \emptyset$, each node of T has at most two children, and each node is of one of the following kinds:

1. *Introduce node*: a node t that has only one child t' where $X_t \supset X_{t'}$ and $|X_t| = |X_{t'}| + 1$.
2. *Forget node*: a node t that has only one child t' where $X_t \subset X_{t'}$ and $|X_t| = |X_{t'}| - 1$.
3. *Join node*: a node t with two children t_1 and t_2 such that $X_t = X_{t_1} = X_{t_2}$.
4. *Base node*: a node t that is a leaf of t, is different than the root, and $X_t = \emptyset$.

Notice that, according to the above definition, the root r of T is either a forget node or a join node. It is well known that any tree decomposition of G can be transformed into a nice tree decomposition in time $O(|V(G)| + |E(G)|)$ maintaining the same width [36]. We use G_t to denote the graph induced on the vertices $\cup_{t'} X'_{t'}$, where t' ranges over all descendants of t, including t. We use H_t to denote $G_t[V(G_t) \setminus X_t]$.

Given a graph G and $S \subseteq V(G)$, we define $\partial_G(S)$ as the set of vertices in S that have a neighbor in $V(G) \setminus S$. For a set $S \subseteq V(G)$ the neighborhood of S is $N_G(S) = \partial_G(V(G) \setminus S)$. When it is clear from the context, we omit the subscripts. We now define the notion of a *protrusion*.

Definition 2. [r-**protrusion**] Given a graph G, we say that a set $X \subseteq V(G)$ is an *r-protrusion* of G if $\mathbf{tw}(G[X]) \leq r$ and $|\partial(X)| \leq r$.

2.4 t-Boundaried Graphs

In this section we define *t-boundaried graphs* and various operations on them. Throughout this section, t is an arbitrary positive integer.

Definition 3. [t-**Boundaried Graphs**] A t-boundaried graph is a graph G with t distinguished vertices, uniquely labeled from 1 to t. The set $\partial(G)$ of labeled vertices is called the boundary of G. The vertices in $\partial(G)$ are referred to as boundary vertices or terminals.

For a graph G and a vertex set $S \subseteq V(G)$, we will sometimes consider the graph $G[S]$ as the $|\partial(S)|$-boundaried graph with $\partial(S)$ being the boundary.

Definition 4. [**Gluing by** $\oplus$] Let G_1 and G_2 be two t-boundaried graphs. We denote by $G_1 \oplus G_2$ the t-boundaried graph obtained by taking the disjoint union of G_1 and G_2 and identifying each vertex of $\partial(G_1)$ with the vertex of $\partial(G_2)$ with the same label; that is, we glue them together on the boundaries. In $G_1 \oplus G_2$ there is an edge between two labeled vertices if there is an edge between them in G_1 or in G_2.

In this paper, t-boundaried graphs often come coupled with a vertex set which represents a partial solution to some optimization problem. For ease of notation we define $\mathcal{H}_t$ be to be the set of pairs (G, S), where G is a t-boundaried graph and $S \subseteq V(G)$.

Definition 5. [**Replacement**] Let G be a graph containing a r-protrusion X. Let G_1 be an r-boundaried graph. The act of replacing $G[X]$ with G_1 corresponds to changing G into $G[(V(G) \setminus X) \cup \partial(X)] \oplus G_1$.

2.5 Finite Integer Index

Definition 6. [**Canonical Equivalence**] For a parameterized problem Π and two t-boundaried graphs G_1 and G_2, we say that $G_1 \equiv_\Pi G_2$ if there exists a constant c such that for all t-boundaried graphs G_3 and for all k,

$$(G_1 \oplus G_3, k) \in \Pi \text{ if and only if } (G_2 \oplus G_3, k + c) \in \Pi.$$

Definition 7. [**Finite Integer Index**] We say that a parameterized problem Π has *finite integer index* if for every t there exists a finite set $\mathcal{S}$ of t-boundaried graphs such that for any t-boundaried graph G_1 there exists $G_2 \in \mathcal{S}$ such that $G_2 \equiv_\Pi G_1$. Such a set $\mathcal{S}$ is called a set of representatives for (Π, t).

Note that for every t, the relation $\equiv_\Pi$ on t-boundaried graphs is an equivalence relation. A problem Π is finite integer index if and only if for every t, $\equiv_\Pi$ is of finite index, that is, has a finite number of equivalence classes. The notion of *strong monotonicity* is an easily checked sufficient condition for a p-ᴍɪɴ-MSO problem to have finite integer index.

Definition 8. [**Signatures**] Let Π be a p-ᴍɪɴ-MSO problem. For a t-boundaried graph G we define the *signature function* $\zeta_G^\Pi : \mathcal{H}_t \to \mathbb{N} \cup \{\infty\}$ as follows. For a pair $(G', S') \in \mathcal{H}_t$, if there is no set $S \subseteq V(G)$ ($S \subseteq E(G)$) such that $P_\Pi(G \oplus G', S \cup S')$ holds, then $\zeta_G^\Pi((G', S')) = \infty$. Otherwise $\zeta_G^\Pi((G', S'))$ is the size of the smallest $S \subseteq V(G)$ ($S \subseteq E(G)$) such that $P_\Pi(G \oplus G', S \cup S')$ holds.

Definition 9. [**Strong Monotonicity**] A p-MIN-MSO problem Π is said to be *strongly monotone* if there exists a function $f : \mathbb{N} \to \mathbb{N}$ such that the following condition is satisfied. For every t-boundaried graph G, there is a subset $S \subseteq V(G)$ such that for every $(G', S') \in \mathcal{H}_t$ such that $\zeta_G^\Pi((G', S'))$ is finite, $P_\Pi(G \oplus G', S \cup S')$ holds and $|S| \leq \zeta_G^\Pi((G', S')) + f(t)$.

2.6 MSO Formulations

We now give MSO formulations for some properties involving $\mathcal{F}$ or θ_c that we use in our arguments. For a graph G and a vertex set $S \subseteq V(G)$, let $Conn(G, S)$ denote the MSO formula which states that $G[S]$ is connected, and let $MaxConn(G, S)$ denote the MSO formula which states that $G[S]$ is a maximal connected subgraph of G.

H minor-models. Let $\mathcal{F}$ be the finite forbidden set. For a graph G, we use $\phi_H(G)$ to denote an MSO formula which states that G contains H as a minor — equivalently, that G contains a minimal H minor model. Let $V(H) = \{h_1, \ldots, h_c\}$. Then, $\phi_H(G)$ is given by:

$$\phi_H(G) \equiv \exists X_1, \ldots, X_c \subseteq V(G)[$$
$$\bigwedge_{i \neq j}(X_i \cap X_j = \emptyset) \wedge \bigwedge_{1 \leq i \leq c} Conn(G, X_i) \wedge$$
$$\bigwedge_{(h_i, h_j) \in E(H)} \exists x \in X_i \wedge y \in X_j[(x, y) \in E(G)]$$
$$] \tag{1}$$

Minimum-size $\mathcal{F}$-hitting set. A minimum-size $\mathcal{F}$-hitting set of graph G can be expressed as:

$$\text{Minimize } S \subseteq V(G)[\bigwedge_{H \in \mathcal{F}} \neg\phi_H(G \setminus S)] \tag{2}$$

Largest θ_c "flower". Let v be a vertex in a graph G. A maximum-size set M of θ_c minor-models in G, all of which pass through v and no two of which share any vertex other than v, can be represented as:

$$\text{Maximize } S \subseteq V(G)[$$
$$\exists F \subseteq E(G)[\forall x \in S[$$
$$\exists X \subseteq V'[MaxConn(G', X) \wedge x \in X \wedge \forall y \in S[y \neq x \implies y \notin X] \wedge \phi_c(X \cup \{v\})]$$
$$]]] \tag{3}$$

Here G' is the graph with vertex set $V(G)$ and edge set F, and $V' = V(G) \setminus \{v\}$. S is a system of distinct representatives for the vertex sets that constitute the elements of M.

3 Kernelization for $\mathcal{F}$-DELETION on $K_{1,t}$ free graphs

In this section we show that if the obstruction set $\mathcal{F}$ contains a planar graph then the $\mathcal{F}$-DELETION problem has a linear vertex kernel on graphs excluding $K_{1,t}$ as an induced subgraph. We start with the following lemma, which is crucial to our kernelization algorithms.

Lemma 1. *Let $\mathcal{F}$ be an obstruction set containing a planar graph of size h. If G has a $\mathcal{F}$-hitting set of S size at most k, then $\mathbf{tw}(G \setminus S) \leq d$ and $\mathbf{tw}(G) \leq k + d$, where $d = 20^{2(14h-24)^5}$.*

Proof. By assumption, $\mathcal{F}$ contains at least one planar graph. Let h be the size of the smallest planar graph H contained in $\mathcal{F}$. By a result of Robertson et al. [45], H is a minor of the $(\ell \times \ell)$-grid, where $\ell = 14h - 24$. In the same paper Robertson et al. [45] have shown that any graph with treewidth greater than $20^{2\ell^5}$ contains a $(\ell \times \ell)$-grid as a minor. Let S be a $\mathcal{F}$–hitting set of G of size at most k. Since the $(\ell \times \ell)$-grid contains H as a minor, we have that $\mathbf{tw}(G \setminus S) \leq 20^{2\ell^5}$. Therefore, $\mathbf{tw}(G) \leq k + d$, where $d = 20^{2\ell^5}$ — indeed, a tree decomposition of width $(k + d)$ can be obtained by adding the vertices of S to every bag in an optimal tree decomposition of $G \setminus S$. This completes the proof of the lemma. $\qquad\square$

3.1 The Protrusion Rule — Reductions Based on Finite Integer Index

Wo obtain our kernelization algorithm for $\mathcal{F}$-DELETION by applying a protrusion based reduction rule. That is, any large r-protrusion for a fixed constant r that depends only on $\mathcal{F}$ (that is, only on the problem) is replaced with a smaller equivalent r-protrusion. For this, we utilize the following lemma of Bodlaender et al. [12].

