Planar-$\mathcal{F}$ Deletion: Approximation, Kernelization and Optimal FPT Algorithms

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

Saket Saurabh[§]

Abstract

Let $\mathcal{F}$ be a finite set of graphs. In the $\mathcal{F}$-Deletion problem, 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. $\mathcal{F}$-Deletion is a generic problem and by selecting different sets of forbidden minors $\mathcal{F}$, one can obtain various fundamental problems such as Vertex Cover, Feedback Vertex Set or Treewidth η-Deletion.

In this paper we obtain a number of generic algorithmic results about $\mathcal{F}$-Deletion, when $\mathcal{F}$ contains at least one planar graph. The highlights of our work are

- A constant factor approximation algorithm for the optimization version of $\mathcal{F}$-Deletion;

- A linear time and single exponential parameterized algorithm, that is, an algorithm running in time $O(2^{O(k)}n)$, for the parameterized version of $\mathcal{F}$-Deletion where all graphs in $\mathcal{F}$ are connected;

- A polynomial kernel for parameterized $\mathcal{F}$-Deletion.

These algorithms unify, generalize, and improve a multitude of results in the literature. Our main results have several direct applications, but also the methods we develop on the way have applicability beyond the scope of this paper. Our results – constant factor approximation, polynomial kernelization and FPT algorithms – are stringed together by a common theme of polynomial time preprocessing.

---

[*]University of Bergen, Norway. `fomin@ii.uib.no`

[†]University of California, USA. `dlokshtanov@cs.ucsd.edu`

[‡]Institute of Mathematical Sciences, India. `neeldhara@imsc.res.in`

[§]Institute of Mathematical Sciences, India. `saket@imsc.res.in`

1 Introduction

Let $\mathscr{G}$ be the set of all finite undirected graphs and let $\mathscr{L}$ be the family of all finite subsets of $\mathscr{G}$. Thus every element $\mathcal{F} \in \mathscr{L}$ is a finite set of graphs and throughout the paper we assume that $\mathcal{F}$ is explicitly given. In this paper we study the following p-$\mathcal{F}$-DELETION problem.

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

The p-$\mathcal{F}$-DELETION problem defines a wide subclass of node (or vertex) removal problems studied from the 1970s. By the classical theorem of Lewis and Yannakakis [63], deciding if removing at most k vertices results with a subgraph with property π is NP-complete for every non-trivial property π. By a celebrated result of Robertson and Seymour, every p-$\mathcal{F}$-DELETION problem is non-uniformly fixed-parameter tractable (FPT). That is, for every k there is an algorithm solving the problem in time $O(f(k) \cdot n^3)$ [73]. The importance of the result comes from the fact that it simultaneously gives FPT algorithms for a variety of important problems such as VERTEX COVER, FEEDBACK VERTEX SET, VERTEX PLANARIZATION, etc. It is conceivable that meta theorems for vertex deletion problems might be formulated by addressing problems that are expressible in logics such as first order and monadic second order. However, since these capture problems that are known to be intractable, for example INDEPENDENT SET or DOMINATING SET, we do not expect to have a theorem that guarantees tractability for vertex deletion problems through this route. Therefore, the systematic study of the p-$\mathcal{F}$-DELETION problems is the more promising way forward to obtain meta-theorems for vertex removal problems on general undirected graphs.

In this paper we show that when $\mathcal{F} \in \mathscr{L}$ contains at least one planar graph, it is possible to obtain a number of generic results advancing known tractability borders of p-$\mathcal{F}$-DELETION. The case when $\mathcal{F}$ contains a planar graph, while being considerably more restricted than the general case, already encompasses a number of the well-studied instances of p-$\mathcal{F}$-DELETION. 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. Another fundamental problem, which is a special case of p-$\mathcal{F}$-DELETION, is TREEWIDTH η-DELETION or η-TRANSVERSAL which is to delete at most k vertices to obtain a graph of treewidth at most η. Since any graph of treewidth η excludes a $(\eta + 1) \times (\eta + 1)$ grid as a minor, we have that the set $\mathcal{F}$ of forbidden minors of treewidth η graphs contains a planar graph. TREEWIDTH η-DELETION plays important role in generic efficient polynomial time approximation schemes based on Bidimensionality Theory [48, 49]. Among other examples of p-$\mathcal{F}$-DELETION that can be found in the literature on approximation and parameterized algorithms, are the cases of $\mathcal{F}$ being $\{K_{2,3}, K_4\}$, $\{K_4\}$, $\{\theta_c\}$, and $\{K_3, T_2\}$, which correspond to removing vertices to obtain an outerplanar graph, a series-parallel graph, a diamond graph, and a graph of pathwidth one, respectively.

The main algorithmic contributions of our work is the following set of results for p-$\mathcal{F}$-DELETION for the case when $\mathcal{F}$ contains a planar graph:

- A constant factor approximation algorithm for an optimization version of p-$\mathcal{F}$-DELETION;
- A linear time and single exponential parameterized algorithm for p-$\mathcal{F}$-DELETION when all graphs in $\mathcal{F}$ are connected, that is, an algorithm running in time $O(2^{O(k)}n)$, where n is the input size;
- A polynomial kernel for p-$\mathcal{F}$-DELETION.

We use $\mathscr{F}$ to denote the subclass of $\mathscr{L}$ such that every $\mathcal{F} \in \mathscr{F}$ contains a planar graph.
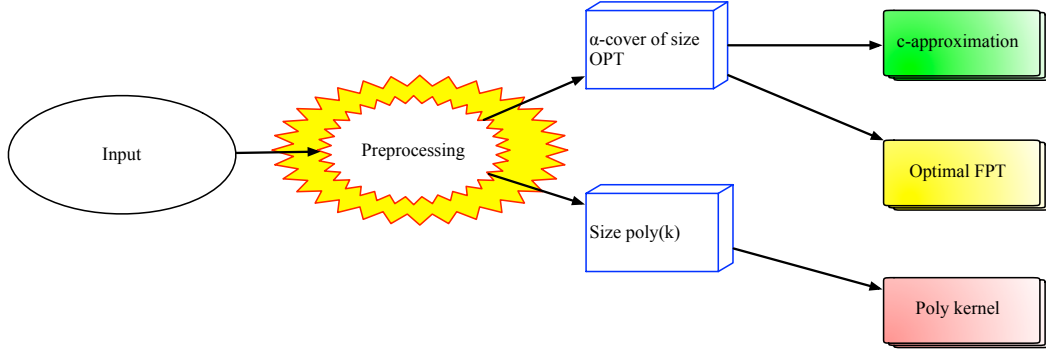
Figure 1: General view of our approach

Methodology. All our results – constant factor approximation, polynomial kernelization and FPT algorithms for p-$\mathcal{F}$-DELETION – have a common theme of polynomial time preprocessing. Preprocessing as a strategy for coping with hard problems is universally applied in practice and the notion of *kernelization* in parameterized complexity provides a mathematical framework for analyzing the quality of preprocessing strategies. In parameterized complexity each problem instance comes with a parameter k and a central notion in parameterized complexity is *fixed parameter tractability (FPT)*. This 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 parameterized problem is said to admit a *polynomial kernel* if there is a polynomial time algorithm (the degree of polynomial is independent of k), called a *kernelization* algorithm, that reduces the input instance down to an instance with size bounded by a polynomial $p(k)$ in k, while preserving the answer.

Thus the goal of kernelization is to apply reduction rules such that the size of the reduced instance can be upper bounded by a function of the parameter. However, if we want to use preprocessing for approximation or FPT algorithms, it is not necessary that the size of the reduced instance has to be upper bounded. What we need is a preprocessing procedure that allows us to navigate the solution search space efficiently. Our first contribution is a notion of preprocessing that is geared towards approximation and FPT algorithms. This notion relaxes the demands of kernelization and thus it is possible that a larger set of problems may admit this simplification procedure, when compared to kernelization. For approximation and FPT algorithms, we use the notion of α-*cover* as a measure of good preprocessing. For $0 < \alpha \leq 1$, we say that a vertex subset $S \subseteq V(G)$ is an α-*cover*, if the sum of vertex degrees $\sum_{v \in S} d(v)$ is at least $2\alpha|E(G)|$. For example, every vertex cover of a graph is also a 1-cover. The defining property of this preprocessing is that the equivalent simplified instance of the problem admits some optimal solution which is also an α-cover. If we succeed with this goal, then for an edge selected uniformly at random, with a constant probability at least one of its endpoints belong to some optimal solution. Using this as a basic step, we can construct approximation and FPT algorithms. But how to achieve this kind of preprocessing?

To achieve our goals we use the idea of graph replacement dating back to Fellows and Langston [42]. Precisely, what we use is the modern notion of "protrusion reduction" that has been recently employed in [16, 50] for obtaining meta-kernelization theorems for problems on sparse graphs like planar graphs, graphs of bounded genus [17], graphs excluding a fixed graph as a minor or induced subgraph [50, 47], or graphs excluding a fixed graph as a topological minor [62]. In this method, we find a large protrusion – a graph of small treewidth and small boundary – and then the preprocessing rule replaces this protrusion by a protrusion of constant size. One repeatedly applies this until no longer possible. Finally, by using combinatorial argu-

2

ments one upper bounds the size of the reduced induced (a graph without large protrusion). The FPT algorithms use the replacement technique developed in [17, 47], while for approximation algorithm we need another type of protrusion reduction. The reason why the normal protrusion replacement does not work for approximation algorithms is the same as why the NP-hardness reduction is not always an approximation preserving reduction. While the normal protrusion replacement works fine for preserving exact solutions, we needed a notion of protrusion reduction that also preserves approximate solutions. To this end, we develop a new notion of *lossless protrusion reduction*, and show that several problems do admit lossless protrusion reductions. We exemplify the usefulness of the new concept by obtaining constant factor approximation algorithms for $\mathcal{F}$-DELETION. These FPT and approximation algorithms are obtained by showing that solutions to the instances of the problem that do not contain protrusion form an α-cover for some fixed constant α.

Our final result is about kernelization for p-$\mathcal{F}$-DELETION. While protrusion replacements work well for constant factor approximation and optimal FPT algorithms, we do not know how to use this technique for kernelization algorithms for p-$\mathcal{F}$-DELETION. The technique was developed and used successfully for kernelization algorithms on sparse graphs [17, 50] but there are several limitations of this techniques which do not allow to use it on general graphs. Even for a sufficiently simple case of p-$\mathcal{F}$-DELETION, namely when $\mathcal{F}$ is a graph with two vertices and constant number of parallel edges, to apply protrusion replacements we have to do a lot of additional work to reduce large vertex degrees in a graph [47]. We do not know how to push these techniques for more complicated families families $\mathcal{F}$ and therefore, employ a different strategy. The new conceptual contribution here is the notion of a *near-protrusion*. Loosely speaking, a near-protrusion is a subgraph which can become a protrusion in the future, after removing some vertices of some optimal solution. The usefulness of near-protrusions is that they allow to find an irrelevant edge, i.e., an edge which removal does not change the problem. However, finding an irrelevant edge is highly non-trivial, and it requires the usage of well-quasi-ordering for graphs of bounded treewdith and bounded boundary as a subroutine.

As far as we are equipped with new tools and concepts: α-cover, lossless protrusion reduction and pseudo-protrusions, we are able to proceed with algorithms for p-$\mathcal{F}$-DELETION. These algorithms unify and generalize a multitude of results in the literature. In what follows we survey earlier results in each direction and discuss our results.

Approximation. In the optimization version of p-$\mathcal{F}$-DELETION, we want to compute the minimum set S, which removal leaves input graph G $\mathcal{F}$-minor-free. We denote this optimization problem by $\mathcal{F}$-DELETION. Characterising graph properties for which the corresponding vertex deletion problem can be approximated within a constant factor is a long standing open problem in approximation algorithms [77]. In spite of long history of research, we are still far from a complete understanding. Constant factor approximation algorithms for VERTEX COVER are known since 1970s [66, 6]. Lund and Yannakakis observed that the vertex deletion problem for any hereditary property with a finite number of minimal forbidden subgraphs can be approximated with a constant ratio [64]. They also conjectured that for every nontrivial, hereditary property with an infinite number of minimal forbidden subgraphs, the vertex deletion problem cannot be approximated with constant ratio. However, it appeared later that FEEDBACK VERTEX SET admits a constant factor approximation [7, 5] and thus the dividing line of approximability lies somewhere else. On a related matter, Yannakakis [76] showed that approximating the number of vertices to delete in order to obtain *connected* graph with some property π within factor $n^{1-\varepsilon}$ is NP-hard, see [76] for the definition of the property π. This result holds for very wide class of properties, in particular for properties being acyclic and outerplanar. There was no much progress on approximability/non-approximability of vertex deletion problems until recent work of Fiorini et al. [45] who gave a constant factor approximation algorithm for p-$\mathcal{F}$-DELETION for the case when $\mathcal{F}$ is a diamond graph, i.e., a graph with two vertices and three parallel edges.