Lemma 2 ([12]). *Let Π be a problem that has finite integer index. Then there exists a computable function $\gamma : \mathbb{N} \to \mathbb{N}$ and an algorithm that given an instance (G, k) and an r-protrusion X of G of size at least $\gamma(r)$, runs in $O(|X|)$ time and outputs an instance (G^*, k^*) such that $|V(G^*)| < |V(G)|$, $k^* \leq k$, and $(G^*, k^*) \in \Pi$ if and only if $(G, k) \in \Pi$.*

Remark: Let us remark that if G does not have $K_{1,t}$ as an induced subgraph then the proof of Lemma 2 also ensures that the graph G' does not contain $K_{1,t}$ as an induced subgraph. This ensures that the reduced instance is belongs to the same graph class as the original. The remark is not only true about the class of graphs excluding $K_{1,t}$ as an induced subgraph, but also for any graph class $\mathcal{G}$ that can be characterized by either finite set of forbidden subgraphs or induced subgraphs or minors. That is, if G is in $\mathcal{G}$ then the graph G' returned by Lemma 2 is also in $\mathcal{G}$.

In order to apply Lemma 2 we need to be able to efficiently find large r-protrusions whenever the instance considered is large enough. Also, we need to prove that $\mathcal{F}$-DELETION has finite integer index. The next lemma yields a divide and conquer algorithm for efficiently finding large r-protrusions.

Lemma 3. *There is a linear time algorithm that given an n-vertex graph G and a set $X \subseteq V(G)$ such that $\mathbf{tw}(G \setminus X) \leq d$, outputs a $2(d + 1)$-protrusion of G of size at least $\frac{n-|X|}{4|N(X)|+1}$. Here d is some constant.*

Proof. Let $F = G \setminus X$. The algorithm starts by computing a nice tree decomposition of F of width at most d. Notice that since d is a constant this can be done in linear time [8]. Let S be the vertices in $V(F)$ that are neighbors of X in G, that is, $S = N_G(X)$.

The nice tree decomposition of F is a pair $(T, \mathcal{B} = \{B_\ell\}_{\ell \in V(T)})$, where T is a rooted binary tree. We will now *mark* some of the nodes of T. For every $v \in S$, we mark the topmost node ℓ in T such that $v \in B_\ell$. In this manner, at most $|S|$ nodes are marked. Now we mark more nodes of T by exhaustively applying the following rule: if u and v are marked, mark their least common ancestor in T. Let M be the set of all marked nodes of T. Standard counting arguments on trees give that $|M| \leq 2|S|$.

Since T is a binary tree, it follows that $T \setminus M$ has at most $2|M| + 1$ connected components. Let the vertex sets of these connected components be $C_1, C_2 \ldots C_\eta$, $\eta \leq 2|M| + 1$. For every $i \leq \eta$, let $C_i' = N_T(C_i) \cup C_i$ and let $P_i = \bigcup_{u \in C_i'} B_u$. By the construction of M, every component of $T \setminus M$ has at most 2 neighbors in M. Also for every $1 \leq i \leq \eta$ and $v \in S$, we have that if $v \in P_i$, then v should be contained in one of the bags of $N_T(C_i)$. In other words, $S \cap P_i \subseteq \bigcup_{u \in C_i' \setminus C_i} B_u$. Thus every P_i is a $2(d + 1)$-protrusion of G. Since $\eta \leq 2|M| + 1 \leq 4|S| + 1$, the pigeon-hole principle yields that there is a protrusion P_i with at least $\frac{n-|X|}{4|S|+1}$ vertices. The algorithm constructs M and $P_1 \ldots P_\eta$ and outputs the largest protrusion P_i. It is easy to implement this procedure to run in linear time. This concludes the proof. $\qquad\square$

No we show that $\mathcal{F}$-DELETION has finite integer index. For this we need the following lemma.

Lemma 4 ([12]). *Every strongly monotone p-MIN-MSO problem has finite integer index.*

Lemma 5. $\mathcal{F}$-DELETION *has finite integer index.*

Proof. One can easily formulate $\mathcal{F}$-DELETION in MSO, which shows that it is a p-MIN-MSO problem (see Section 2.6). To complete the proof that $\mathcal{F}$-DELETION has finite integer index we show that $\Pi = \mathcal{F}$-DELETION is strongly monotone. Given a t-boundaried graph G, with $\partial(G)$ as its boundary, let $S'' \subseteq V(G)$ be a minimum set of vertices in G such that $G \setminus S''$ does not contain any graph in $\mathcal{F}$ as a minor. Let $S = S'' \cup \partial(G)$.

Now for any $(G', S') \in \mathcal{H}_t$ such that $\zeta_G^{\Pi}((G', S'))$ is finite, we have that $G \oplus G'[(V(G) \cup V(G')) \setminus (S \cup S')]$ does not contain any graph in $\mathcal{F}$ as a minor and $|S| \le \zeta_G^{\Pi}((G', S')) + t$. This proves that $\mathcal{F}$-DELETION is strongly monotone. By Lemma 4, $\mathcal{F}$-DELETION has finite integer index. □

3.2 Analysis and Kernel Size – Proof of Theorem 1

Now we give the desired kernel for $\mathcal{F}$-DELETION. We first prove an useful combinatorial lemma.

Lemma 6. *Let G be a graph excluding $K_{1,t}$ as an induced subgraph and S be a $\mathcal{F}$-hitting set. If $\mathcal{F}$ contains a planar graph of size h, then $|N(S)| \le g(h,t) \cdot |S|$ for some function g of h and t.*

Proof. By Lemma 1, $\mathbf{tw}(G \setminus S) \le d$ for $d = 20^{2(14h-24)^5}$. It is well known that a graph of treewidth d is $d + 1$ colorable. Let $v \in S$ and let S_v be its neighbors in $G \setminus S$. We first show that $|S_v| \le (t - 1)(d + 1)$. Consider the graph $G^* = G[S_v]$. Since $\mathbf{tw}(G \setminus S) \le d$ we have that $\mathbf{tw}(G^*) \le d$ and hence G^* is $d + 1$ colorable. Fix a coloring κ of G^* with $d + 1$ colors and let η be the size of the largest color class. Clearly $\eta \ge (|S_v|/d + 1)$. Since each color class is an independent set, we have that $\eta \le (t - 1)$, else we will get $K_{1,t}$ as an induced subgraph in G. This implies that $|S_v| \le (t - 1)(d + 1)$. Since v was an arbitrary vertex of S, we have that $\sum_{v \in S} |S_v| \le \sum_{v \in S} (t - 1)(d + 1) \le |S| \cdot g(h,t)$. Here $g(h,t) = (t - 1)(20^{2(14h-24)^5} + 1)$. Finally the observation that $N(S) = \cup_{v \in S} S_v$, yields the result. □

Now we are ready to prove Theorem 1.

Proof of Theorem 1. Let (G,k) be an instance of $\mathcal{F}$-DELETION and h be the size of a smallest planar graph in the obstruction set $\mathcal{F}$. We first apply Theorem 3 (to be proved in next section), an approximation algorithm for $\mathcal{F}$-DELETION with factor $O(\log^{3/2} OPT)$, and obtain a set X such that $G \setminus X$ contains no graph in $\mathcal{F}$ as a minor. If the size of the set X is more than $O(k \log^{3/2} k)$ then we return that (G,k) is a NO-instance to $\mathcal{F}$-DELETION. This is justified by the approximation guarantee provided by the Theorem 3.

Let d denote the treewidth of the graph after the removal of X, that is, $d := \mathbf{tw}(G \setminus S)$. Now we obtain the kernel in two phases: we first apply the protrusion rule selectively (Lemma 2) and get a polynomial kernel. Then, we apply the protrusion rule exhaustively on the obtained kernel to get a smaller kernel. This is done in order to reduce the running time complexity of the kernelization algorithm. To obtain the kernel we follow the following steps.

Applying the Protrusion Rule. By Lemma 1, $d \le 20^{2(14h-24)^5}$. We apply Lemma 3 and obtain a $2(d+1)$-protrusion Y of G of size at least $\frac{|V(G')|-|X|}{4|N(X)|+1}$. By Lemma 5, $\mathcal{F}$-DELETION has finite integer index. Let $\gamma : \mathbb{N} \to \mathbb{N}$ be the function defined in Lemma 2. If $\frac{|V(G')|-|X|}{4|N(X)|+1} \ge \gamma(2d + 1)$, then using Lemma 2 we replace the $2(d + 1)$-protrusion Y in G and obtain an instance $(G^*, k*)$ such that $|V(G^*)| < |V(G)|$, $k^* \le k$, and (G^*, k^*) is a YES-instance of $\mathcal{F}$-DELETION if and only if (G,k) is a YES-instance of $\mathcal{F}$-DELETION. Recall that G^* also excludes $K_{1,t}$ as an induced subgraph.

Let (G^*, k^*) be a reduced instance with hitting set X. In other words, there is no $(2d + 2)$-protrusion of size $\gamma(2d + 2)$ in $G^* \setminus X$, and Protrusion Rule no longer applies. We claim that the number of vertices

in this graph is bounded by $O(k \log^{3/2} k)$. Indeed, since we cannot apply the Protrusion Rule, we have that $\frac{|V(G^*)|-|X|}{4|N(X)|+1} \leq \gamma(2d+2)$. Because $k^* \leq k$, we have that

$$|V(G^*)| \leq \gamma(2d+2)(4|N(X)|+1) + |X|.$$

By Lemma 6, $|N(X)| \leq g(h,d) \cdot |X|$ and thus

$$|V(G^*)| = O(\gamma(2d+2) \cdot k \log^{3/2} k) = O(k \log^{3/2} k).$$

This gives us a polynomial time algorithm that returns a vertex kernel of size $O(k \log^{3/2} k)$.

Now we give a kernel of smaller size. We would like to replace every large $(2d+2)$-protrusion in graph by a smaller one. We find a $(2d+2)$-protrusion Y of size at least $\gamma(2d+2)$ by guessing the boundary $\partial(Y)$ of size at most $2d+2$. This could be performed in time $k^{O(d)}$. So let (G^*, k^*) be the reduced instance on which we cannot apply the Protrusion Rule. If G is a YES-instance then there is a $\mathcal{F}$-hitting set X of size at most k such that $\mathbf{tw}(G \setminus X) \leq d$. Now applying the analysis above with this X yields that $|V(G^*)| = O(k)$. Hence if the number of vertices in the reduced instance G^*, to which we can not apply the Protrusion Rule, is more than $O(k)$ then we return that G is a NO-instance. This concludes the proof of the theorem. $\qquad\qquad\square$

Corollary 1. *p*-FEEDBACK VERTEX SET, *p*-DIAMOND HITTING SET, *p*-PATHWIDTH ONE DELETION SET, *p*-OUTERPLANAR DELETION SET *admit linear vertex kernel on graphs excluding $K_{1,t}$ as an induced subgraph.*

The methodology used in proving Theorem 1 is not limited to $\mathcal{F}$-DELETION. For example, it is possible to obtain an $O(k \log k)$ vertex kernel on $K_{1,t}$-free graphs for *p*-DISJOINT CYCLE PACKING, which is for a given graph G and positive integer k to determine if there are k vertex disjoint cycles in G. It is iteresting to note that *p*-DISJOINT CYCLE PACKING does not admit a polynomial kernel on general graphs [13]. For our kernelization algorithm, we use the following Erdős-Pósa property [27]: given a positive integer ℓ every graph G either has ℓ vertex disjoint cycles or there exists a set $S \subseteq V(G)$ of size at most $O(\ell \log \ell)$ such that $G \setminus S$ is a forest. So given a graph G and positive integer k, we first apply the factor 2 approximation algorithm given in [5] and obtain a set S such that $G \setminus S$ is a forest. If the size of S is more than $O(k \log k)$ then we return that G has k vertex disjoint cycles. Else, we use the fact that *p*-DISJOINT CYCLE PACKING [12] has finite integer index and apply the protrusion reduction rule in $G \setminus S$ to obtain an equivalent instance (G^*, k^*), as in Theorem 1. The analysis for kernel size used in the proof of Theorem 1 together with the observation that $\mathbf{tw}(G \setminus S) \leq 1$ shows that if (G, k) is an yes instance then the size of $V(G^*)$ is at most $O(k \log k)$.

Corollary 2. *p*-DISJOINT CYCLE PACKING *has $O(k \log k)$ vertex kernel on graphs excluding $K_{1,t}$ as an induced graph.*

Next, we extend the methods used in this section for obtaining kernels for $\mathcal{F}$-DELETION on graphs excluding $K_{1,t}$ as an induced graph to all graphs, though for restricted $\mathcal{F}$ — we consider the families $\mathcal{F}$ that contain θ_c. However, to achieve this we need a polynomial time approximation algorithm with a factor polynomial in optimum size and not depending on the input size. For example, an approximation algorithm with factor $O(\log n)$ would not serve our purpose. Here, we obtain an approximation algorithm for $\mathcal{F}$-DELETION with a factor $O(\log^{3/2} OPT)$ whenever the finite obstruction set $\mathcal{F}$ contains a planar graph. Here OPT is the size of a minimum $\mathcal{F}$-hitting set. This immediately implies a factor $O(\log^{3/2} n)$ algorithm for all the problems that can categorized by $\mathcal{F}$-DELETION. We believe this result has its own significance and is of independent interest.

4   An approximation algorithm for finding a $\mathcal{F}$–hitting set

In this section, we present an $O(\log^{3/2} OPT)$-approximation algorithm for the $\mathcal{F}$-DELETION problem when the finite obstruction set $\mathcal{F}$ contains at least one planar graph.

Algorithm 1 Hɪᴛ-Sᴇᴛ-I-(G)

1: **if tw**$(G) \leq d$ **then**
2:     Find a minimum $\mathcal{F}$-hitting set Y of G and return Y.
3: **end if**
4: Compute an approximate tree decomposition $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ of width ℓ.
5: **if** $\ell > (k + d)\sqrt{\log(k + d)}$, where d is as in Lemma 1 **then**
6:     Return that G does not have $\mathcal{F}$–hitting set of size at most k.
7: **end if**
8: Convert $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ to a nice tree decomposition of the same width.
9: Find a partitioning of vertex set $V(G)$ into V_1, V_2 and X (a bag corresponding to a node in T) such that **tw**$(G[V_1]) = d$ as described in the proof.
10: Return $\left(X \bigcup \text{Hɪᴛ-Sᴇᴛ-I-}(G[V_1]) \bigcup \text{Hɪᴛ-Sᴇᴛ-I-}(G[V_2]) \right)$.

Lemma 7. *There is a polynomial time algorithm that, given a graph G and a positive integer k, either reports that G has no $\mathcal{F}$-hitting set of size at most k or finds a $\mathcal{F}$-hitting set of size at most $O(k \log^{3/2} k)$.*

Proof. We begin by introducing some definitions that will be useful for describing our algorithms. First is the notion of a *good labeling function*. Given a nice tree decomposition $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ of a graph G, a function $g : V(T) \to \mathbb{N}$ is called a *good labeling function* if it satisfies the following properties:

- if t is a base node then $g(t) = 0$;
- if t is an introduce node, then $g(t) = g(s)$, where s is the child of t;
- if t is a join node, then $g(t) = g(s_1) + g(s_2)$, where s_1 and s_2 are the children of t; and
- if t is a forget node, then $g(t) \in \{g(s), g(s) + 1\}$, where s is the child of t.

A *max labeling function g* is defined analogously to a good labeling function, the only difference being that for a join node t, we have the condition $g(t) = \max\{g(s_1), g(s_2)\}$. We now turn to the approximation algorithm.

Our algorithm has two phases. In the first phase we obtain a $\mathcal{F}$-hitting set of size $O(k^2 \sqrt{\log k})$ and in the second phase we use the hitting set obtained in the first phase to get a $\mathcal{F}$-hitting set of size $O(k \log^{3/2} k)$. The second phase could be thought of as "bootstrapping" where one uses an initial solution to a problem to obtain a better solution.

By assumption we know that $\mathcal{F}$ contains at least one planar graph. Let h be the number of vertices in the smallest planar graph H contained in $\mathcal{F}$. By a result of Robertson et al. [45], H is a minor of the $(t \times t)$-grid, where $t = 14h - 24$. Robertson et al. [45] have also shown that any graph with treewidth greater than 20^{2t^5} contains a $t \times t$ grid as a minor. In the algorithm we set $d = 20^{2t^5}$.

We begin by describing the first phase of the algorithm, see Algorithm 1. We start by checking whether a graph G has treewidth at most d (the first step of the algorithm) using the linear time algorithm of Bodlaender [8]. If **tw**$(G) \leq d$ then we find an optimum $\mathcal{F}$-hitting set of G in linear time using a modification of Lemma 9. If the treewidth of the input graph is more than d then we find an approximate tree decomposition of width ℓ using an algorithm of Feige et al. [28] such that **tw**$(G) \leq \ell \leq d' \textbf{tw}(G) \sqrt{\log \textbf{tw}(G)}$, where d' is a fixed constant.

If $\ell > (k + d)d' \sqrt{\log(k + d)}$, then by Lemma 1, we know that the size of a minimum $\mathcal{F}$-hitting set of G is at least $(k + 1)$. Hence from now onwards we assume that **tw**$(G) \leq \ell \leq (k + d)d' \sqrt{\log(k + d)}$. In the next step we convert the given tree decomposition to a nice tree decomposition of the same width in linear time [36]. Given a nice tree decomposition $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ of G, we compute a *partial* function $\beta : V(T) \to \mathbb{N}$, defined as $\beta(t) = \textbf{tw}(H_t)$. Observe that β is a max labeling function. We compute β in a bottom up fashion starting from base nodes and moving towards the root. We stop this computation the first time that we find a node t such that $\beta(t) = \textbf{tw}(H_t) = d$. Let $V_1 = V(H_t)$, $V_2 = V(G) \setminus V_1 \setminus X_t$ and $X = X_t$. After this we recursively solve the problem on the graphs induced on V_1 and V_2.

11

Algorithm 2 HIT-SET-II-(G, Z)

1: **if** $\mathbf{tw}(G) \leq d$ **then**
2:    Find a minimum $\mathcal{F}$-hitting set Y of G and Return Y.
3: **end if**
4: Compute an approximate tree decomposition $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ of width ℓ.
5: Convert it to a nice tree decomposition of G. Now compute the function $\mu : V(T) \to \mathbb{N}$, defined as
    follows: $\mu(t) = |V(H_t) \cap Z|$.
6: **if** $(\mu(r) = 0)$ **then**
7:    Return ϕ.
8: **else**
9:    Find the partitioning of the vertex set $V(G)$ into V_1, V_2 and X (a bag corresponding to a node in
       T) as described in Cases 1 and 2 of the proof of Theorem 7.
10: **end if**
11: Return $\left(X \bigcup \text{HIT-SET-II-}(G[V_1], Z) \bigcup \text{HIT-SET-II-}(G[V_2], Z) \right)$.