Our first contribution is the theorem stating that every graph property π expressible by a finite set of forbidden minors containing at least one planar graph, the vertex deletion problem for property π admits a constant factor approximation algorithm. In other words, we prove the following theorem

Theorem 1. *For every set $\mathcal{F} \in \mathscr{F}$, $\mathcal{F}$-DELETION admits a randomized constant ratio approximation algorithm.*

Let us remark that for all known constant factor approximation algorithms of vertex deletion to a hereditary property π, property π is either characterized by an finite number of minimal forbidden subgraphs or by finite number of forbidden minors, one of which is planar. Theorem 1 together with the result of Lund and Yannakakis, not only encompass all known vertex deletion problems with constant factor approximation ratio but significantly extends known tractability borders for such types of problems.

Kernelization. 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, 16, 50, 60]. The recent development of a framework for ruling out polynomial kernels under certain complexity-theoretic assumptions [15, 35, 51] has added a new dimension to the field and strengthened its connections to classical complexity. For overviews of kernelization we refer to surveys [13, 53] and to the corresponding chapters in books on Parameterized Complexity [46, 67].

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. [39], that a parameterized problem is fixed parameter tractable, or belongs to the class FPT, if and only if it has a (not necessarily polynomial) kernel. Kernelization enables us to classify problems within the class FPT further, based on the sizes of the problem kernels. One of the fundamental challenges in the area is the possibility of characterising general classes of parameterized problems possessing kernels of polynomial sizes.

Polynomial kernels for several special cases of p-$\mathcal{F}$-DELETION were studied in the literature. Different kernelization techniques were invented for VERTEX COVER, eventually resulting in a $2k$-sized vertex kernel [1, 26, 34, 55]. For the kernelization of FEEDBACK VERTEX SET, there has been a sequence of dramatic improvements starting from an $O(k^{11})$ vertex kernel by Buragge et al. [22], improved to $O(k^3)$ by Bodlaender [12], and then finally to $O(k^2)$ by Thomassé [75]. A polynomial kernel for p-$\mathcal{F}$-DELETION for class $\mathcal{F}$ consisting of a graph with two vertices and several parallel edges is given in [47]. Philip et al. [68] and Cygan et al. [31] obtained polynomial kernels for PATHWIDTH 1-DELETION. Our next theorem generalizes all these kernelization results.

Theorem 2. *For every set $\mathcal{F} \in \mathscr{F}$, p-$\mathcal{F}$-DELETION admits a polynomial kernel.*

In fact, we prove more general result—the kernelization algorithm of Theorem 3 always outputs a minor of the input graph. This has interesting combinatorial consequences. By Robertson and Seymour theory every non-trivial minor-closed class of graphs can be characterized by a finite set of forbidden minors or *obstructions*. While Graph Minors Theory insures that many interesting graph properties have finite obstructions sets, these seem to be disappointingly huge in many cases. There are a number of results that bound the size of the obstructions for specific minor closed families of graphs. Fellows and Langston [43, 44] suggested a systematic method of computing the obstructions sets for many natural properties, see also the recent work of Adler et al. [2]. Bodendiek and Wagner gave bounds on sizes of obstructions of genus at most k [9], later improved by Djidjev and Reif [38]. Gupta and Impagliazzo studied bounds on the size of

4

a planar intertwine of two given planar graphs [54]. Lagergren [61] showed that the number of edges in every obstruction to a graph of treewidth k is at most double exponential in $O(k^5)$. Dvořák et al. [40] provide similar bound on obstructions to graphs of tree-depth at most k. Dinneen and Xiong have shown that the number of vertices in connected obstruction for graphs with vertex cover at most k is at most $2k+1$ [37]. Obstructions for graphs with feedback vertex set of size at most k is discussed in the work of Dinneen et al. [36].

For a finite set of graphs $\mathcal{F}$, let $\mathcal{G}_{\mathcal{F},k}$ be a class of graphs such that for every $G \in \mathcal{G}_{\mathcal{F},k}$ there is a subset of vertices S of size at most k such that $G \setminus S$ has no minor from $\mathcal{F}$. As a corollary of kernelization algorithm, we obtain the following combinatorial result.

Theorem 3. *For every set $\mathcal{F} \in \mathscr{F}$, every minimal obstruction for $\mathcal{G}_{\mathcal{F},k}$ is of size polynomial in k.*

Fast FPT Algorithms. The study of parameterized problems proceeds in several steps. The first step is to establish if the problem on hands is fixed parameter tractable or not. If the problem is in FPT, then the next steps are to identify if the problem admits a polynomial kernel and to find the fastest possible FPT algorithm solving the problem. The running time of every FPT algorithm is $O(f(k)n^c)$, that is, the product of a super-polynomial function $f(k)$ depending only on the parameter k and polynomial n^c, where n is the input size and c is some constant. Both steps, minimizing super-polynomial function $f(k)$ and minimizing the exponent c of the polynomial part, are important parts in the design and analysis of parameterized algorithms.

The p-$\mathcal{F}$-DELETION problem was introduced by Fellows and Langston [41], who gave a non-constructive algorithm running in time $O(f(k) \cdot n^2)$ for some function $f(k)$ [41, Theorem 6]. This result was improved by Bodlaender [10] to $O(f(k) \cdot n)$, for $f(k) = 2^{2^{O(k \log k)}}$. There is a substantial amount of work on improving the exponential function $f(k)$ for special cases of p-$\mathcal{F}$-DELETION. For the VERTEX COVER problem the existence of single-exponential algorithms is well-known since almost the beginnings of the field of Parameterized Complexity, the current best algorithm being by Chen et al. [27]. Randomized parameterized single exponential algorithm for FEEDBACK VERTEX SET was given by Becker et al. [8] but existence of deterministic single-exponential algorithms for FEEDBACK VERTEX SET was open for a while and it took some time and discovery of iterative compression [71] to reduce the running time to $2^{O(k)}n^{O(1)}$ [23, 25, 30, 33, 52, 70]. The current champion for FEEDBACK VERTEX SET are the deterministic algorithm of Cao et al. [23] with running time $O(3.83^k k n^2)$ and the randomized of Cygan et al. with running time time $3^k n^{O(1)}$ [30]. Recently, Joret et al. [57] showed that p-$\mathcal{F}$-DELETION for $\mathcal{F} = \{\theta_c\}$, where θ_c is the graph with two vertices and c parallel edges, can be solved in time $2^{O(k)}n^{O(1)}$ for every fixed c. Philip et al. [69] studied PATHWIDTH 1-DELETION and obtained an algorithm with running time $O(7^k n^2)$ that was later improved to $O(4.65^k n^{O(1)})$ in [31]. Kim et al. [58] gave a single exponential algorithm for $\mathcal{F} = \{K_4\}$. Unless Exponential Time Hypothesis (ETH) fails [24, 56], single exponential dependence on the parameter k is asymptotically the best bound one can obtain for p-$\mathcal{F}$-DELETION, and thus our next theorem provides asymptotically optimal bounds on the exponential function of the parameter and polynomial contribution of the input.

We call a family $\mathcal{F} \in \mathscr{F}$ connected if every graph in $\mathcal{F}$ is connected.

Theorem 4. *For every connected set $\mathcal{F} \in \mathscr{F}$ containing a planar graph, there is a randomized algorithm solving p-$\mathcal{F}$-DELETION in time $O(c^k n)$ for some constant $c > 1$.*

We finally give a deterministic algorithm for p-$\mathcal{F}$-DELETION. Surprisingly, our algorithm does not use iterative compression but is based on branching on degree sequences.

Theorem 5. *For every connected set $\mathcal{F} \in \mathscr{F}$ containing a planar graph, p-$\mathcal{F}$-DELETION is solvable in time $O(c^k n \log^2 n)$ for some constant $c > 1$.*

2 Preliminaries

In this section we give various definitions which we use in the paper. 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 $d(v)$. 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 $N_G(S)$ the open neighborhood of S, i.e. the set of vertices in $V(G) \setminus S$ adjacent to S. Whenever the graph G is clear from the context, we omit the subscript in $N_G(S)$ and denote it only by $N(S)$. By $N[S]$ we denote $N(S) \cup S$. 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 $\mathcal{F}$-*deletion set* if $G \setminus S$ does not contain any graphs in the family $\mathcal{F}$ as a minor.

2.1 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. We assume that k is *given in unary* and hence $k \leq |x|$. 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**] *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 K, called* kernelization algorithm, *or, in short, a* kernelization, *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) $\max\{|x'|, k'\} \leq g(k)$.

When $g(k) = k^{O(1)}$ or $g(k) = O(k)$ then we say that Π admits a polynomial or linear kernel *respectively. If additionally $k' \leq k$ we say that the kernel is* strict.

2.2 Treewidth.

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:

- $\forall e = uv \in E(G), \exists t \in V(T) : \{u, v\} \subseteq X_t$ and

- $\forall v \in V(G), T[\{t \mid v \in X_t\}]$ 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 equal to $\max\{|X_t| - 1 \mid t \in V(T)\}$ and the *treewidth* of G is the minimum width over all tree decompositions of G.

A *nice tree decomposition* is a pair $(T, \mathcal{X})$ where $(T, \mathcal{X})$ is a tree decomposition such that T is a rooted tree and the following conditions are satisfied:

- Every node of the tree T has at most two children;

- if a node t has two children t_1 and t_2, then $X_t = X_{t_1} = X_{t_2}$; and

- if a node t has one child t_1, then either $|X_t| = |X_{t_1}| + 1$ and $X_{t_1} \subset X_t$ (in this case we call t_1 *insert node*) or $|X_t| = |X_{t_1}| - 1$ and $X_t \subset X_{t_1}$ (in this case we call t_1 *insert node*).

It is possible to transform a given tree decomposition $(T, \mathcal{X})$ into a nice tree decomposition $(T', \mathcal{X}')$ in time $O(|V| + |E|)$ [11].

2.3 Minors

Given an edge $e = xy$ of a graph G, the graph G/e is obtained from G by contracting the edge e, that is, the endpoints x and y are replaced by a new vertex v_{xy} which is adjacent to the old neighbors of x and y (except from x and y). A graph H obtained by a sequence of edge-contractions is said to be a *contraction* of G. We denote it by $H \leq_c G$. A graph H is a *minor* of a graph G if H is the contraction of some subgraph of G and we denote it by $H \leq_m G$. We say that a graph G is H-*minor-free* when it does not contain H as a minor. We also say that a graph class $\mathcal{G}$ is H-*minor-free* (or, excludes H as a minor) when all its members are H-minor-free. It is well-known [72] that if $H \leq_m G$ then $tw(H) \leq tw(G)$. We will also use the following fact about excluding planar graphs as minors.

Proposition 1. *There is a constant c such that for every planar H and graph G with $tw(G) \geq 2^{c|V(H)|^3}$, H is a minor of G.*

2.4 t-Boundaried graphs and Gluing.

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 $\ell_G(v) \in \{1, \ldots, t\}$. Each vertex in B has a unique label. We refer to B as the *boundary* of G. For a t-boundaried G the function $\delta(G)$ returns the boundary of G. Two t-boundaried graphs G and H are isomprphic if there is a bijection f from $V(G)$ to $V(H)$ such that $uv \in E(G) \iff f(u)f(v) \in E(H)$, for every $v \in \delta(G)$ we have $f(v) \in \delta(H)$ and $\ell_G(v) = \ell_H(f(v))$. Specifically f is an isomorphism between G and H in the normal graph sense, but additionally f respects the labels of the border vertices. Observe that a t-boundaried graph may have no boundary at all. A graph G is isomorphic to a t-boundaried graph H of there is an isomorphism between G and H.

Two t-boundaried graphs G_1 and G_2 can be *glued* together to form a graph $G = G_1 \oplus G_2$. The gluing operation takes the disjoint union of G_1 and G_2 and identifies the vertices of $\delta(G_1)$ and $\delta(G_2)$ with the same label. If there are vertices $u_1, v_1 \in \delta(G_1)$ and $u_2, v_2 \in \delta(G_2)$ such that $\ell_{G_1}(u_1) = \ell_{G_2}(u_2)$ and $\ell_{G_1}(v_1) = \ell_{G_2}(v_2)$ then G has vertices u formed by unifying u_1 and u_2 and v formed by unifying v_1 and v_2. The new vertices u and v are adjacent if $u_1 v_1 \in E(G_1)$ or $u_2 v_2 \in E(G_2)$.