Let us assume that G has a $\mathcal{F}$-hitting set of size at most k. We show that in this case the size of the hitting set returned by the algorithm can be bounded by $O(k^2 \sqrt{\log k})$. The above recursive procedure can be thought of as a rooted binary tree $\mathcal{T}$ where at each non-leaf node of the tree the algorithm makes two recursive calls. We will assume that the left child of a node of $\mathcal{T}$ corresponds to the graph induced on V_1 such that the treewidth of $G[V_1]$ is d. Assuming that the root is at depth 0 we show that the depth of $\mathcal{T}$ is bounded by k. Let $P = a_0 a_1 \cdots a_q$ be a longest path from the root to a leaf and let G_i be the graph associated with the node a_i. Observe that for every $i \in \{0, \dots, q-1\}$, a_i has a left child, or else a_i cannot be a non-leaf node of $\mathcal{T}$. Let the graph associated with the left child of a_i, $i \in \{0, \dots, q-1\}$, be denoted by H_i. Observe that for every $0 \leq i < j \leq q-1$, $V(H_i) \cap V(H_j) = \emptyset$ and $\mathbf{tw}(H_i) = d$. This implies that every H_i has at least one H minor model and all of these are vertex-disjoint. This implies that $q \leq k$ and hence the depth of $\mathcal{T}$ is bounded by k.

Let us look at all the subproblems at depth i in the recursion tree $\mathcal{T}$. Suppose at depth i the induced subgraphs associated with these subproblems are $G[V_i]$, $i \in [\tau]$, where τ is some positive integer. Then observe that for every $i, j \in [\tau]$ and $i \neq j$, we have that $V_i \cap V_j = \emptyset$, there is no edge (u, v) such that $u \in V_i$, $v \in V_j$, and hence $\sum_{i=1}^{\tau} k_i \leq k$, where k_i is the size of the minimum $\mathcal{F}$–hitting set of $G[V_i]$. Furthermore the number of instances at depth i such that it has at least one H minor model and hence contributes to the hitting set is at most k. Now Lemma 1 together with the factor $d' \sqrt{\log \mathbf{tw}(G)}$ approximation algorithm of Feige et al. [28] implies that the treewidth of every instance is upper bounded by $(k_i + d)d' \sqrt{\log(k_i + d)}$, where k_i is the size of the minimum $\mathcal{F}$–hitting set of $G[V_i]$. Hence the total size of the union of sets added to our hitting set at depth i is at most

$$\sum_{i=1}^{\tau} \chi(i)(k_i + d)d' \sqrt{\log(k_i + d)} \leq d'(k + d) \sqrt{\log(k + d)}.$$

Here $\chi(i)$ is 1 if $G[V_i]$ contains at least one H minor model and is 0 otherwise. We have shown that for each i the size of the union of the sets added to the hitting set is at most $d'(k + d) \sqrt{\log(k + d)}$. This together with the fact that the depth is at most k implies that the size of the $\mathcal{F}$-hitting set is at most $O(k^2 \sqrt{\log k})$. Hence if the size of the hitting set returned by the algorithm is more than $d'(k + d)k \sqrt{\log(k + d)}$ then we return that G has at no $\mathcal{F}$-hitting set of size at most k. Hence when we move to the second phase we assume that we have a hitting set of size $O(k^2 \sqrt{\log k})$. This concludes the description of the first phase of the algorithm.

Now we describe the second phase of the algorithm. Here we are given the hitting set Z of size $O(k^2 \sqrt{\log k})$ obtained from the first phase of the algorithm. The algorithm is given in Algorithm 2. The new algorithm essentially uses Z to define a good labeling function μ which enables us to argue that the depth of recursion is upper bounded by $O(\log |Z|)$. In particular, consider the function $\mu : V(T) \to \mathbb{N}$,

defined as follows: $\mu(t) = |V(H_t) \cap Z|$. Let $k' := \mu(r)$, where r is the node corresponding to the root of a fixed nice tree decomposition of G.

Let $t \in V(T)$ be the node where $\mu(t) > 2k'/3$ and for each child t' of t, $\mu(t') \leq 2k'/3$. Since μ is a good labeling function, it is easy to see that this node exists and is unique provided that $k' > 0$. Moreover, observe that t could either be a forget node or a join node. We distinguish these two cases.

- *Case 1.* If t is a forget node, we set $V_1 = V(H_{t'})$ and $V_2 = V(G) \setminus (V_1 \cup X_{t'})$ and observe that $P_{\theta_c}(G[V_i]) \leq \lfloor 2k'/3 \rfloor, i = 1, 2$. Also we set $X = X_{t'}$.

- *Case 2.* If t is a join node with children t_1 and t_2, we have that $\mu(t_i) \leq 2k'/3, i = 1, 2$. However, as $\mu(t_1) + \mu(t_2) > 2k'/3$, we also have that either $\mu(t_1) \geq k'/3$ or $\mu(t_2) \geq k'/3$. Without loss of generality we assume that $\mu(t_1) \geq k'/3$ and we set $V_1 = V(H_{t_1})$, $V_2 = V(G) \setminus (V_1 \cup X_{t_1})$ and $X = X_{t_1}$.

Now we argue that if G has a $\mathcal{F}$–hitting set of size at most k then then the size of the hitting set returned by the algorithm is upper bounded by $O(k \log^{3/2} k)$. As in the first phase we can argue that the size of the union of the sets added to the hitting set in the subproblems at depth i is at most $d'(k + d) \sqrt{\log(k + d)}$. Observe that the recursive procedure in Algorithm 2 is such that the value of the function $\mu()$ drops by at least a constant fraction at every level of recursion. This implies that the depth of recursion is upper bounded by $O(\log |Z|) = O(\log k)$. Hence the size of the hitting set returned by the algorithm is upper bounded by $O(k \log^{3/2} k)$ whenever G has a $\mathcal{F}$–hitting set of size at most k. Thus if the size of the hitting set returned by Hit-Set-II-(G, Z) is more than $d'(k + d) \sqrt{\log^{3/2}(k + d)}$, we return that G does not have a $\mathcal{F}$–hitting set of size at most k. This concludes the proof. $\qquad \square$

Proof of Theorem 3. Given a graph G on n vertices, let k be the minimum positive integer in $\{1, \ldots, n\}$ such that Lemma 7 returns a $\mathcal{F}$-hitting set S when applied on (G, k). We return this S as an approximate solution. By our choice of k we know that G does not have $\mathcal{F}$-hitting set of size at most $k - 1$ and hence $OPT \geq k$. This implies that the size of S returned by Lemma 7 is at most $O(k \log^{3/2} k) = O(OPT \log^{3/2} OPT)$. This concludes the proof. $\qquad \square$

We now define a generic problem. Let η be a fixed constant. In the Treewidth η-Deletion Set problem, we are given an input graph G and the objective is to delete minimum number of vertices from a graph such that the resulting graph has treewidth at most η. For an example Treewidth 1-Deletion Set is simply the Feedback vertex set problem. We obtain the following corollary of Theorem 3.

Corollary 3. Feedback Vertex Set, Diamond Hitting Set, Pathwidth One Deletion Set, Outerplanar Deletion Set *and* Treewidth η-Deletion Set *admit a factor* $O(\log^{3/2} n)$ *approximation algorithm on general undirected graphs.*

5 Kernelization for p-θ_c-Free-Deletion

In this section we obtain a polynomial kernel for p-θ_c-Free-Deletion on general graphs. To obtain our kernelization algorithm we not only need approximation algorithm presented in the last section but also a variation of classical Hall's theorem. We first present this combinatorial tool and other auxiliary results that we make use of.

5.1 Combinatorial Lemma and some Linear-Time Subroutines.

We need a variation of the celebrated Hall's Theorem, which we call the q–Expansion Lemma. The q–Expansion Lemma is a generalization of a result due to Thomassé [46, Theorem 2.3], and captures a certain property of neighborhood sets in graphs that implicitly has been used by several authors to obtain polynomial kernels for many graph problems. For $q = 1$, the application of this lemma is exactly the well-known Crown Reduction Rule [1].

The Expansion Lemma. Consider a bipartite graph G with vertex bipartition $A \uplus B$. Given subsets $S \subseteq A$ and $T \subseteq B$, we say that S has $|S|$ q-stars in T if to every $x \in S$ we can associate a subset $F_x \subseteq N(x) \cap T$ such that (a) for all $x \in S$, $|F_x| = q$; (b) for any pair of vertices $x, y \in S$, $F_x \cap F_y = \emptyset$. Observe that if S has $|S|$ q-stars in T then every vertex x in S could be thought of as the center of a star with its q leaves in T, with all these stars being vertex-disjoint. Further, a collection of $|S|$ q-stars is also a family of q edge-disjoint matchings, each saturating S. We use the following result in our kernelization algorithm to bound the degrees of vertices.

Lemma 8. [The q–Expansion Lemma] *Let q be a positive integer, and let m be the size of the maximum matching in a bipartite graph G with vertex bipartition $A \uplus B$. If $|B| > mq$, and there are no isolated vertices in B, then there exist nonempty vertex sets $S \subseteq A, T \subseteq B$ such that S has $|S|$ q-stars in T and no vertex in T has a neighbor outside S. Furthermore, the sets S, T can be found in time polynomial in the size of G.*

Proof. Consider the graph $H = (X \uplus B, E)$ obtained from $G = (A \uplus B, E)$ by adding $(q - 1)$ copies of all the vertices in A, and giving all copies of a vertex v the same neighborhood in B as v. Let M be a maximum matching in H. In further discussions, vertices are saturated and unsaturated with respect to this fixed matching M.

Let U_X be the vertices in X that are unsaturated, and R_X be those that are reachable from U_X via alternating paths. We let $S_A = X \setminus (U_X \cup R_X)$. Let U_B be the set of unsaturated vertices in B, and let S' denote the set of partners of S_A in the matching M, that is, $S' = \{x \in B \mid \{u, x\} \in M \text{ and } u \in S_A\}$. Let $T = S' \cup U_B$ (see Figure 2).
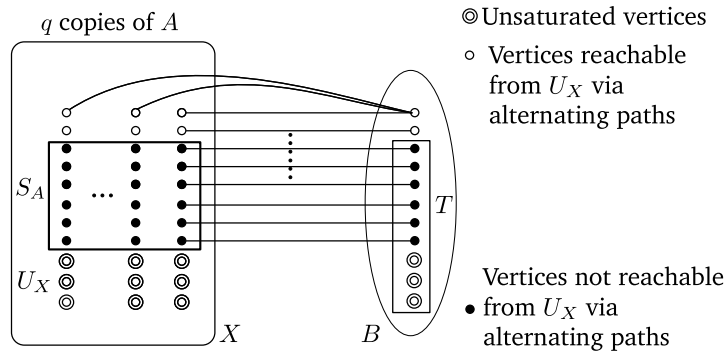

Figure 2: The construction used in the proof of the q–Expansion Lemma

For every $v \in A$, let $C(v)$ be the set of all copies of v (including v). We claim that either $C(v) \cap S_A = C(v)$, or $C(v) \cap S_A = \emptyset$. Suppose that $v \in S_A$ but a copy of v, say u, is in U_X. Let $\{v, w\} \in M$. Then v is reachable from u because $\{u, w\} \in E(H)$, and hence w is not unsaturated in M, contradicting the assumption that $w \in U_X$. In the case when $v \in S_A$ but a copy of u is in R_X, let $\{w, u\}$ be the last edge on some alternating path from U_X to u. Since $\{w, v\} \in E(H)$, we have that there is also an alternating path from U_X to v, contradicting the fact that $v \in S_A$. Let $S = \{v \in A | C(v) \subseteq S_A\}$. Then the subgraph $G[S \cup T]$ contains q edge-disjoint matchings, each of which saturates S in G — this is because in H, M saturates each copy of $v \in S$ separately.

If no vertex in T has a neighbor outside S_A in H, then from the construction no vertex in T has a neighbor outside S in G. We now prove that no vertex in T has a neighbor outside S_A in H. For the purpose of contradiction, let us assume that for some $v \in T$, $u \in N(v)$, but $u \notin S_A$. First, consider the case when $v \in S'$. Suppose $u \in R_X$. We know that $u \in R_X$ because there is some unsaturated vertex (say w) that is connected by an alternating path to u. This path can be extended to a path to v using the edge $\{u, v\}$, and can be further extended to v', where $\{v, v'\} \in M$. However, $v' \in S_A$, and by construction, there is no path from $w \in U_X$ to v', a contradiction. If $u \in U_X$, then we arrive at a contradiction along the same

lines (in fact, the paths from w to a vertex in S will be of length two in this case). Now consider the case when $v \in U_B$. Again, we may arrive at u from some $w \in U_X$ (if $u \in R_X$) or $\{u, w\}$ is an independent edge outside M (if $u \in U_X$). In both cases, we have an *augmenting* path, contradicting the fact that M is a maximum matching. This proves the claim that no vertex in T has a neighbor outside S_A in H.

We now note that the set S_A obtained in the first part of the proof is non-empty. First, observe that since $|B| > mq$, the set U_B of unsaturated vertices of B in H is non-empty. Further, by the assumption that B admits no isolated vertices, the neighbors of U_B form a non-trivial subset of A. Now recall that the neighbors of T lie in S_A, and $U_B \subseteq T$. So in particular, the neighbors of U_B lie in S_A and thus S_A is non-empty. This concludes the proof.

$\square$

We will need the following proposition for the proof of next observation. Its proof follows from definitions.

Proposition 1. *For any $c \in \mathbb{N}$, a subgraph M of graph G is a minimal minor-model of θ_c in G if and only if M consists of two trees, say T_1 and T_2, and a set S of c edges, each of which has one end vertex in T_1 and the other in T_2.*

Observation 1. *For $c \geq 2$, any minimal θ_c minor-model M of a graph G is a connected subgraph of G, and does not contain a vertex whose degree in M is less than 2, or a vertex whose deletion from M results in a disconnected graph (a cut vertex of M).*

Proof. From Proposition 1, whose terminology we use in this proof, M is connected and contains no isolated vertex. Suppose x is a vertex of degree exactly one in M. Then x is a leaf node in one of the two trees in M, say T_1, and no edge in S is incident on x. Removing x from T_1 results in a smaller θ_c minor-model, contradicting the minimality of M. It follows that every vertex of M has degree at least two.

Now suppose x is a cut vertex in M which belongs to, say, the tree T_1. Let $T_1^1, T_1^2, \ldots, T_1^l$ be the subtrees of T_1 obtained when x is deleted from T_1. Let M' be the graph obtained by deleting x from M. If $l > 0$, then each T_1^i has a leaf node, which, by the above argument, has at least one neighbor in T_2. If $l = 0$, then $M' = T_2$. Thus M' is connected in all cases, and so x is not a cut vertex, a contradiction. $\square$

The following well known result states that every optimization problem expressible in MSO has a linear time algorithm on graphs of bounded treewidth.

Proposition 2 ([4, 8, 14, 20, 22]). *Let ϕ be a property that is expressible in Monadic Second Order Logic. For any fixed positive integer t, there is an algorithm that, given a graph G of treewidth at most t as input, finds a largest (alternatively, smallest) set S of vertices of G that satisfies ϕ in time $f(t, |\phi|)|V(G)|$.*

Proposition 2 together with MSO formulations 2 and 3 given in Section 2.6 implies the following lemma.

Lemma 9. *Let G be a graph on n vertices and v a vertex of G. Given a tree decomposition of width $t \in O(1)$ of G, we can, in $O(n)$ time, find both (1) a smallest set $S \subseteq V$ of vertices of G such that the graph $G \setminus S$ does not contain θ_c as a minor, and (2) a largest collection $\{M_1, M_2, \ldots, M_l\}$ of θ_c minor models of G such that for $1 \leq i < j \leq l, (V(M_i) \cap V(M_j)) = \{v\}$.*

Now we describe the reduction rules used by the kernelization algorithm. In contrast to the reduction rules employed by most known kernelization algorithms, these rules cannot always be applied on general graphs in polynomial time. Hence the algorithm does not proceed by applying these rules exhaustively, as is typical in kernelization programs. We describe how to arrive at situations where these rules can in fact be applied in polynomial time, and prove that even this selective application of rules results in a kernel of size polynomial in the parameter k.

5.2 Bounding the Maximum Degree of a Graph

Now we present a set of reduction rules which, given an input instance (G, k) of p-θ_c-Free-Deletion, obtains an equivalent instance (G', k') where $k' \leq k$ and the maximum degree of G' is at most a polynomial in k. In the sequel a vertex v is *irrelevant* if it is not a part of any θ_c minor model, and is *relevant* otherwise. For each rule below, the input instance is (G, k).

Reduction Rule 1 (Irrelevant Vertex Rule). *Delete all irrelevant vertices in G.*

Given a graph G and a vertex $v \in V(G)$, an *ℓ-flower passing through v* is a set of ℓ different θ_c minor-models in G, each containing v and no two sharing any vertex other than v.

Reduction Rule 2 (Flower Rule). *If a $(k + 1)$-flower passes through a vertex v of G, then include v in the solution and remove it from G to obtain the equivalent instance $(G \setminus \{v\}, (k - 1))$.*

The argument for the soundness of these reduction rules is simple and is hence omitted. One can test whether a particular vertex v is part of any minimal minor-model corresponding to θ_c using the rooted minor testing algorithm of Robertson and Seymour [44]. It is not clear, however, that one might check whether a vertex is a part of $(k+1)$-θ_c flower in polynomial time. Hence we defer the application of these rules and apply them only when the vertices are "evidently" irrelevant or finding a flower can be solved in polynomial time. Now we state an auxiliary lemma which will be useful in bounding the maximum degree of the graph.