The *boundaried gluing operation* $\oplus_\delta$ is similar to the normal gluing operation, but results in a t-boundaried graph rather than a graph. Specifically $G_1 \oplus_\delta G_2$ results in a t-boundaried graph where the graph is $G = G_1 \oplus G_2$ and a vertex is in the boundary of G if it was in the boundary of G_1 or G_2. Vertices in the boundary of G keep their label from G_1 or G_2. Both for gluing and boundaried gluing we will refer to $G_1 \oplus G_2$ or $G_1 \oplus_\delta G_2$ as the *sum* of G_1 and G_2, and G_1 and G_2 are the *terms* of the sum.

For a t-boundaried graph G and boundary vertex $v \in \delta(G)$, *forgetting* v results in a t-boundaried graph identical to G, except that v is no longer a boundary vertex. All other boundary vertices keep their labels. Forgetting a non-boundary vertex leaves the graph unchanged, as does forgetting a vertex that is not in the vertex set of G. Forgetting a set $S \subseteq \delta(G)$ of vertices means forgetting all vertices in the set. The function **forget**(G, S) returns the t-boundaried graph resulting from forgetting S in G.

We will frequently need to construct t-boundaried graphs from subgraphs of a graph G. For a graph G and two disjoint vertex sets P and B we define G_P^B to be the t-boundaried graph

$G[P \cup B]$ with boundary B. The labelling of the border B is chosen in a manner independent of P - such that if P_1, P_2 and B are disjoint then $G^B_{P_1} \oplus_\delta G^B_{P_2} = G^B_{P_1 \cup P_2}$.

2.5 Monadic Second Order Logic (MSO)

The syntax of MSO on graphs includes the logical connectives $\vee$, $\wedge$, $\neg$, $\Leftrightarrow$, $\Rightarrow$, 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 [20, 28].

H minor-models. Recall that a t-boundaried graph H is a minor of a t-boundaried graph G if (a t-boundaried graph isomorphic to) H can be obtained from G by deleting vertices or edges or contracting edges, but never contracting edges with both endpoints being boundary vertices. Let $V(H) = \{h_1, \ldots, h_c\}$, and let $B_G := \{b^G_1, \ldots b^G_t\}$ and $B_H := \{b^H_1, \ldots b^H_t\}$ denote $\delta(G)$ and $\delta(H)$ respectively. Then, the formulation that $H \leq_m G$ is given by $\phi(G, H, B_G, B_G)$:

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

2.6 Finite Integer Index and Protrusions

For a parameterized problem Π and two t-boundaried graphs $G_1, G_2 \in \mathcal{G}$, we say that $G_1 \equiv_\Pi G_2$ if there exists a constant c such that for every t-boundaried graph G and for every integer k, $(G_1 \oplus G, k) \in \Pi$ if and only if $(G_2 \oplus G, k + c) \in \Pi$. For every t, the relation $\equiv_\Pi$ on t-boundaried graphs is an equivalence relation, and we call $\equiv_\Pi$ the *canonical equivalence relation* of Π. We say that a problem Π has *Finite Integer Index* if for every t, $\equiv_\Pi$ has finite index on t-boundaried graphs. Thus, if Π has finite integer index then for every t there is a finite set $\mathcal{S}$ of t-boundaried graphs for every 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). We will repeatedly make use of the following proposition.

Proposition 2 ([17]). *For every connected $\mathcal{F} \in \mathscr{F}$, $\mathcal{F}$-DELETION has finite integer index.*

Protrusions and Protrusion Replacement For 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 *neighbourhood* of S is $N_G(S) = \partial_G(V(G) \setminus S)$. When it is clear from the context, we omit the subscripts. A *r-protrusion* in a graph G is a set $X \subseteq V$ such that $|\partial(X)| \leq r$ and $\mathbf{tw}(G[X]) \leq r$. If G is a graph containing a r-protrusion X and X' is a r-boundaried graph, the act of *replacing X by X'* means replacing G by $G^{\partial(X)}_{V(G) \setminus X} \oplus X'$.

A *protrusion replacer* for a parameterized graph problem Π is a family of algorithms, with one algorithm for every constant r. The r'th algorithm has the following specifications. There exists a constant r' (which depends on r) such that given an instance (G, k) and an r-protrusion X in G of size at least r', the algorithm runs in time $O(|X|)$ and outputs an instance (G', k') such that $(G', k') \in \Pi$ if and only if $(G, k) \in \Pi$, $k' \leq k$ and G' is obtained from G by replacing X by a r-boundaried graph X' with less than r' vertices. Observe that since X has at least r' vertices and X' has less than r' vertices this implies that $|V(G')| < |V(G)|$. The following proposition is the driving force of [17] and the starting point for our algorithms.

Proposition 3 ([32, 17]). *Every parameterized problem with finite integer index has a protrusion replacer.*

Together, Propositions 2 and 3 imply that for every connected $\mathcal{F} \in \mathscr{F}$, $\mathcal{F}$-DELETION has a protrusion replacer.

Bodlaender et al. [16] were first to use protrusion techniques (or rather graph reduction techniques) to obtain kernels, but the idea of graph replacement for algorithms dates back to Fellows and Langston [42]. Arnborg et al. [4] essentially showed that protrusion replacement is possible and useful for many problems on graphs of bounded treewidth, and gave safe ways of reducing graphs. Using this, Arnborg et al. [4] obtained a linear time algorithm for MSO expressible problems on graphs of bounded treewidth. Bodlaender and Fluiter [14, 19, 32] generalized these ideas in several ways — in particular, they lifted graph reduction techniques to optimization problems and proved Proposition 3. It is also important to mention the work of Bodlaender and Hagerup [18], who used the concept of graph reduction to obtain parallel algorithms for MSO expressible problems on bounded treewidth graphs.

2.6.1 Least Common Ancestor-Closure of Sets in Trees.

For a rooted tree T and vertex set M in $V(T)$ the least common ancestor-closure (*LCA-closure*) **LCA-closure**(M) is obtained by the following process. Initially, set $M' = M$. Then, as long as there are vertices x and y in M' whose least common ancestor w is not in M', add w to M'. When the process terminates, output M' as the LCA-closure of M. The following folklore lemma summarizes two basic properties of LCA closures.

Lemma 1. *Let T be a tree, $M \subseteq V(T)$ and $M' = $ **LCA-closure**(M). Then $|M'| \leq 2|M|$ and for every connected component C of $T \setminus M'$, $|N(C)| \leq 2$.*

Proof. To prove that $|M'| \leq 2|M|$ make a tree T' with vertex set M', and for every vertex $v \in M'$ adding an edge to the lowermost ancestor of v in M' in the tree T. Observe that in T' all leaves are from M, since every vertex in $M' \setminus M$ is the least common ancestor of two vertices below it in T. Furthermore, for the same reason every vertex in $M' \setminus M$ has at least two decendants in T'. A standard counting argument for trees shows that the number of vertices with at least two decendants is at most the number of leaves. Hence $|M' \setminus M| \leq |M|$ and so $|M'| \leq 2|M|$.

We now prove that $|N(C)| \leq 2$. Suppose not, and let r be the root of P. At most one of C's neighbours is the parent of r and hence at least two of C's neighbours, say u and v are children of vertices in C. The vertices u and v are both in M', and they are both descendents of r. But

then the least common ancestor of u and v must lie in C and hence is not in M', contradicting the construction of M'. So we conclude that $|N(P)| \leq 2$. $\qquad \square$

3   A Randomized Algorithm for "connected" p-$\mathcal{F}$-Deletion

In this section we give a randomized algorithm for p-$\mathcal{F}$-Deletion when every graph in $\mathcal{F} \in \mathscr{F}$ is connected. Recall that we call a family $\mathcal{F}$ connected if all the graphs in $\mathcal{F}$ is connected. We will show that for every connected $\mathcal{F}$ the algorithm runs in polynomial time, with the exponent of the polynomial depending on the family $\mathcal{F}$. If the input graph has a $\mathcal{F}$-deletion set of size at most k, the algorithm will detect a $\mathcal{F}$-deletion set of size at most k with probability at least $\frac{1}{c^k}$. Here the constant c depends on $\mathcal{F}$. The algorithm has no false positives - we show that if it reports that a $\mathcal{F}$-deletion set of size at most k exists then G indeed has such a set.

In the following sections we will progressively improve the algorithm; first we give an implementation of the algorithm with expected running time $O(n \cdot OPT)$. Then we show how to modify the (sped up) algorithm so that it not only decides whether G has a $\mathcal{F}$-deletion set of size at most k, but also outputs a solution. We show that if G has a $\mathcal{F}$-deletion set of size at most k, the algorithm will output a solution of size k with probability at least $\frac{1}{c^k}$. We then proceed to show that this algorithm in fact outputs constant factor approximate solutions with constant probability, yielding a constant factor approximation for p-$\mathcal{F}$-Deletion for connected $\mathcal{F}$ in expected $O(n \cdot OPT)$ time. The main structure of the improved algorithm remains the same as the one described here.

The first building block of our algorithm is a simple algorithm to reduce the input instance to an equivalent instance that does not contain any large protrusions with small border.

Lemma 2. *For every $\mathcal{F} \in \mathscr{F}$ and constants r and r' such that p-$\mathcal{F}$-Deletion has a protrusion replacer that reduces r-protrusions of size r', there is an algorithm that takes as input an instance (G, k) of p-$\mathcal{F}$-Deletion, runs in $n^{O(r')}$ time and outputs an equivalent instance (G', k') such that $|V(G')| \leq V(G)$, $k' \leq k$ and G' has no r-protrusion of size at least r'.*

Proof. It is sufficient to give a $n^{O(r')}$ time algorithm to find a r-protrusion X in G of size at least r', if such a protrusion exists. If we had such an algorithm to find a protrusion we could keep looking for r-protrusions X in G of size at least r', and if one is found replacing them using the protrusion replacer. Since each replacement decreases the number of vertices by one we converge to an instance (G', k') with the desired properties after at most n iterations.

To find an r-protrusion of size at least r' observe that if such a protrusion exists, then there must be at least one such protrusion X such that $G[X \setminus \partial(X)]$ has at most r' connected components. Indeed, if $G[X \setminus \partial(X)]$ has more than r' connected components then let X' be $\partial(X)$ plus the union of any r' components of $G[X \setminus \partial(X)]$. Now X' is an r-protrusion of size at least r' and $G[X' \setminus \partial(X')]$ has at most r' components. To find a r-protrusion X of size at least r' on at most r' components, guess $\partial(X)$ and then guess which components of $G \setminus \partial(X)$ are in X. The size of the search space is bounded by $n^r \cdot n^{r'}$ and for each candidate X we can test whether it is a protrusion in linear time using Bodlaender's linear time treewidth algorithm [11]. $\qquad \square$

The second building block of our algorithm is a lemma whose proof we postpone until the end of this section. The lemma states that for any $\mathcal{F} \in \mathscr{F}$, if G contains no large protrusions with small border then any feasible solution to p-$\mathcal{F}$-Deletion is incident to a linear fraction of the edges of G. Recall that an α-cover in G is a set S such that $\sum_{v \in S} d(v) \geq \alpha \cdot \sum_{v \in V(G)} d(v) = 2\alpha \cdot m$.

Lemma 3. *For every $\mathcal{F} \in \mathscr{F}$ there exist constants r and α such that if a graph G has no r-protrusion of size at least r', then every $\mathcal{F}$-deletion set S of G is a $\frac{\alpha}{r'}$-cover of G.*

We now combine Lemmata 2 and 3 to give a randomized algorithm for p-$\mathcal{F}$-DELETION for all $\mathcal{F} \in \mathscr{F}$ such that each graph in $\mathcal{F}$ is connected.

Randomized-FPT-beta((G,k))

Set $G_{current} := G$ and $k_{current} := k$.

While ($G_{current}$ is not $\mathcal{F}$-free) do as follows:

1. If $k_{current} \leq 0$ return that G does not have a k-sized $\mathcal{F}$-deletion set .

2. Apply Lemma 2 on $(G_{current}, k_{current})$ and obtain an equivalent instance (G', k').

3. Pick a vertex $u \in V(G)$ at random with probability $\frac{d(u)}{2m}$. Set $G_{current} := G' \setminus \{u\}$ and $k_{current} := k' - 1$

Return that G has a k-sized $\mathcal{F}$-deletion set .

Figure 2: In Algorithm **Randomized-FPT-beta**, let r be the constant as guaranteed by Lemma 3 and let r' be the smallest integer such that the protusion replacer for $\mathcal{F}$-DELETION reduces r-protrusions of size r'.

Lemma 4. *Algorithm 2 runs in polynomial time, if (G, k) is a "no" instance it outputs "no" and if (G, k) is a "yes" instance it outputs "yes" with probability at least $\frac{1}{c^k}$ where c is a constant depending only on $\mathcal{F}$.*

Proof. Since each iteration runs in polynomial time and reduces the number of vertices in $G_{current}$ by at least one, Algorithm 2 runs in polynomial time. Furthermore, Step 2 reduces the instance to an equivalent instance with $k' \leq k_{current}$ and Step 3 only decreases $k_{current}$ when it puts a vertex into the solution. Hence when the algorithm outputs "yes" then a k-sized $\mathcal{F}$-deletion set exists. It remains to show the last part of the statement.

We say that an iteration of Step 3 is *successful* if there exists a $\mathcal{F}$-deletion set S of G' with $|S| \leq k'$ such that the vertex u selected in this step is in S. If the step is successfull then $S \setminus \{u\}$ is a $\mathcal{F}$-deletion set of G' of size at most $k' - 1$. Thus, if the input graph G has a k-sized $\mathcal{F}$-deletion set and all the iterations of Step 3 are successful then the algorithm maintains the invariant that $G_{current}$ has a $\mathcal{F}$-deletion set of size at most $k_{current}$, and thus after at most k iterations it terminates and outputs that (G, k) is a "yes" instance. When Step 3 is executed the graph G' has no r-protrusions of size at least r'. Thus by Lemma 3 every $\mathcal{F}$-deletion set set of G' is an $\frac{\alpha}{r'}$-cover for a constant α depending only on $\mathcal{F}$. Hence the probability that u is in a minimum size $\mathcal{F}$-deletion set of G' is at least $\frac{\alpha}{r'}$. We conclude that the probability that the first k executions of Step 3 are successful is at least $(\frac{\alpha}{r'})^k$ concluding the proof. □

Repeating the algorithm presented in Figure 2 $O(c^k)$ times yields a $O(2^{O(k)} n^{O(1)})$ time algorithm for p-$\mathcal{F}$-DELETION for all connected $\mathcal{F} \in \mathscr{F}$. However we are not entirely done with the proof of Lemma 4, as it remains to prove Lemma 3. In order to complete the proof we need to define protrusion decompositions.

3.1 Protrusion Decompositions and Proof of Lemma 3

We recall the notion of a protrusion decomposition defined in [17] and show that if a graph G has a set X such that $\mathbf{tw}(G \setminus X) \leq d$, then it admits a protrusion decomposition for an appropriate value of the parameters. We then use this result to prove Lemma 3.

Definition 1. [**Protrusion Decomposition**][[**17**]] *A graph G has an (α, β)-protrusion decomposition if $V(G)$ has a partition $\mathcal{P} = \{R_0, R_1, \ldots, R_t\}$ where*

- $\max\{t, |R_0|\} \leq \alpha$,

- each $N_G[R_i]$, $i \in \{1, \ldots, t\}$ is a β-protrusion of G, and

- for all $i > 1$, $N[R_i] \subseteq R_0$.

We call the sets $R_i^+ = N_G[R_i]$, $i \in \{1, \ldots, t\}$ protrusions of $\mathcal{P}$.

We now show that for every $\mathcal{F} \in \mathscr{F}$ every graph with an $\mathcal{F}$-deletion set X has an (α, β)-protrusion decomposition where β is constant and $\alpha = O(|N[X]|)$.

Lemma 5 (Protrusion Decomposition Lemma). *If a n-vertex graph G has a vertex subset X such that $\mathbf{tw}(G \setminus X) \leq b$, then G admits a $((4|N[X]|)(b+1), 2(b+1))$-protrusion decomposition. Furthermore, if we are given the set X then this protrusion decomposition can be computed in linear time. Here b is a constant.*

Proof. We give a proof for the case when X is explicitly given to us. The proof will automatically imply the existence of a $(4(b+1)|N(X)|, 2b+2)$-protrusion decomposition of G for the case when we are just guaranteed the existence of X. The algorithm starts by computing a nice tree decomposition $(T, \mathcal{B})$ of $G \setminus X$ with width at most b. Notice that since b is a constant this can be done in linear time [11].

For every $v \in N(X)$ add a node u in T such that $v \in B_u$ to a set M'. We have that $M' \leq |N(X)|$. Let M' be the set of marked nodes and set $M = \mathbf{LCA\text{-}closure}(M')$. By Lemma 1, $M \leq 2|M'| \leq 2|N(X)|$. Let $Q_1, Q_2 \ldots Q_t$ be the connected components of $T \setminus Q$. Since T is a binary tree $T \setminus M$ has at most $2|M| + 1$ connected components, so $t \leq 4|N(X)| + 1$. By Lemma 1 we have that for every $i \leq t$, $|N_T(Q_i)| \leq 2$.

Define $R_0 = X \cup \bigcup_{u \in M} B_u$ and for each $1 \leq i \leq t$ set $R_i = \bigcup_{u \in Q_i} B_u \setminus R_0$. Since every vertex of $G \setminus X$ appears in a bag of the tree-decomposition, $R_0, \ldots R_t$ forms a partition of $V(G)$. By construction we have that for every $i \geq 1$, $N(R_i) \subseteq R_0$ and $\mathbf{tw}(G[N[R_i]]) \leq b$. Furthermore, since $|N_T(Q_i)| \leq 2$ we have $|N(R_i)| \leq 2(b+1)$. Thus $R_0 \ldots R_t$ form a (α, β)-protrusion decomposition of G where $\beta \leq 2(b+1)$ and $\alpha \leq \max(|R_0|, t) \leq (4|N[X]|)(b+1)$. It is easy to implement a procedure that computes $R_0 \ldots R_t$ in this way in linear time. $\square$

We are now in a position to prove Lemma 3

Proof of Lemma 3. We need to prove that for every $\mathcal{F} \in \mathscr{F}$ there exist constants r and α such that if a graph G has no r-protrusion of size at least r', then every minimal $\mathcal{F}$-deletion set S of G is a $\frac{\alpha}{r'}$-cover of G. By Proposition 1 there exists a constant η depending only on $\mathcal{F}$ such that $\mathbf{tw}(G \setminus S) \leq \eta$. By Lemma 5, G has a $((4|N[S]|)(\eta + 1), 2(\eta + 1))$-protrusion decomposition $R_0 \ldots R_t$. Set $r = 2(\eta + 1)$ and suppose G has no r-protrusions of size at least r'. Then $t \leq (4|N[S]|)(\eta + 1)$, $|R_0| \leq (4|N[S]|)(\eta + 1)$ and so $|V(G)| = |R_0| + \sum_i |R_i| \leq (4|N[S]|)(\eta + 1)(r' + 1) \leq (8|N[S]|)(\eta + 1)r'$. Since $\mathbf{tw}(G \setminus S) \leq \eta + 1$ it follows that $G \setminus S$ is $(\eta + 1)$-degenerate and so $\sum_{v \in V(G) \setminus S} d(v) \leq (8|N[S]|)(\eta + 1)^2 r'$. Set $\alpha = \frac{1}{18(\eta+1)^2}$ and observe that

$$\sum_{v \in V(G)} d(v) \leq \sum_{v \in S} d(v) + \sum_{v \in V(G) \setminus S} d(v) \leq \sum_{v \in S} d(v) + (8|N[S]|)(\eta + 1)^2 r' \leq \frac{r}{\alpha} \cdot \sum_{v \in S} d(v).$$

The last inequality follows from the fact that there are no isolated vertex in S. $\square$

4 Fast Protrusion Replacement

What makes the polynomial factor of Algorithm 2 large is the algorithm of Lemma 2 to remove all large enough protrusions with small border size. In this section we give much faster algorithms that reduce "almost all" large protrusions with small border. We then show that reducing almost all protrusions instead of all protrusions is sufficient to obtain the conclusion of Lemma 3. The "fast protrusion reduction" algorithms we design in this section are applicable to any problem that uses protrusion reducer, and hence they are useful well beyond the scope of this paper. We give two algorithms for fast protrusion replacement, a randomized algorithm and a slightly slower deterministic algorithm.

The Randomized Fast Protrusion Replacer. We now describe an algorithm that we call the *Randomized Fast Protrusion Replacer (RFPR)*. The algorithm works for parameterized graph problems Π that have a protrusion replacer, takes as input an instance (G, k) and outputs another instance (G', k'). Just as a normal protrusion replacer, the RFPR is actually a family of algorithms with one algorithm for each value of the integer r. We describe how the algorithm proceeds for a fixed value of r. Let r' be the smallest integer such that the protrusion replacer for Π replaces r-protrusions of size at least r'.

The RFPR proceeds as follows. We select a random partition of $V(G)$ into $r + 1$ sets $X_1, X_2, \ldots X_{r+1}$. For every $i \leq r + 1$ we compute the connected components of $G[X_i]$ and add these components to a collection $\mathcal{C}'$. This results in a partition of $V(G)$ into $\mathcal{C}' = C_1', C_2', \ldots C_{t'}'$. Now, discard every component C_i' such that $N(C_i') > r$ and every component C_i such that $\mathbf{tw}(G[N[C_i]]) > r$. Discarding all of these components can be done in linear time - the only computationally hard step is to check whether the treewidth of the components is at most r, this can be done in linear time using Bodlaenders's algorithm [11]. Let $\mathcal{C}^* = C_1^*, \ldots, C_{t^*}^*$ be the remaining components.

For every $C_i^* \in \mathcal{C}^*$, $N[C_i^*]$ is a r-protrusion in G. However some, if not all of the components in $\mathcal{C}^*$ could have less than r' vertices and so the protrusion replacer can't reduce them. However it could be possible to group some components in $\mathcal{C}^*$ with the same neighbourhood together such that their union is a protrusion that is large enough to be reduced. From $\mathcal{C}^*$ we will compute a collection $\mathcal{R}$ of disjoint vertex sets such that for every $R \in \mathcal{R}$, $N[R]$ is an r-protrusion in G of size at least r'. Our aim is to compute such a set with $|\mathcal{R}|$ being large. For every component $C_i^* \in \mathcal{C}^*$ of size at least r' we add C_i^* to $\mathcal{R}$ and remove C_i^* from $\mathcal{C}^*$. Let $\mathcal{C} = C_1 \ldots C_t$ be the remaining components. All components in $\mathcal{C}$ have size at most r'. Set R_{big} to be the number of components $C_i^* \in \mathcal{C}^*$ on at least r' vertices that are added to $\mathcal{R}$.

Now we partition $\mathcal{C}$ into groups according to the neighbourhood of the components. Specifically we compute a partition of $\mathcal{C}$ into $\mathcal{Z}_1, \ldots \mathcal{Z}_q$ such that for every pair $C_i \in \mathcal{C}$, $C_{i'} \in \mathcal{C}$ such that $N(C_i) = N(C_{i'})$, C_i and $C_{i'}$ are in the same $\mathcal{Z}_j$, while for every pair $C_i \in \mathcal{C}$, $C_{i'} \in \mathcal{C}$ such that $N(C_i) \neq N(C_{i'})$ we have $C_i \in \mathcal{Z}_j \rightarrow C_{i'} \notin \mathcal{Z}_j$. Such a partition can be computed in time $O(nr)$ because every component in $\mathcal{C}$ has at most r neighbours; First we sort the neighbor lists of each component according to some ordering of the vertex set, for example an arbitrary labelling of the vertices from 1 to n. Then we do r stable bucket sorts on $\mathcal{C}$ sorting the components first on their first neighbour, then their second neighbor, etc.

Having computed the partitioning $\mathcal{Z}_1, \ldots \mathcal{Z}_q$ we now compute $\mathcal{R}$ as follows. As long as there is a $\mathcal{Z}_i$ such that $\sum_{C_j \in \mathcal{Z}_i} |C_j| \geq r'$ select a minimal collection $\mathcal{Z} \subseteq \mathcal{Z}_i$ such that $\sum_{C_j \in \mathcal{Z}} |C_j| \geq r'$. Add $\bigcup_{C_j \in \mathcal{Z}} C_j$ to $\mathcal{R}$ and remove the components of $\mathcal{Z}$ from $\mathcal{Z}_i$. This procedure can easily be implemented in linear time. This concludes the construction of $\mathcal{R}$.

Given $\mathcal{R}$ we proceed as follows, for a set $R \in \mathcal{R}$ we run the protrusion replacer for Π on (G, k) with protrusion $N[R]$. The protrusion replacer outputs an equivalent instance (G^*, k^*) with $|V(G^*)| < |V(G)|$. Here G^* is a graph where R has been replaced by a smaller protrusion

R'. Since all the sets in $\mathcal{R}$ are disjoint, the other sets in $\mathcal{R}$ are now r-protrusions in G^* of size at least r'. Thus we can run the protrusion replacer on all the sets in $\mathcal{R}$. This takes time $\sum_{R \in \mathcal{R}} O(|R|) = O(n)$. Let (G', k') be the instance obtained after running the protrusion replacer on all the sets in $\mathcal{R}$. The RFPR outputs the instance (G', k'). We collect a few simple facts about the RFPR in the following lemma.