Lemma 10. *Let G be a n-vertex graph containing θ_c as a minor and v be a vertex such that $G' = G \setminus \{v\}$ does not contain θ_c as a minor and the maximum size of a flower containing v is at most k. Then there exists a set T_v of size $O(k)$ such that $v \notin T_v$ and $G \setminus T_v$ does not contain θ_c as a minor. Moreover we can find the set T_v in polynomial time.*

Proof. We first bound the treewidth of G'. Robertson, Seymour and Thomas [45] have shown that any graph with treewidth greater than 20^{2c^5} contains a $c \times c$ grid, and hence θ_c, as a minor. This implies that for a fixed c, $\mathbf{tw}(G') \leq 20^{2c^5} = O(1)$. Now we show the existence of a T_v of the desired kind. Recall the algorithm used to show the existence of a θ_c hitting set for a graph described in Algorithm 2. We use the same algorithm to construct the desired T_v. Let $F_{\theta_c}(G)$ denote the size of the maximum flower passing through v in G. Consider a nice tree decomposition $(T, X = \{X_t\}_{t \in V(T)})$ of G' of width at most $\mathbf{tw}(G')$. We define the function $\mu(t) := F_{\theta_c}(G[V(H_t) \cup \{v\}])$. It is easy to see that μ is a good labeling function, and can be computed in polynomial time due to Lemma 9. Observe that $\mu(r) \leq k$, where r is the root node of the tree decomposition. Let $\mathcal{S}(G', k)$ denote the size of the hitting set returned by the algorithm. Thus the size of the hitting set returned by the algorithm Hit-Set-II (Algorithm 2) is governed by the following recurrence:

$$\mathcal{S}(G', k) \leq \max_{1/3 \leq \alpha \leq 2/3} \Big\{ \mathcal{S}(G[V_1], \alpha k) + \mathcal{S}(G[V_2], (1 - \alpha)k) + O(1) \Big\}.$$

Using Akra-Bazzi [2] it follows that the above recurrence solves to $O(k)$. This implies that there exists a set T_v of size $O(k)$ such that $v \notin T_v$ and $G \setminus T_v$ does not contain θ_c as a minor. We now proceed to find an optimal hitting set in G avoiding v. To make the algorithm Hit-Set-II run in polynomial time we only need to find the tree decomposition and compute the function $\mu()$ in polynomial time. Since $\mathbf{tw}(G) = O(1)$, we can find the desired tree decomposition of G or one of its subgraphs in linear time using the algorithm of Bodlaender [8]. Similarly we can compute a flower of the maximum size using Lemma 9 in linear time. Hence the function $\mu()$ can also be computed in polynomial time. This concludes the proof of the lemma. $\qquad\square$

Flowers, Expansion and the Maximum Degree. Now we are ready to prove the lemma which bounds the maximum degree of the instance.

Lemma 11. *There exists a polynomial time algorithm that, given an instance (G, k) of p-θ_c-*Deletion *returns an equivalent instance (G', k') such that $k' \leq k$ and that the maximum degree of G' is $O(k \log^{3/2} k)$. Moreover it also returns a θ_c-hitting set of G' of size $O(k \log^{3/2} k)$.*

Proof. Given an instance (G, k) of p-θ_c-Deletion, we first apply Lemma 7 on (G, k). The polynomial time algorithm described in Lemma 7, given a graph G and a positive integer k either reports that G has no θ_c-hitting set of size at most k, or finds a θ_c-hitting set of size at most $k^* = O(k \log^{3/2} k)$. If the algorithm reports that G has has no θ_c-hitting set of size at most k, then we return that (G, k) is a NO-instance to p-θ_c-Deletion. So we assume that we have a hitting set $\mathcal{S}$ of size k^*. Now we proceed with the following two rules.

Selective Flower Rule. To apply the Flower Rule selectively we use $\mathcal{S}$, the θ_c-hitting set. For a vertex $v \in \mathcal{S}$ let $\mathcal{S}_v := \mathcal{S} \setminus \{v\}$ and let $G_v := G \setminus \mathcal{S}_v$. By a result of Robertson et. al. [45] we know that any graph of treewidth greater than 20^{2c^5} contains a $c \times c$ grid, and hence θ_c, as a minor. Since deleting v from G_v makes it θ_c-minor-free, $\mathbf{tw}(G_v) \leq 20^{2c^5} + 1 = O(1)$. Now by Lemma 9, we find in linear time the size of the largest flower centered at v, in G_v. If for any vertex $v \in \mathcal{S}$ the size of the flower in G_v is at least $k + 1$, we apply the Flower Rule and get an equivalent instance $(G \leftarrow G \setminus \{v\}, k \leftarrow k - 1)$. Furthermore, we set $\mathcal{S} := \mathcal{S} \setminus \{v\}$. We apply the Flower Rule selectively until no longer possible. We abuse notation and continue to use (G, k) to refer to the instance that is reduced with respect to exhaustive application of the Selective Flower Rule. Thus, for every vertex $v \in \mathcal{S}$ the size of any flower passing through v in G_v is at most k.

Now we describe how to find, for a given $v \in V(G)$, a hitting set $H_v \subseteq V(G) \setminus \{v\}$ for all minor-models of θ_c that contain v. Notice that this hitting set is required to *exclude* v, so H_v cannot be the trivial hitting set $\{v\}$. If $v \notin \mathcal{S}$, then $H_v = \mathcal{S}$. On the other hand, suppose $v \in \mathcal{S}$. Since the maximum size of a flower containing v in the graph G_v is at most k by Lemma 10, we can find a set T_v of size $O(k)$ that does not contain v and hits all the θ_c minor-models passing through v in G_v. Hence in this case we set $H_v = \mathcal{S}_v \cup T_v$ (See Figure 3.). We denote $|H_v|$ by h_v. Notice that H_v is defined algorithmically, that is, there could be many small hitting sets in $V(G) \setminus \{v\}$ hitting all minor-models containing v, and H_v is one of them.
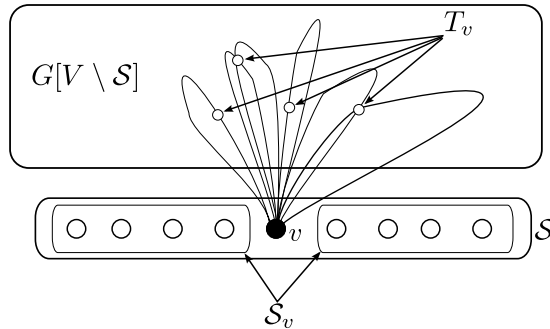

Figure 3: The hitting set in Selective Flower Rule

q-expansion Rule with $q = c$. Given an instance (G, k), $\mathcal{S}$, and a family of sets H_v, we show that if there is a vertex v with degree more than $ch_v + c(c - 1)h_v$, then we can reduce its degree to at most $ch_v + c(c - 1)h_v$ by repeatedly applying the q–Expansion Lemma with $q = c$. Observe that for every vertex v the set H_v is also a θ_c hitting set for G, that is, H_v hits *all* minor-models of θ_c in G. Consider the graph $G \setminus H_v$. Let the components of this graph that contain a neighbor of v be $C_1, C_2, \ldots, C_r$. Note

that v cannot have more than $(c-1)$ neighbors into any component, else contracting the component will form a θ_c minor and will contradict the fact that H_v hits all the θ_c minors. Also note that none of the C_i's can contain a minor model of θ_c.

We say that a component C_i is adjacent to H_v if there exists a vertex $u \in C_i$ and $w \in H_v$ such that $(u, w) \in E(G)$. Next we show that vertices in components that are not adjacent to H_v are irrelevant in G. Recall a vertex is irrelevant if there is no minimal minor model of θ_c that contains it. Consider a vertex u in a component C that is not adjacent to H_v. Since $G[V(C) \cup \{v\}]$ does not contain any θ_c minor we have that if u is a part of a minimal minor model $M \subseteq G$, then $v \in M$ and also there exists a vertex $u' \in M$ such that $u' \notin C \cup \{v\}$. Then the removal of v disconnects u from u' in M, a contradiction to Observation 1 that for $c \geq 2$, any minimal θ_c minor model M of a graph G does not contain a cut vertex. Applying the Irrelevant Vertex Rule to the vertices in all such components leaves us with a new set of components $D_1, D_2, \ldots, D_s$, such that for every i, in D_i, there is at least one vertex that is adjacent to a vertex in H_v.

As before, we continue to use G to refer to the graph obtained after the Irrelevant Vertex Rule has been applied in the context described above. We also update the sets H_v for $v \in V(G)$ by deleting all the vertices w from these sets those have been removed using Irrelevant Vertex Rule.

Now, consider a bipartite graph $\mathcal{G}$ with vertex bipartitions H_v and D. Here $D = \{d_1, \ldots, d_s\}$ contains a vertex d_i corresponding to each component D_i. We add an edge (v, d_i) if there is a vertex $w \in D_i$ such that $\{v, w\} \in E(G)$. Even though we start with a simple graph (graphs without parallel edges) it is possible that after applying reduction rules parallel edges may appear. However, throughout the algorithm, we ensure that the number of parallel edges between any pair of vertices is at most c. Now, v has at most ch_v edges to vertices in H_v. Since v has at most $(c-1)$ edges to each D_i, it follows that if $d(v) > ch_v + c(c-1)h_v$, then the number of components $|D|$ is more than ch_v. Now by applying q–Expansion Lemma with $q = c$, $A = H_v$, and $B = D$, we find a subset $S \subseteq H_v$ and $T \subseteq D$ such that S has $|S|$ c-stars in T and $N(T) = S$.

The reduction rule involves deleting edges of the form (v, u) for all $u \in D_i$, such that $d_i \in T$, and adding c edges between v and w for all $w \in S$. We add these edges only if they were not present before so that the number of egdes between any pair of vertices remains at most c. This completes the description of the q-expansion reduction rule with $q = c$. Let G_R be the graph obtained after applying the reduction rule. The following lemma shows the correctness of the rule.

Lemma 12. *Let G, S and v be as above and G_R be the graph obtained after applying the c-expansion rule. Then (G, k) is an yes instance of p-θ_c-DELETION if and only if (G_R, k) is an yes instance of p-θ_c-DELETION.*

Proof. We first show that if G_R has hitting set Z of size at most k, then the same hitting set Z hits all the minor-models of θ_c in G. Observe that either $v \in Z$ or $S \subseteq Z$. Suppose $v \in Z$, then observe that $G_R \setminus \{v\}$ is the same as $G \setminus \{v\}$. Therefore $Z \setminus \{v\}$, a hitting set of $G_R \setminus \{v\}$ is also a hitting set of $G \setminus \{v\}$. This shows that Z is a hitting set of size at most k of G. The case when $S \subseteq Z$ is similar.