Lemma 6. *Given an instance (G, k), the RFPR runs in time $O(n+m)$, computes a collection $\mathcal{R}$ of protrusions and and outputs an equivalent instance (G', k'), such that $|V(G')| \leq |V(G)| - |\mathcal{R}|$. Furthermore $\mathcal{R} \geq R_{big} + \left\lceil \sum_{i \leq q} \frac{\sum_{C \in \mathcal{Z}_i} |C| - (r'-1)}{2(r'-1)} \right\rceil$.*

Proof. The instances (G, k) and (G', k') are equivalent because (G', k') is obtained from (G, k) by repetitive applications of a protrusion replacer. In the description of the algorithm we made sure that each individual stage of the algorithm runs in linear time. Finally, each application of the protrusion replacer reduces the size of the graph by at least one. We apply the protrusion replacer $|\mathcal{R}|$ times. Hence $|V(G')| \leq |V(G)| - |\mathcal{R}|$.

Finally, when the RFPR selects a minimal collection $\mathcal{Z} \subseteq \mathcal{Z}_i$ such that $\sum_{C_j \in \mathcal{Z}} |C_j| \geq r'$, since each $C_j \in \mathcal{Z}_i$ has size at most r' it follows that $\sum_{C_j \in \mathcal{Z}} |C_j| \leq 2(r'-1)$. Thus every time we add a set to $\mathcal{R}$, $\sum_{C \in \mathcal{Z}_i} |C|$ decreases by at most $2(r'-1)$. At the end when we can not add more sets to $\mathcal{R}$ we have that for every i, $\sum_{C \in \mathcal{Z}_i} |C| \leq r'$. This proves the last part of the statement of the lemma. $\qquad \square$

Analyzing the Randomized Fast Protrusion Replacer. We now analyze how many vertices the Fast Protrusion Replacer reduces the instance by. To that end we need to define the notion protrusion covers.

Definition 2. *An (a, b, r)-protrusion cover in a graph G is a collection $\mathcal{Z} = Z_1, \ldots, Z_t$ of sets such that for every i, $N[Z_i]$ is a r-protrusion in G and $a \leq |Z_i| \leq b$, and for every $i \neq j$, $Z_i \cap Z_j = \emptyset$ and there are no edges from Z_i to Z_j. The size of $\mathcal{Z}$ is $|\mathcal{Z}|$.*

Lemma 7. *Let Π be a problem that has a protrusion replacer which replaces r-protrusions of size at least r', and let $s \geq r' \cdot 2^r$. If G is a graph with a $(s, 6s, r)$-protrusion cover $\mathcal{X}$, then if the RFPR is run on (G, k), with probabilty at least $1 - e^{-\frac{|\mathcal{X}|}{8(r+1)^{6s}}}$ the output instance (G', k') satisfies $|V(G)| - |V(G')| \geq \frac{|\mathcal{X}|}{4(r+1)^{6s}}$.*

Proof. By Lemma 6 the RFPR computes a set $\mathcal{R}$ of protrusions and $|V(G)| - |V(G')| \geq |\mathcal{R}|$. Thus it is sufficient to show that with high probablility, $\mathcal{R} \geq \frac{|\mathcal{X}|}{4(r+1)^{6s}}$. Define $\overline{X} = V(G) \setminus \bigcup_{X \in \mathcal{X}} X$. Since no edge goes between different sets in $\mathcal{X}$ we have that for every $X \in \mathcal{X}$, $N(X) \subseteq \overline{X}$. The only randomized step of the RFPR is the initial partitioning of $V(G)$ into sets $X_1, \ldots X_{r+1}$. We may think of this partitioning step as selecting a random coloring of $V(G)$ with colors from $\{1, \ldots, r+1\}$.

We say that a set X in $\mathcal{X}$ *succeeds* if all vertices in X are colored with the same color, and no vertex of $N(X)$ is colored with that color. Since every set $X \in \mathcal{X}$ has at most r neighbours we have that the probability that X succeeds given any coloring of $\overline{X}$ is at least $\frac{1}{(r+1)^{|X|}}$. Hence the expected number of sets $X \in \mathcal{X}$ that succeed is at least $\frac{|\mathcal{X}|}{(r+1)^{6s}}$. Suppose t sets succeed. We prove that the set $\mathcal{R}$ constructed by the Randomized Fast Protrusion Replacer has size at least $t/2$.

For each set X that succeeds, the connected components of X are added to $\mathcal{C}'$, and since they all have treewidth at most r and have at most r neighbors, none of them are discarded. Hence the connected components of X are all added to $\mathcal{C}^*$. Since $|Z| \geq r' \cdot 2^r$, if we group the connected components of Z by their neighbourhood, at least one group has combined size at least r'. If

14

this group contains a connected component on at least r' vertices then this component is added to $\mathcal{R}$ directly and X contributes one to R_{big}. If this group does not contain any components of size at least r' then the group is added in its entirety to some set $\mathcal{Z}_i$. In this case the group contributes at least r' to $\sum_{C \in \mathcal{Z}_i} |C|$. By Lemma 6, $\mathcal{R} \geq R_{big} + \left\lceil \sum_{i \leq q} \frac{\sum_{C \in \mathcal{Z}_i} |C| - (r'-1)}{2(r'-1)} \right\rceil$. Hence the total number of sets added to $\mathcal{R}$ is at least $t/2$.

Since the neighbourhoods of different sets in $\mathcal{X}$ may overlap there are dependencies between which sets succeed. However, given any coloring of $\overline{X}$ the success of different sets in $\mathcal{X}$ is independent, since whether X succeeds or not depends only on the color of vertices in X and the color of vertices in $N(X) \subseteq \overline{X}$. Thus for every coloring of $\overline{X}$ the number of sets that succeed is a sum of independent 0-1 variables taking value 1 with probability at least $\frac{1}{(r+1)^{|X|}}$. Standard Chernoff bounds for the binomial distribution show that if T is a sum of n independent 0-1 variables taking value 1 with probabily p, then $P[X \leq np/2] < e^{-\frac{np}{8}}$. Plugging this in for the number of sets in $\mathcal{X}$ that succeed yields that the probability that $|\mathcal{R}| \leq \frac{|\mathcal{X}|}{4(r+1)^{6s}}$ is at most $e^{-\frac{|\mathcal{X}|}{8(r+1)^{6s}}}$. $\square$

The Deterministic Fast Protrusion Replacer We prove that the RFPR can be made deterministic at the cost of a $\log n$ factor in the running time. The only randomized step of the RFPR is the initial step where the vertices of G are partitioned into $r + 1$ sets $X_1, \ldots X_{r+1}$. We may think of this partitioning step as selecting a random coloring of $V(G)$ with colors from $\{1, \ldots, r + 1\}$. The main difference between the randomized and the deterministic Fast Protrusion Replacer is how this coloring is chosen. The Deterministic Fast Protusion Replacer only partitions $V(G)$ in two sets X_1 and X_2 - this corresponds to coloring the vertices with colors 1 and 2. To describe the colorings the Deterministic Fast Protrusion Replacer (DFPR) uses we use the notion of *universal sets*.

Definition 2 ([65])**.** *A (n, t)-universal set $\mathcal{P}$ of a ground set U on n elements is a collection $\mathcal{P}$ of subsets of U such that for every set $S \subseteq U$ and set $S' \subseteq S$ there is a set $P \in \mathcal{P}$ such that $P \cap S = S'$.*

Theorem 6 ([65])**.** *There is a deterministic algorithm with running time $O(2^{t+o(t)} n \log n)$ that constructs an (n, t)-universal set $\mathcal{P}$ such that $|\mathcal{P}| = 2^{t+o(t)} \log n$.*

The DFPR has two parameters, r and s, instead of just one parameter r. It constructs a $(n, 6s + r)$-universal set $\mathcal{P}$ in time $O(2^{6s+r+o(6s+r)} n \log n) = O(2^{20s} n \log n)$ and selects the first set $P \in \mathcal{P}$. It sets $X_1 = P$, $X_2 = V(G) \setminus P$ and then it proceeds just as the RFPR would. For a fixed set $P \in \mathcal{P}$ this takes linear time and will reduce (G, k) to an equivalent instance (G', k'). Choosing different sets $P \in \mathcal{P}$ results in different output instances (G', k'). The DFPR tries all possible choices for $P \in \mathcal{P}$ and then finally outputs the instance (G', k') that maximizes $|V(G)| - |V(G')|$. The total time taken by the DFPR is $O((2^{20s} n \log n) + |\mathcal{P}| \cdot O(n + m) = O((2^{20s}(n + m) \log n)$. This proves the following lemma.

Lemma 8. *Given an instance (G, k), the DFPR runs in time $O((2^{20s}(n + m) \log n)$, computes a collection $\mathcal{R}$ of protrusions and outputs an equivalent instance (G', k'), such that $|V(G')| \leq |V(G)| - |\mathcal{R}|$.*

We now give a lemma analogous to Lemma 7 for the DFPR.

Lemma 9. *Let Π be a problem that has a protrusion replacer which replaces r-protrusions of size at least r', and let $s \geq r' \cdot 2^r$. If G is a graph with a $(s, 6s, r)$-protrusion cover $\mathcal{X}$, then if the RFPR is run on (G, k), the output instance (G', k') satisfies $|V(G)| - |V(G')| \geq \frac{|\mathcal{X}|}{2^{20s} \log n}$.*

15

Proof. In the proof of Lemma 7 we showed that $|V(G)| - |V(G')|$ is lower bounded by the number of sets that succeeds. Since each set $X \in \mathcal{X}$ has size at most $6s$ and $|N[X]| \leq 6s + r \leq 7s$ it follows that for every $X \in \mathcal{X}$ there is some coloring set $P \in \mathcal{P}$ that makes X succeed. Hence there is a coloring $P \in \mathcal{P}$ that makes at least $\frac{|\mathcal{X}|}{|\mathcal{P}|} \geq \frac{|\mathcal{X}|}{2^{r+6s+o(r+6s)} \log n}$ sets succed. In the proof of Lemma 7 we showed that $|V(G)| - |V(G')|$ is at least half the number of succeeding sets. Since $2 \cdot 2^{r+6s+o(r+6s)} \log n \leq 2^{20s}$ we have $|V(G)| - |V(G')| \geq \frac{|\mathcal{X}|}{2^{20s} \log n}$. $\qquad\square$

We now proceed to prove that if G has a protrusion decomposition such that a linear fraction of the vertices appear in large enough r-protrusions then with high probability the Randomized Fast Protrusion Replacer will reduce G by a linear fraction of its vertices. To that end we need to have a closer look at the relationship between protrusion decompositions and protrusion covers.

Protrusion Covers from Protrusion Decompositions. First we prove that in a graph of small treewidth we can always find protrusion covers with large size.

Lemma 10. *There exists a constant c such that for any integers $n \geq s > b \geq 2$ and n-vertex graph G of treewidth b, G has a $(s, 6s, 2(b+1))$ cover of size at least $\frac{n}{122s}$.*

Proof. Let $(T, \mathcal{B})$ be a nice tree-decomposition of G of width b. For a subset $Q \subseteq V(T)$ by $P(Q)$ we denote $\cup_{q \in Q} B_q$. For a rooted tree T, and a vertex $v \in T$, a component C of $T \setminus \{v\}$ is said to be below v if all vertices of C are descendants of v in T. We start by constructing a set $S \subseteq V(T)$ and a collection $Q_1, \ldots, Q_{|S|}$ of connected components of $T \setminus S$ using the following greedy procedure.

Let r be the root of T. In the beginning $S = \emptyset$ and $T^r = T$. We maintain a loop invariant that T^r is the connected component of $T \setminus S$ that contains r. Now, at step i of the greedy procedure we pick a lowermost vertex v_i in $V(T^r)$ such that there is a connected component Q_i of $T^r \setminus \{v_i\}$ *below* v_i such that $|P(Q_i)| \geq 3s + 7(b+1)$. Now we add v_i to S and update T^r accordingly. The procedure terminates when no vertex v in T^r has this property. In particular, if for any $v \in T^r$, every component Q of $T^r \setminus \{v\}$ below v, $|P(Q)| < 3s + 7(b+1)$, the procedure terminates. Since $(T, \mathcal{B})$ is a nice tree decomposition, we have that for any vertex $v \in T_r$ and parent u of v, if C_v and C_u are the components of $T^r \setminus \{v\}$ and $T^r \setminus \{u\}$ maximizing $|P(C_v)|$ and $|P(C_u)|$ respectively, then $|P(C_u)| \leq 2|P(C_v)|$. Hence we know that for every component Q of $T \setminus S$, $|P(Q)| < 6s + 14(b+1) \leq 20s$. This bound holds both for the components included in the collection $Q_1, \ldots, Q_{|S|}$ and the ones that do not.