To prove that a hitting set of size at most k in G implies a hitting set of size at most k in G_R, it suffices to prove that whenever there is a hitting set of size at most k, there also exists a hitting set of size at most k that contains either v or all of S. Consider a hitting set W that does not contain v, and omits at least one vertex from S. Note the $|S|$ c-stars in $\mathcal{G}[S \cup \mathcal{T}]$, along with v, correspond to minor-models of θ_c centered at v in G, vertex-disjoint except for v. Thus, such a hitting set must pick at least one vertex from one of the components. Let $\mathcal{D}$ be the collection of components D_i such that the (corresponding) vertex $d_i \in T$. Let X denote the set of all vertices of W that appeared in any $D_i \in \mathcal{D}$. Consider the hitting set W' obtained from W by removing X and adding S, that is, $W' := (W \setminus X) \cup S$.

We now argue that W' is also a hitting set of size at most k. Indeed, let S' be the set of vertices in S that do not already belong to W. Clearly, for *every* such vertex that W omitted, W must have had to pick distinct vertices from $\mathcal{D}$ to hit the θ_c minor-models formed by the corresponding c-stars. Formally, there exists a $X' \subseteq X$ such that there is a bijection between S' and X', implying that $|W'| \leq |W| \leq k$.

Finally, observe that W' must also hit all minor-models of θ_c in G. If not, there exists a minor-model M that contains some vertex $u \in X$. Hence, $u \in D_i$ for some i, and M contains some vertex in $H_v \setminus S$. However, v separates u from $H_v \setminus S$ in $G \setminus S$, contradicting Observation 1 that M does not contain a cut vertex. This concludes the proof. □

Observe that all edges that are added during the application of the q-expansion reduction rule have at least one end point in $\mathcal{S}$, and hence $\mathcal{S}$ remains a hitting set of G_R. We are now ready to summarize the algorithm that bounds the degree of the graph (see Algorithm 3).

Algorithm 3 BOUND-DEGREE$(G, k, \mathcal{S})$

1: Apply the Selective Flower Rule
2: **if** $\exists v \in V(G)$ such that $d(v) > ch_v + c(c-1)h_v$ **then**
3:     Apply the q-expansion reduction rule with $q = c$.
4: **else**
5:     Return $(G, k, \mathcal{S})$.
6: **end if**
7: Return BOUND-DEGREE$(G, k, \mathcal{S})$.

Let the instance output by Algorithm 3 be $(G', k', \mathcal{S})$. Clearly, in G', the degree of every vertex is at most $ch_v + c(c-1)h_v \leq O(k \log^{3/2} k)$. The routine also returns $\mathcal{S}$ — a θ_c-hitting set of G' of size at most $O(k \log^{3/2} k)$.

We now show that the algorithm runs in polynomial time. For $x \in V(G)$, let $v(x)$ be the number of neighbors of x to which x has fewer than c parallel edges. Observe that the application of q-expansion reduction rule never increases $v(x)$ for any vertex and decreases $v(x)$ for at least one vertex. The other rules delete vertices, which can never increase $v(x)$ for any vertex. This concludes the proof. □

5.3   Analysis and Kernel Size – Proof of Theorem 2

In this section we give the desired kernel for p-θ_c-FREE-DELETION.

Proof of Theorem 2. Let (G, k) be an instance to p-θ_c-FREE-DELETION. We first bound the maximum degree of the graph by applying Lemma 11 on (G, k). If Lemma 11 returns that (G, k) is a NO-instance to p-θ_c-FREE-DELETION then we return the same. Else we obtain an equivalent instance (G', k') such that $k' \leq k$ and the maximum degree of G' is bounded by $O(k \log^{3/2} k)$. Moreover it also returns a θ_c-hitting set, X, of G' of size at most $O(k \log^{3/2} k)$. Let d denote the treewidth of the graph after the removal of X, that is, $d := \mathbf{tw}(G \setminus X)$.

Now, we obtain our kernel in two phases: we first apply the protrusion rule selectively (Lemma 2) and get a polynomial kernel. Then, we apply the protrusion rule exhaustively on the obtained kernel to get a smaller kernel. To obtain the kernel we follow the following steps.

Applying the Protrusion Rule. By a result of Robertson et. al. [45] we know that any graph of treewidth greater than 20^{2c^5} contains a $c \times c$ grid, and hence θ_c, as a minor. Hence $d \leq 20^{2c^5}$. Now we apply Lemma 3 and get a $2(d+1)$-protrusion Y of G' of size at least $\frac{|V(G')|-|X|}{4|N(X)|+1}$. By Lemma 5, p-θ_c-FREE-DELETION has finite integer index. Let $\gamma : \mathbb{N} \to \mathbb{N}$ be the function defined in Lemma 2. Hence if $\frac{|V(G')|-|X|}{4|N(X)|+1} \geq \gamma(2d+1)$ then using Lemma 2 we replace the $2(d+1)$-protrusion Y of G' and obtain an instance G^* such that $|V(G^*)| < |V(G')|$, $k^* \leq k'$, and (G^*, k^*) is a YES-instance of p-θ_c-FREE-DELETION if and only if (G', k') is a YES-instance of p-θ_c-FREE-DELETION .

Before applying the Protrusion Rule again, if necessary, we bound the maximum degree of the graph by reapplying Lemma 11. This is done because the application of the protrusion rule could potentially increase the maximum degree of the graph. We alternately apply the protrusion rule and Lemma 11 in this fashion, until either Lemma 11 returns that G is a NO instance, or the protrusion rule ceases to apply.

Observe that this process will always terminate as the procedure that bounds the maximum degree never increases the number of vertices and the protrusion rule always reduces the number of vertices.

Let (G^*, k^*) be a reduced instance with hitting set X. In other words, there is no $(2d + 2)$-protrusion of size $\gamma(2d + 2)$ in $G^* \setminus X$, and the protrusion rule no longer applies. Now we show that the number of vertices and edges of this graph is bounded by $O(k^2 \log^3 k)$. We first bound the number of vertices. Since we cannot apply the Protrusion Rule, $\frac{|V(G^*)| - |X|}{4|N(X)| + 1} \leq \gamma(2d + 2)$. Since $k^* \leq k$ this implies that

$$
\begin{aligned}
|V(G^*)| &\leq \gamma(2d + 2)(4|N(X)| + 1) + |X| \\
&\leq \gamma(2d + 2)(4|X|\Delta(G^*) + 1) + |X| \\
&\leq \gamma(2d + 2)(O(k \log^{3/2} k) \times O(k \log^{3/2} k) + 1) + O(k \log^{3/2} k) \\
&\leq O(k^2 \log^3 k).
\end{aligned}
$$

To get the desired bound on the number of edges we first observe that since $\mathbf{tw}(G^* \setminus X) \leq 20^{2c^5} = d$, we have that the number of edges in $G^* \setminus X \leq d|V(G^*) \setminus X| = O(k^2 \log^3 k)$. Also the number of edges incident on the vertices in X is at most $|X| \cdot \Delta(G^*) \leq O(k^2(\log k)^3)$. This gives us a polynomial time algorithm that returns a kernel of size $O(k^2 \log^3 k)$.

Now we give a kernel of smaller size. To do so we apply combination of rules to bound the degree and the protrusion rule as before. The only difference is that we would like to replace any large $(2d + 2)$-protrusion in graph by a smaller one. We find a $2d + 2$-protrusion Y of size at least $\gamma(2d + 2)$ by guessing the boundary $\partial(Y)$ of size at most $2d + 2$. This could be performed in time $k^{O(d)}$. So let (G^*, k^*) be the reduced instance on which we can not apply the Protrusion Rule. Then we know that $\Delta(G^*) = O(k \log^{3/2} k)$. If G is a YES-instance then there exists a θ_c-hitting set X of size at most k such that $\mathbf{tw}(G \setminus X) \leq 20^{2c^5} = d$. Now applying the analysis above with this X yields that $|V(G^*)| = O(k^2 \log^{3/2} k)$ and $|E(G^*)| \leq O(k^2 \log^{3/2} k)$. Hence if the number of vertices or edges in the reduced instance G^*, to which we can not apply the Protrusion Rule, is more than $O(k^2 \log^{3/2} k)$ then we return that G is a NO-instance. This concludes the proof of the theorem. $\qquad\square$

Theorem 2 has following immediate corollary.

Corollary 4. p-Vertex Cover , p-Feedback Vertex Set *and* p-Diamond Hitting Set *have kernel of size* $O(k^2 \log^{3/2} k)$.

6 Conclusion

In this paper we gave the first kernelization algorithms for a subset of $\mathcal{F}$-Deletion problems and a generic approximation algorithm for the $\mathcal{F}$-Deletion problem when the set of excluded minors $\mathcal{F}$ contains at least one planar graph. Our approach generalizes and unifies known kernelization algorithms for p-Vertex Cover and p-Feedback Vertex Set. By the celebrated result of Robertson and Seymour, every $\mathcal{F}$-Deletion problem is FPT and our work naturally leads to the following question: does every $\mathcal{F}$-Deletion problem have a polynomial kernel? Can it be that for some finite sets of minor obstructions $\mathcal{F} = \{O_1, \ldots, O_p\}$ the answer to this question is NO? Even the case $\mathcal{F} = \{K_5, K_{3,3}\}$, vertex deletion to planar graphs, is an interesting challenge. Another interesting question is if our techniques can be extended to another important case when $\mathcal{F}$ contains a planar graph.

References

[1] F. N. Abu-Khzam, M. R. Fellows, M. A. Langston, and W. H. Suters, *Crown structures for vertex cover kernelization*, Theory Comput. Syst., 41 (2007), pp. 411–430.

[2] M. Akra and L. Bazzi, *On the solution of linear recurrence equations*, Computational Optimization and Applications, 10 (1998), pp. 195–210.