Having constructed S and $Q_1, \ldots, Q_{|S|}$ we let $S' = \textbf{LCA-closure}(S)$. By Lemma 1 we have $|S'| \leq 2|S|$. Let $S^* = S' \setminus S$. Since $|S^*| \leq |S|$, at most $\frac{|S|}{2}$ of the components $Q_1, \ldots, Q_{|S|}$ contain at least two vertices of S^*. This implies that at least $\frac{|S|}{2}$ of the components $Q_1, \ldots, Q_{|S|}$ contain at most one vertex of S^*. Without loss of generality, let $Q_1, \ldots, Q_{|S|/2}$ contain at most one vertex of S^* each. For every $i \leq |S|/2$, if Q_i contains no vertex of S^* then $Q_i' = Q_i$ is a component of $Q \setminus S'$ with $|P(Q_i')| \geq 3s + 7(b+1) \geq s + 2(b+1)$. If Q_i contains one vertex v of S^*, since v has degree at most 3 and $|P(Q_i)| \geq 3s + b$, $Q_i \setminus \{v\}$ has at least one component Q_i' with $|P(Q_i')| \geq s + 2(b+1)$. Thus we have constructed a set S' and a collection of components $Q_1', \ldots, Q_{|S|/2}'$ of $T \setminus S'$ of size at least $s + 2(b+1)$. By Lemma 1 every Q_i' has at most two neighbors in T.

We make a collection $\mathcal{Z}$ as follows. For every $i \leq |S|/2$ let $Z_i = P(Q_i') \setminus P(S')$. Since Q_i' has at most two neighbors in T it follows that $N[Z_i]$ is a $2(b+1)$-protrusion and that $|Z_i| \geq s + 2(b+1) - 2(b+1) = s$. We have already shown that $|Q_i'| \leq 20s$ so $|Z_i| \leq 20s$ as well. Hence $\mathcal{Z}$ is in fact a $(s, 6s, 2(b+1))$-protrusion cover of G. It remains to lower bound $|\mathcal{Z}|$. We have that $|\mathcal{Z}| = |S|/2$. Furthermore we have that S, together with the connected components of $T \setminus S$ cover T. Since every bag has size at most $(b+1) \leq s$, $T \setminus S$ has at most

$2|S| + 1 \leq 3|S|$ connected components and for every component Q of $T \setminus S$, $|P(Q)| \leq 20s$ we have that $|S|(b+1) + 3|S| \cdot 20s \geq n$. Since $s \geq b+1$ this implies that $|S| \geq \frac{n}{122s}$. $\qquad \square$

Lemma 11. *If G has an (α, β)-protrusion decomposition, then for every $s > \beta$, G has a $(s, 6s, 3(\beta+1))$-protrusion cover of size at least $\frac{n}{122s} - \alpha$.*

Proof. Let $R_0, \ldots R_t$ be an (α, β)-protrusion decomposition of G. At most α vertices are in R_0, and at most $\alpha \cdot s$ vertices are in sets R_i for $i \geq 1$ such that $|R_i| < s$. For each $i \geq 1$ such that $|R_i| \geq s$ we apply Lemma 10 and obtain a $(s, 6s, 2(\beta+1))$-protrusion cover $\mathcal{Z}_i$ in $G[R_i]$. We let $\mathcal{Z}$ be the union of all the $\mathcal{Z}_i$'s constructed in this manner. For every $Z \in \mathcal{Z}_i$, $N_{G[R_i]}[Z_i]$ is a $2(\beta+1)$-protrusion in $G[R_i]$. However Z might have neighbors also in R_0. The number of neighbors of Z in R_0 is at most β and hence $N[Z]$ is a $3(\beta+1)$-protrusion in G. We conclude that $\mathcal{Z}$ is a $(s, 6s, 3(\beta+1))$-protrusion cover in G. The size of $\mathcal{Z}$ is at least $\frac{n - \alpha - \alpha \cdot s}{122s} \geq \frac{n}{122s} - \alpha$. $\qquad \square$

The Fast Protrusion Replacer Theorems We are now ready to prove our main results on Fast Protrusion Replacement.

Theorem 7 (Randomized Fast Protrusion Replacer Theorem). *Let Π be a problem that has a protrusion replacer that replaces r protrusions of size at least r', and let s and β be constants such that $r \geq 3(\beta+1)$ and $s \geq 2^r \cdot r'$. Given an instance (G, k) as input, the RFPR will run in time $O(n+m)$ and produce an equivalent instance (G', k') with $|V(G')| \leq |V(G)|$ and $k' \leq k$. If additionally G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$, then with probability at least $1 - e^{-\frac{n}{2000s(r+1)^{6s}}}$ we have $|V(G)| - |V(G')| \geq \frac{n}{1000(r+1)^{6s}}$.*

Proof. The first part of the statement follows directly from Lemma 6. If G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$, then by Lemma 11, G has a $(s, 6s, 3(\beta+1))$-protrusion cover $\mathcal{X}$ of size at least $\frac{n}{122s} - \alpha \geq \frac{n}{244s}$. Plugging $\mathcal{X}$ into Lemma 7 yields that with probability at least $1 - e^{-\frac{|\mathcal{X}|}{8s(r+1)^{6s}}} \geq 1 - e^{-\frac{n}{2000s(r+1)^{6s}}}$ we have $|V(G)| - |V(G')| \geq \frac{|\mathcal{X}|}{4(r+1)^{6s}} \geq \frac{n}{1000(r+1)^{6s}}$. $\qquad \square$

Theorem 8 (Deterministic Fast Protrusion Replacer Theorem). *Let Π be a problem that has a protrusion replacer that replaces r protrusions of size at least r', and let s and β be constants such that $r \geq 3(\beta+1)$ and $s \geq 2^r \cdot r'$. Given an instance (G, k) as input, the DFPR will run in time $O(2^{20s} \cdot (n+m)\log n)$ and produce an equivalent instance (G', k') with $|V(G')| \leq |V(G)|$ and $k' \leq k$. If additionally G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$ then we have $|V(G)| - |V(G')| \geq \frac{n}{244 \cdot 2^{20s}\log n}$.*

Proof. The first part of the statement follows directly from Lemma 8. If G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$, then by Lemma 11, G has a $(s, 6s, 3(\beta+1))$-protrusion cover $\mathcal{X}$ of size at least $\frac{n}{122s} - \alpha \geq \frac{n}{244s}$. Plugging $\mathcal{X}$ into Lemma 9 yields that $|V(G)| - |V(G')| \geq \frac{|\mathcal{X}|}{2^{20s}\log n} \geq \frac{n}{244 \cdot 2^{20s}\log n}$. $\qquad \square$

It can be shown that Theorem 7 could replace the simple protrusion reduction algorithm of Lemma 2 and make thus Algorithm 2 run in linear time. However we are first going to refine Algorithm 2 even further so that it becomes simultaneously single exponential parameterized algorithm and an approximation algorithm for $\mathcal{F}$-DELETION for all connected $\mathcal{F} \in \mathscr{F}$. To that end we develop the notion of lossless protrusion replacement.

5 Lossless Protrusion Replacement

In this section we develop the notion of lossless protrusion replacement. We consider $CMSO$ vertex subset problems. In a MIN-CMSO vertex subset problem, Π, we are given a graph G as input. The objective is to find a set $S \subseteq V(G)$ minimizing $|S|$ such that such that the CMSO-expressible predicate $P_\Pi(G, S)$ is satisfied. Similarly, in a MAX-CMSO vertex subset problem, Π, we are given a graph G as input. The objective is to find a set $S \subseteq V(G)$ maximizing $|S|$ such that the CMSO-expressible predicate $P_\Pi(G, S)$ is satisfied. Given a MIN-CMSO (MAX-CMSO) vertex subset problem, Π and an input graph G to Π, by $OPT(G)$ we denote the size of the smallest (largest) set S such that the CMSO-expressible predicate $P_\Pi(G, S)$ is satisfied. Next we define the notion of a lossless protrusion replacer. A lossless protrusion replacer is essentially a protrusion replacer that reduces protrusions in such a way that any feasible solution to the reduced instance can be changed into a feasible solution of the original instance without changing the gap between the feasible solution and the optimum. The notion of lossless protrusion replacement is central in our approximation algorithms.

Definition 3 (Lossless Protrusion Replacer). *A lossless protrusion replacer for* MIN-CMSO *(*MAX-CMSO*) vertex subset problem* Π *is a family of algorithms, with one algorithm for every constant* r. *The* r'*th algorithm has the following specifications. There exists a constant* r' *(which depends on* r*) such that given an instance* G *and an* r*-protrusion* X *in* G *of size at least* r'*, the algorithm runs in time* $O(|X|)$ *and outputs an instance* G' *with the following properties.*

- G' *is obtained from* G *by replacing* X *by a* r*-bounderied graph* X' *with less than* r' *vertices and thus* $|V(G')| < |V(G)|$.

- $OPT(G') \leq OPT(G)$.

- *There is an algorithm that runs in* $O(|X|)$ *time and given a feasible solution* S' *to* G' *outputs a set* $X^* \subseteq X$ *such that* $S = (S' \setminus X') \cup X^*$ *is a feasible solution to* G *and* $|S| \leq |S'| + OPT(G) - OPT(G')$.

We would like to give sufficient conditions for a problem to have a lossless protrusion replacer. An ideal setting would be that every graph optimization problem that has finite integer index when parameterized by the size of the optimal solution has a lossless protrusion replacaer. Unfortunately such a theorem seems to be out of reach, and it is quite possible that this is not true. However, in [17] a sufficient condition is given for a $CMSO$ vertex subset problem to have finite integer index. This condition is called *strong monotonicity* and it is proved that every $CMSO$ vertex subset problem that is stronly monotone has finite integer index and hence has a protrusion replacer. It turns out that strong monotonicity is a sufficient condition for a $CMSO$ vertex subset problem to not only have a protrusion replacer, but also a lossless protrusion replacer. We now prove this fact.

Let Π be a MIN-CMSO problem and $\mathcal{F}_t$ be the set of pairs (G, S) where G is a t-bounderied graph and $S \subseteq V(G)$. For a t-bounderied graph G we define the function $\zeta_G : \mathcal{F}_t \to \mathbb{N} \cup \{\infty\}$ as follows. For a pair $(G', S') \in \mathcal{F}_t$, if there is no set $S \subseteq V(G)$ such that $P_\Pi(G \oplus G', S \cup S')$ holds, then $\zeta_G((G', S')) = \infty$. Otherwise $\zeta_G((G', S'))$ is the size of the smallest $S \subseteq V(G)$ such that $P_\Pi(G \oplus G', S \cup S')$ holds. If Π is a MAX-CMSO problem then we define $\zeta_G((G', S'))$ to be the size of the largest $S \subseteq V(G)$ such that $P_\Pi(G \oplus G', S \cup S')$ holds. If there is no set $S \subseteq V(G)$ such that $P_\Pi(G \oplus G', S \cup S')$ holds, then $\zeta_G((G', S')) = \infty$.

Definition 3 ([17]). *A* MIN-CMSO *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*-bounderied graph* G*, there is a subset* $S \subseteq V(G)$ *such that for every* $(G', S') \in \mathcal{F}_t$ *such that* $\zeta_G((G', S'))$ *is finite,* $P_\Pi(G \oplus G', S \cup S')$ *holds and* $|S| \leq \zeta_G((G', S')) + f(t)$.

Definition 4 ([17])**.** *A* MAX-CMSO *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 $(G', S') \in \mathcal{F}_t$ such that $\zeta_G((G', S'))$ is finite, $P_\Pi(G \oplus G', S \cup S')$ holds and $|S| \geq \zeta_G((G', S')) - f(t)$.*

Theorem 9. *Every* MIN-CMSO *or* MAX-CMSO *vertex subset problem Π, that is also strongly monotone admits a lossless protrusion replacer.*

Before proving the theorem we will need an auxiliary lemma.