[3] N. ALON, G. GUTIN, E. J. KIM, S. SZEIDER, AND A. YEO, *Solving max-r-sat above a tight lower bound*, in Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), ACM-SIAM, 2010, pp. 511–517.

[4] S. ARNBORG, J. LAGERGREN, AND D. SEESE, *Easy problems for tree-decomposable graphs*, J. Algorithms, 12 (1991), pp. 308–340.

[5] V. BAFNA, P. BERMAN, AND T. FUJITO, *A 2-approximation algorithm for the undirected feedback vertex set problem*, SIAM J. Discr. Math., 12 (1999), pp. 289–297.

[6] R. BAR-YEHUDA AND S. EVEN, *A linear-time approximation algorithm for the weighted vertex cover problem*, J. Algorithms, 2 (1981), pp. 198–203.

[7] R. BAR-YEHUDA, D. GEIGER, J. NAOR, AND R. M. ROTH, *Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference*, SIAM J. Computing, 27 (1998), pp. 942–959.

[8] H. L. BODLAENDER, *A linear-time algorithm for finding tree-decompositions of small treewidth*, SIAM J. Comput., 25 (1996), pp. 1305–1317.

[9] H. L. BODLAENDER, *A cubic kernel for feedback vertex set*, in Proceedings of 24th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2007), vol. 4393 of Lecture Notes in Comput. Sci., Springer, 2007, pp. 320–331.

[10] ——, *Kernelization: New upper and lower bound techniques*, in Proceedings of the 4th Workshop on Parameterized and Exact Computation (IWPEC 2009), vol. 5917 of Lecture Notes in Computer Science, Springer, 2009, pp. 17–37.

[11] H. L. BODLAENDER, R. G. DOWNEY, M. R. FELLOWS, AND D. HERMELIN, *On problems without polynomial kernels*, J. Comput. Syst. Sci., 75 (2009), pp. 423–434.

[12] H. L. BODLAENDER, F. V. FOMIN, D. LOKSHTANOV, E. PENNINKX, S. SAURABH, AND D. M. THILIKOS, *(Meta) Kernelization*, in Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2009), IEEE, 2009, pp. 629–638.

[13] H. L. BODLAENDER, S. THOMASSÉ, AND A. YEO, *Kernel Bounds for Disjoint Cycles and Disjoint Paths*, in Proceedings of the 17th Annual European Symposium (ESA 2009), vol. 5757 of Lecture Notes in Comput. Sci., Springer, 2009, pp. 635–646.

[14] R. B. BORIE, G. R. PARKER, AND C. A. TOVEY, *Automatic Generation of Linear-Time Algorithms from Predicate Calculus Descriptions of Problems on Recursively Constructed Graph Families*, Algorithmica, 7 (1992), pp. 555–581.

[15] K. BURRAGE, V. ESTIVILL CASTRO, M. R. FELLOWS, M. A. LANGSTON, S. MAC, AND F. A. ROSAMOND, *The Undirected Feedback Vertex Set Problem Has a Poly(k) Kernel*, in Proceedings of 2nd International Workshop on Parameterized and Exact Computation (IWPEC 2006), vol. 4169 of Lecture Notes in Comput. Sci., Springer, 2006, pp. 192–202.

[16] J. CHEN, I. A. KANJ, AND W. JIA, *Vertex cover: further observations and further improvements*, J. Algorithms, 41 (2001), pp. 280–301.

[17] B. CHOR, M. R. FELLOWS, AND D. W. JUEDES, *Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps*, in Proceedings of the 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2004), vol. 3353 of LNCS, Springer, 2004, pp. 257–269.

[18] F. A. CHUDAK, M. X. GOEMANS, D. S. HOCHBAUM, AND D. P. WILLIAMSON, *A primal-dual interpretation of two 2-approximation algorithms for the feedback vertex set problem in undirected graphs*, Operations Research Letters, 22 (1998), pp. 111–118.

[19] B. N. CLARK, C. J. COLBOURN, AND D. S. JOHNSON, *Unit disk graphs*, Discrete Math., 86 (1990), pp. 165–177.

[20] B. COURCELLE, *The monadic second-order logic of graphs. i. recognizable sets of finite graphs*, Information and Computation, 85 (1990), pp. 12–75.

[21] B. COURCELLE, *The expression of graph properties and graph transformations in monadic second-order logic*, in Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations, G. Rozenberg, ed., World Scientific, 1997, ch. 5.

[22] B. COURCELLE AND M. MOSBAH, *Monadic second-order evaluations on tree-decomposable graphs*, Theor. Comp. Sci., 109 (1993), pp. 49–82.

[23] M. CYGAN, M. PILIPCZUK, M. PILIPCZUK, AND J. O. WOJTASZCZYK, *Improved fpt algorithm and quadratic kernel for pathwidth one vertex deletion*, in Proceedings of the 5th International Symposium on Parameterized and Exact Computation (IPEC 2010), Lecture Notes in Comput. Sci., Springer, 2010, p. to appear.

[24] F. K. H. A. DEHNE, M. R. FELLOWS, F. A. ROSAMOND, AND P. SHAW, *Greedy localization, iterative compression, and modeled crown reductions: New FPT techniques, an improved algorithm for set splitting, and a novel 2k kernelization for Vertex Cover*, in Proceedings of the First International Workshop on Parameterized and Exact Computation (IWPEC 2004), vol. 3162 of Lecture Notes in Comput. Sci., Springer, 2004, pp. 271–280.

[25] H. DELL AND D. VAN MELKEBEEK, *Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses*, in Proceedings of 42th ACM Symposium on Theory of Computing (STOC 2010), ACM, 2010, pp. 251–260.

[26] R. G. DOWNEY AND M. R. FELLOWS, *Parameterized Complexity*, Springer, 1998.

[27] P. ERDŐS AND L. PÓSA, *On independent circuits contained in a graph*, Canadian J. Math., 17 (1965), pp. 347–352.

[28] U. FEIGE, M. HAJIAGHAYI, AND J. R. LEE, *Improved approximation algorithms for minimum weight vertex separators*, SIAM J. Comput., 38 (2008), pp. 629–657.

[29] S. FIORINI, G. JORET, AND U. PIETROPAOLI, *Hitting diamonds and growing cacti*, in Proceedings of the 14th Conference on Integer Programming and Combinatorial Optimization (IPCO 2010), vol. 6080 of Lecture Notes in Comput. Sci., Springer, 2010, pp. 191–204.

[30] J. FLUM AND M. GROHE, *Parameterized Complexity Theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2006.

[31] F. V. FOMIN, D. LOKSHTANOV, S. SAURABH, AND D. M. THILIKOS, *Bidimensionality and kernels*, in Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), ACM-SIAM, 2010, pp. 503–510.

[32] L. FORTNOW AND R. SANTHANAM, *Infeasibility of instance compression and succinct PCPs for NP*, in Proceedings of the 40th ACM Symposium on Theory of Computing (STOC 2008), ACM, 2008, pp. 133–142.

[33] J. Guo and R. Niedermeier, *Invitation to data reduction and problem kernelization*, ACM SIGACT News, 38 (2007), pp. 31–45.

[34] D. S. Hochbaum and J. Naor, *Simple and fast algorithms for linear and integer programs with two variables per inequality*, SIAM J. Comput., 23 (1994), pp. 1179–1192.

[35] R. M. Karp, *Reducibility among combinatorial problems*, in Proc. Sympos complexity of computer computations, Plenum Press, New York, 1972, pp. 85–103.

[36] T. Kloks, *Treewidth – computations and approximations*, vol. 842 of Lecture Notes in Comput. Sci., Springer, 1994.

[37] P. G. Kolaitis and M. N. Thakur, *Approximation properties of NP minimization classes*, J. Comput. System Sci., 50 (1995), pp. 391–411.

[38] S. Kratsch, *Polynomial kernelizations for MIN F^+pi_1 and MAX NP*, in Proceedings of the 26th International Symposium on Theoretical Aspects of Computer Science (STACS 2009), vol. 3 of (LIPIcs), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2009, pp. 601–612.

[39] J. M. Lewis and M. Yannakakis, *The node-deletion problem for hereditary properties is NP-complete*, J. of Comp. System Sci., 20 (1980), pp. 219 – 230.

[40] C. Lund and M. Yannakakis, *The approximation of maximum subgraph problems*, in Proceedings of the 20th International Colloquium Automata, Languages and Programming (ICALP 1993), vol. 700 of Lecture Notes in Comput. Sci., Springer, 1993, pp. 40–51.

[41] G. L. Nemhauser and L. E. Trotter, Jr., *Properties of vertex packing and independence system polyhedra*, Math. Programming, 6 (1974), pp. 48–61.

[42] R. Niedermeier, *Invitation to fixed-parameter algorithms*, vol. 31 of Oxford Lecture Series in Mathematics and its Applications, Oxford University Press, Oxford, 2006.

[43] G. Philip, V. Raman, and S. Sikdar, *Solving dominating set in larger classes of graphs: FPT algorithms and polynomial kernels*, in Proceedings of the 17th Annual European Symposium on Algorithms (ESA 2009), vol. 5757 of Lecture Notes in Comput. Sci., Springer, 2009, pp. 694–705.

[44] N. Robertson and P. D. Seymour, *Graph minors. XIII. The disjoint paths problem*, J. Comb. Theory Ser. B, 63 (1995), pp. 65–110.

[45] N. Robertson, P. D. Seymour, and R. Thomas, *Quickly excluding a planar graph*, J. Comb. Theory Ser. B, 62 (1994), pp. 323–348.

[46] S. Thomassé, *A quadratic kernel for feedback vertex set*, ACM Transactions on Algorithms, 6 (2010).