Lemma 12. *If a graph G contains an r-protrusion X where $|X| > c > 0$, then it also contains a $(2r + 1)$-protrusion Y where $c < |Y| \leq 2c$. Moreover, given X we can compute Y and a tree decomposition of Y of width $\leq 2r$ in $O(|X|)$ time.*

Proof. Let $(T, \mathcal{X})$ be a nice tree decomposition of $G[X]$ rooted at a node r. We can compute $(T, \mathcal{X})$ from $G[X]$ in time $O(|X|)$ using Bodlaender's algorithm [11]. If $|X| \leq 2c$, we are done. Given a vertex x of the rooted tree T, we denote by $\mathcal{D}_T(x)$ the subset of $V(T)$ containing x and all its descendants in T. Let B_T be the set containing each vertex x of T with the property that the vertices appearing in $\bigcup_{y \in \mathcal{D}_T(x)} X_y$ (i.e. the vertices of the nodes corresponding to x and its descendants) are more than c. As $|X| \geq 2c$, B_T is a non-empty set. We choose b to be a member of B_T whose descendants do not belong in $B_{T'}$. This choice of b ensures that $c < |\bigcup_{y \in \mathcal{D}(b)} X_y| \leq 2c$. We define $Y = \partial_G X \cup \bigcup_{y \in \mathcal{D}_T(b)} X_y$. As $G[Y]$ is an induced subgraph of X it follows that $\mathbf{tw}(G[Y]) \leq r$. Furthermore $\partial_G(Y) \subseteq \partial_G X \cup X_b$, therefore Y is a $(2r + 1)$-protrusion of G. $\square$

Proof of Theorem 9. We prove the theorem for MIN-CMSO problems; the proof for MAX-CMSO problems is similar. Let Π be a monotone MIN-CMSO problem. We define a partial order $\leq_\Pi$ on pairs (G, S) such that G is a t-boundaried graph and $S \subseteq V(G)$. We say that $(G, S) \leq_\Pi (G', S')$ if for every (G_3, S_3), $P_\Pi(G \oplus G_3, S \cup S_3) \to P_\Pi(G' \oplus G_3, S' \cup S_3)$. We say that that $(G, S) \equiv_\Pi (G', S')$ if $(G, S) \leq_\Pi (G', S')$ and $(G', S') \leq_\Pi (G, S)$. Clearly $\equiv_\Pi$ is an equivalence relation and since P_Π is a CMSO-expressible predicate it follows from [21, 29] that for every fixed t, $\equiv_\Pi$ has finitely many equivalence classes. Thus there exists finite set $\mathcal{S}$ of pairs (G_R, S_R) such that for every (G, S) there is a $(G_R, S_R) \in \mathcal{S}$ such that $(G, S) \equiv (G_R, S_R)$. We say that a pair (G, S) is *bad* if there is no (G', S') such that $P_\Pi(G \oplus G', S \cup S')$ holds. A pair that is not bad is called *useful*. Let $\mathcal{U}$ be the set of all useful pairs in $\mathcal{S}$.

For a graph G and pair $(G_R, S_R) \in \mathcal{U}$ define $\gamma_G(G_R, S_R)$ to be the size of the smallest set $S \subseteq V(G)$ such that $(G_R, S_R) \leq_\Pi (G, S)$. If no such set S exists, $\gamma_G(G_R, S_R) = \infty$. We now prove that for any G, the maximum finite value of γ_G and the minimum (finite) value of γ_G differs by at most $f(t)$. Let $S \subseteq V(G)$ be the set such for every $(G', S') \in \mathcal{F}_t$ such that $\zeta_G((G', S'))$ is finite, $P_\Pi(G \oplus G', S \cup S')$ holds and $|S| \leq \zeta_G((G', S')) + f(t)$. Consider a useful pair $(G_R, S_R) \in \mathcal{U}$ such that $\gamma_G(G_R, S_R)$ is finite. Then there exists a set $S' \subseteq V(G)$ of size $\gamma_G(G_R, S_R)$ such that $(G_R, S_R) \leq_\Pi (G, S')$. Since $(G, S') \leq_\Pi (G, S)$ and S' is the smallest set such that $(G_R, S_R) \leq_\Pi (G, S')$ it follows that $|S'| \leq |S|$. On the other hand since (G_R, S_R) is useful there exists some (G^*, S^*) such that $P_\Pi(G_R \oplus G^*, S_R \cup S^*)$ holds. Then $P_\Pi(G \oplus G^*, S' \cup S^*)$ holds as well and hence $\zeta_G((G^*, S^*)) \leq |S'|$. Since $\zeta_G((G^*, S^*))$ is finite it follows that $|S| \leq \zeta_G((G^*, S^*)) + f(t) \leq |S'| + f(t)$. But this means that $|S| - f(t) \leq \gamma_G(G_R, S_R) \leq |S|$ and so the finite values of γ_G differ by at least $f(t)$. By the pigeon hole principle there exists a finite collection $\mathcal{R}$ of t-boundaried graphs such that for any t-boundaried G there is a $G_R \in \mathcal{R}$ and a constant $c_R \geq 0$ such that for every useful pair (G', S'), $\gamma_G(G', S') = \gamma_{G_R}(G', S') + c_R$. We call $\mathcal{R}$ a *set of representatives* for (Π, t).

For every integer c we define a relation $\prec_c$ on t-boundaried graphs. We say that $G_1 \prec_c G_2$ if for every useful pair (G, S), $\gamma_{G_1}(G, S) + c = \gamma_{G_2}(G, S)$. Observe that if $G_1 \prec_c G_2$ then

$G_2 \prec_{-c} G_1$. Also, we have just shown that for every G there is a $G_R \in \mathcal{R}$ and constant $c_R \geq 0$ such that $G_R \prec_{c_R} G$. We now show that if $G \prec_c G'$ then for any t-boundaried graph G_3 and feasible solution S to Π on $G \oplus G_3$, there is a set $X^* \subseteq V(G')$ depending only on $S \cap V(G)$ and G such that $S' = X^* \cup S \setminus V(G)$ is also a feasible solution to Π on $G' \oplus G_3$ and $|S'| \leq |S| + c$.

Let $G \prec_c G'$ and consider a t-boundaried G_3 and a feasible solution S of Π on $G \oplus G_3$. Let $S_G = S \cap V(G)$ and $S_3 = S \setminus S_G$. (G, S_G) is a useful pair and so there is a pair $(G_R, S_R) \in \mathcal{U}$ such that $(G_R, S_R) \equiv_\Pi (G, S_G)$. Thus $\gamma_G(G_R, S_R) \leq |S_G|$ and hence $\gamma_{G'}(G_R, S_R) \leq |S_G| + c$. There is a set $X^* \subseteq V(G')$ such that $(G_R, S_R) \leq_\Pi (G', X^*)$ and $|X^*| \leq |S_G| + c$. The set X^* depends solely on (G_R, S_R) which depends solely on $S \cap V(G)$ and G. Furthermore, since $(G_R, S_R) \leq_\Pi (G', X^*)$ we have that $S' = X^* \cup S_3$ is also also a feasible solution to Π on $G' \oplus G_3$ and $|S'| \leq |S_G| + c + |S_3| \leq |S| + c$.

We can now describe the lossless protrusion replacer for the problem Π. For parameter r consider the set $\mathcal{R}$ of representatives for $(\Pi, 2(r+1))$. Let r' be the size of the largest graph in $\mathcal{R}$ plus one. The lossless protrusion replacer for Π will reduce r-protrusions of size at least r'. Given an r-protrusion X of size at least r' we find a $2(r+1)$ protrusion $Y \subseteq X$ such that $r' \leq |Y| \leq 2r'$. This can be done in $O(|X|)$ time by Lemma 12. Consider now the $2(r+1)$-boundaried graph $G^{\delta Y}_{Y \setminus \delta(Y)}$. There exists a $2(r+1)$-boundaried graph $G_R \in \mathcal{R}$ and constant $c_R \geq 0$ such that $G_R \prec_{c_R} G^{\delta Y}_{Y \setminus \delta(Y)}$. Furthermore since $|Y| \geq r'$ we have that $|V(G_R)| < |Y|$. The protrusion replacer outputs the graph G' obtained by replacing Y by G_R in G.

For every subset $S_R \subseteq V(G_R)$ such that the pair (G_R, S_R) is useful, the protrusion replacer stores a subset $S_Y \subset Y$ such that $(G_R, S_R) \leq_\Pi (G^{\delta Y}_{Y \setminus \delta(Y)}, S_Y)$. Since $G_R \prec_{c_R} G^{\delta Y}_{Y \setminus \delta(Y)}$ there is such a set S_Y of size at most $|S_R| + c$. Now, for any feasible solution S in G' let $S_R = S \cup V(G_R)$. The pair (G_R, S_R) is useful and so the lossless protrusion replacer outputs the set $X^* = S_Y$ which it has stored for S_R. Now $S' = S_Y \cup (S \setminus V(G_R))$ is a feasible solution to G because $(G_R, S_R) \leq_\Pi (G^{\delta Y}_{Y \setminus \delta(Y)}, S_Y)$. Furthermore, since $|S_Y| \leq |S_R| + c$ we have that $|S'| \leq |S_Y| + |S \setminus V(G_R)| \leq |S_R| + c + |S \setminus V(G_R)| \leq |S| + c$. Thus it remains to prove that $c \leq OPT(G) - OPT(G')$, or in other words that $OPT(G') \leq OPT(G) - c$.

However $G^{\delta Y}_{Y \setminus \delta(Y)} \prec_{-c_R} G_R$, and hence for an optimal solution S of $G = G^{\delta Y}_{Y \setminus \delta(Y)} \oplus G^\delta_{V(G) \setminus Y}(Y)$ there is a feasible solution S' in $G_R \oplus G^\delta_{V(G) \setminus Y}(Y)$ of size at most $|S| - c_R$. Hence $OPT(G') \leq OPT(G) - c$ and the theorem follows. $\qquad \square$

Inserting a lossless protrusion replacer instead of a normal protrusion replacer into the Fast Protrusion Replacer algorithms directly yields the following theorems.

Theorem 10. *Let Π be a minimization (maximization) problem that has a lossless protrusion replacer that replaces r protrusions of size at least r', and let s and β be constants such that $r \geq 3(\beta+1)$ and $s \geq 2^r \cdot r'$. Given an instance G as input, the Randomized Fast Protrusion Replacer will run in time $O(n+m)$ and produce an instance G' with $|V(G')| \leq |V(G)|$. Given any feasible solution S' to G' a feasible solution S of G of size at most (at least) $|S'| - OPT(G') + OPT(G)$ can be computed in $O(n+m)$ time. If additionally G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$, then with probability at least $1 - e^{-\frac{n}{2000s(r+1)^{6s}}}$ we have $|V(G)| - |V(G')| \geq \frac{n}{1000(r+1)^{6s}}$.*

Theorem 11. *Let Π be a minimization (maximization) problem that has a lossless protrusion replacer that replaces r protrusions of size at least r', and let s and β be constants such that $r \geq 3(\beta+1)$ and $s \geq 2^r \cdot r'$. Given an instance G as input, the Deterministic Fast Protrusion Replacer will run in time $O(2^{20s} \cdot (n+m) \log n)$ and produce an instance G' with $|V(G')| \leq |V(G)|$. Given any feasible solution S' to G' a feasible solution S of G of size at most (at least) $|S'| - OPT(G') + OPT(G)$ can be computed in $O(n+m)$ time. If additionally G has a (α, β)-protrusion decomposition such that $\alpha \leq \frac{n}{244s}$ then we have $|V(G)| - |V(G')| \geq \frac{n}{244 \cdot 2^{20s} \log n}$.*

6 Approximation and Fast Parameterized Algorithm for p-$\mathcal{F}$- DELETION

We are now ready to give the linear time, lossless variant of Lemma 2. Throughout this section $OPT(G)$ is the size of the smallest $\mathcal{F}$-deletion set of G, for the set $\mathcal{F}$ currently under consideration. First we give an auxiliary lemma analyzing an execution of the Lossless RFPR on a graph with an $\mathcal{F}$-deletion set S.

Lemma 13. *For every connected $\mathcal{F} \in \mathscr{F}$, there exist constants ρ, r, s, $c < 1$ and $\gamma > 0$ such that if we run the Lossless RFPR with parameters r, s on a graph G which has a $\mathcal{F}$ deletion set S which is not a ρ-cover, then with probability at least $1 - e^{-\gamma n}$ the output instance G' satisfies $V(G') \leq |V(G)|(1-c)$.*

Proof. If G has a $\mathcal{F}$-deletion set S' which is not an ρ-cover, it also has a inclusion minimal $\mathcal{F}$-deletion set S which is not an ρ-cover. Such a minimal S contains no isolated vertices and hence satisfies $N[S] \leq 2\sum_{v \in S} d(v) \leq 2\rho m$.

By Proposition 1, there exists a constant b such that $\mathbf{tw}(G \setminus S) \leq b$. By Lemma 5, G has a $(4(b+1)|N[S]|, 2(b+1))$-protrusion decomposition. Set $\beta = 2(b+1)$, $r = 3(\beta+1)$ and r' to be the smallest integer such that the lossless protrusion replacer will replace r-protrusions of size at least r'. Set $s = 2^r \cdot r'$. The protrusion decomposition of G is a $(4(b+1)|N[S]|, \beta)$-protrusion decomposition. By Theorem 10 there exist constants $0 < c < 1$ and $0 < \gamma$ such that if we run the Lossless RFPR on G and $4(b+1)|N[S]| \leq \frac{n}{244s}$ then with probability at least $1 - e^{-\gamma n}$, the output graph G' satisfies $|V(G)| - |V(G')| \geq c|V(G)|$. We show that there is a constant $\rho < \frac{1}{3}$ such that if S is not a ρ-cover, then $|N[S]| \leq \frac{n}{1000(b+1)s}$.

Since $\mathbf{tw}(G \setminus S) \leq b$ we have that $G \setminus S$ is $(b+1)$-degenerate. If S is not a ρ-cover then $m \leq n(b+1) + \sum_{v \in S} d(v) \leq n(b+1) + 2\rho m$. Rearranging yields that $N[S] \leq 2\rho m \leq n\frac{2\rho(b+1)}{1-2\rho} \leq n\rho 6(b+1)$. Choosing $\rho = 6000(b+1)^2 s$ yields that $|N[S]| \leq \frac{n}{1000(b+1)s}$. Hence, if S is not a ρ-cover then with probability at least $1 - e^{-\gamma n}$ the output instance G' of the Lossless RFPR satisfies $V(G') \leq |V(G)|(1-c)$. $\square$

Lemma 14. *For every connected $\mathcal{F} \in \mathscr{F}$ there is an algorithm that given a graph G, takes $O(n + m)$ time and outputs a graph G' such that $V(G') \leq V(G)$ and $OPT(G') \leq OPT(G)$. Given a $\mathcal{F}$-deletion set S' of G' the algorithm can compute an $\mathcal{F}$-deletion set S of G of size $|S'| + OPT(G) - OPT(G')$ in time $O(n + m)$. Furthermore there exist a constant $0 < \rho < 1$ such that with probability at least $\frac{1}{2}$, every $\mathcal{F}$-deletion set S' of G' is a ρ-cover of G.*

Proof. By Lemma 13 there exist constants ρ, r, s, $c < 1$ and $\gamma > 0$ such that if we run the Lossless RFPR with parameters r, s on a graph G which has a $\mathcal{F}$ deletion S which is not a ρ-cover, then with probability at least $1 - e^{-\gamma n}$ the output instance G' satisfies $V(G') \leq |V(G)|(1-c)$. We set these constants as guaranteed by Lemma 13.

The algorithm sets $G_1 := G$, $i = 1$ and enters a loop that proceeds as follows. The algorithm runs the Lossless RFPR on G_i with parameters r and s, let the output of the Lossless RFPR be G_{i+1}. If $|V(G_{i+1})| > |V(G_i)|(1-c)$ the algorithm halts and outputs G_i. Otherwise, the algorithm increments i and returns to the beginning of the loop.

The total time spent by the algorithm is upper bounded by a geometric series, and so the running time of the algorithm is $O(n + m)$. Similarly, by repeatedly applying Theorem 10 we can in linear time transform any $\mathcal{F}$-deletion set S_i of G_i back into a $\mathcal{F}$-deletion set S of G of size at most $|S'| + OPT(G) - OPT(G')$. It remains to prove that when the algorithm terminates, with probability at least $\frac{1}{2}$ we have that every $\mathcal{F}$-deletion set S' of G' is an ρ-cover of G.

The algorithm makes $t = O(\log n)$ calls to the Lossless RFPR. For $i \leq t+1$ let $n_i = |V(G_i)|$. In call i, by Lemma 13, if G_i has an $\mathcal{F}$-deletion set S which is not a ρ-cover then the probability

that $V(G_{i+1}) > V(G_i)(1 - c)$ is at most $e^{-\gamma n_i}$. By the union bound the probability that this occurs at some step i is $\sum_{i \le t} e^{-\gamma n_i}$. The n_i's are a decreasing geometric series and so for a sufficiently large (constant) $\bar{N}$ we have that if $n_t \ge N$ then $\sum_{i \le t} e^{-\gamma n_i} \le 2e^{-\gamma n_t} \le 1/2$.

Finally, if $n_t \le N$ then any non-empty set S is a $\frac{1}{N^2}$ cover, and so if $\rho > \frac{1}{N^2}$ we can adjust ρ to $\frac{1}{N^2}$. This proves the lemma. $\qquad \square$

We are now ready to give the algorithm which is the main engine behind both our $2^{O(k)}n$ time algorithm and the quadratic approximation algorithm for $\mathcal{F}$-DELETION for connected sets $\mathcal{F} \in \mathscr{F}$.

Randomized-$\mathcal{F}$-Deletion(G)

Set $G_1 := G$ and i $:= 1$

While (G_i is not $\mathcal{F}$-free) do as follows:

1. Apply Lemma 14 on G_i and obtain a new graph G_i'

2. Pick a vertex $u_i \in V(G_i')$ at random with probability $\frac{d_{G_i'}(u)}{2|E(G_i')|}$. Set $G_{i+1} := G_i' \setminus \{u_i\}$.

3. Increment i by 1.

Set $S_i = \emptyset$

For $j = i$ downto 2:

1. Set $S_{j-1}' := S_j \cup \{u_{j-1}\}$.
2. Apply Lemma 14 on G_j' and S_j' and obtain a set S_j.

Output $S := S_1$.

Figure 3: Randomized Algorithm for $\mathcal{F}$-DELETION for connected $\mathcal{F} \in \mathscr{F}$

We say that a *round* of Algorithm 3 is an iteration of the while-loop. Round x is the iteration when the value of i is x. The algorithm *suceeds* in round i if $OPT(G_i') = OPT(G_{i+1}) + 1$ and it *fails* in round i otherwise. The *number of rounds* of a run of Algorithm 3 is the maximum value i takes. We make a series of observations about Algorithm 3. For every i we have that $|V(G_i')| \le |V(G_i)|$ and $|V(G_{i+1})| < |V(G_i')|$. Hence we make the following observation.

Observation 1. *Algorithm 3 terminates after at most n rounds.*

The next observation follows directly from Lemma 14.

Observation 2. *The time taken in each round and each iteration of the for loop is $O(n + m)$.*

Next we prove that the algorithm always outputs feasible solutions.

Observation 3. *Algorithm 3 outputs an $\mathcal{F}$-deletion set of G.*

Proof. Let t be number of rounds. We have that G_t is F-free and so $S_t = \emptyset$ is a $\mathcal{F}$-deletion set of G_t. If S_j is a $\mathcal{F}$-deletion set of G_j then $S_j' = S_j \cup \{u_{j-1}\}$ is a $\mathcal{F}$-deletion set of G_{j-1}'. Then, by Lemma 14, S_{j-1} is a $\mathcal{F}$-deletion set of G_{j-1}. Hence, by downward induction on j, S_1 is a $\mathcal{F}$-deletion set of $G_1 = G$. $\qquad \square$

Next we upper bound the size of the output solution S.

Lemma 15. *Let p be the number of rounds in which Algorithm 3 fails. Then the size of the output solution S is $|S| = OPT(G) + p$.*

22

Proof. For every x, define f_x to be the number of rounds $i \geq x$ such that the algorithm fails in round i. Let t be the be number of rounds. We prove by downward induction on i that $|S_i| = OPT(G_i) + f_i$. Since $|S_t| = |f_t| = |OPT(G_t)| = 0$ this clearly holds for t. Consider now some $i < t$ such that the equation holds for $i + 1$.

If the algorithm succeeded in round i we have that $|S_i'| = |S_{i+1}| + 1$, that $OPT(G_i') = OPT(G_{i+1}) + 1$ and that $f_i = f_{i+1}$ hence $|S_i'| = |S_{i+1}| + 1 = OPT(G_{i+1}) + f_{i+1} + 1 = OPT(G_i') + f_i$. On the other hand if the algorithm fails in round i we have $|S_i'| = |S_{i+1}| + 1$, that $OPT(G_i') = OPT(G_{i+1})$ and that $f_i = f_{i+1} + 1$. Then $|S_i'| = |S_{i+1}| + 1 = OPT(G_{i+1}) + f_{i+1} + 1 = OPT(G_i') + f_i$. Hence in both cases we have that $|S_i'| = OPT(G_i') + f_i$. By Lemma 14 we have that $|S_i| = |S_i'| + OPT(G_i) - OPT(G_i') = OPT(G_i) + f_i$. This concludes the proof. $\square$

Now we lower bound the success probability of any round i of Algorithm 3.

Lemma 16. *There is a constant $p > 0$ such that the probability that Algorithm 3 succeeds in any given round i is at least p.*

Proof. By Lemma 14 there is a constant ρ such that with probability $1/2$, every $\mathcal{F}$-deletion set of G_i' is a ρ-cover. Let S^* be an optimal $\mathcal{F}$-deletion set of G_i'. If $u_i \in S^*$ then $S^* \setminus u_i$ is an optimal $\mathcal{F}$-deletion set of $G_i' \setminus u_i = G_{i+1}$. So if $u_i \in S^*$ then the algorithm succeds in round i. If S^* is a ρ-cover of G_i' then the probability that $u_i \in S^*$ is at least ρ. Hence the probability that every $\mathcal{F}$-deletion set of G_i' is a ρ-cover *and* $u_i \in S^*$ is at least $p = \rho/2$. $\square$

In each round Algorithm 3 succeeds probability at least p. In a round i where the algorithm succeeds we have that $OPT(G_{i+1}) < OPT(G)$. Since the algorithm terminates when $OPT(G_i) = 0$ we get the following observation.

Observation 4. *There exists a constant $p > 0$ such that the expected number of rounds of a run of Algorithm 3 is at most $\frac{1}{p}OPT(G)$.*

Since the number of rounds where Algorithm 3 fails is at most the total number of rounds it follows form Lemma 15 that the expected size of the output solution $|S|$ is at most $OPT(G) + \frac{1}{p}OPT(G)$. This proves the following lemma.

Lemma 17. *For every connected $\mathcal{F} \in \mathscr{F}$, Algorithm 3 runs in time $O(n + m)$, expected time $O((n + m)OPT(G))$ and outputs an F solution S with $E[|S|] = c \cdot OPT(G)$ for some constant c.*

While Lemma 12 only gives constant factor approximation algorithms for $\mathcal{F}$-DELETION for *connected* $\mathcal{F} \in \mathscr{F}$, we can use this approximation algorithm to make an approximation algorithm for all $\mathcal{F} \in \mathscr{F}$.

Theorem 12. *For every $\mathcal{F} \in \mathscr{F}$, $\mathcal{F}$-DELETION has a constant factor approximation running in time $O(nm)$ and expected time $O((n + m)OPT(G))$. It outputs a feasible solution S with expected size $c \cdot OPT(G)$ for a constant c.*

Proof. By Proposition 1, for every $\mathcal{F} \in \mathscr{F}$ there is a constant η such that for any $\mathcal{F}$-deletion set S of G we have $\mathbf{tw}(G \setminus S) \leq \eta$. Since TREEWIDTH η-DELETION is a $\mathcal{F}$-DELETION problem for a connected $\mathcal{F}' \in \mathscr{F}$ it follows from Lemma 12 has a constant factor approximation with the desired running time. We run this algorithm and find a set S' such that $\mathbf{tw}(G \setminus S') \leq \eta$. We have that $E[|S'|] = O(OPT(G))$, where OPT(G) refers to the size of the smallest $\mathcal{F}$-deletion set in G. Since $\mathbf{tw}(G \setminus S') \leq \eta$ we can solve $\mathcal{F}$-DELETION on $G \setminus S$ in linear time and find a set S^* of size $OPT(G \setminus S') \leq OPT(G)$. We return $S = S' \cup S^*$, S is a $\mathcal{F}$-deletion set of G with expected size $O(OPT(G))$. $\square$

Interestingly we can also use Algorithm 3 to give a fast randomized FPT algorithm for p-$\mathcal{F}$-Deletion.

Theorem 13. *For every connected $\mathcal{F} \in \mathscr{F}$, p-$\mathcal{F}$-Deletion has a randomized $O(c^k n)$ time algorithm. Given a yes instance the algorithm finds a solution and outputs it with probability $1/2$. If the algorithm outputs a solution, it is a feasible solution of size at most k.*

Proof. We modify Algorithm 3 in the following way; if G_{k+1} is not $\mathcal{F}$-free then output "no" and halt. If the size of the output solution is more than k then output "no" instead. The algorithm runs for at most $k+2$ rounds so the total running time is at most $O(nk)$. If it outputs a solution S then S is an $\mathcal{F}$ deletion of size at most k. We prove that if G has an $\mathcal{F}$ deletion of size at most k then the algorithm will output a solution with probability at least $\frac{1}{c^{k+2}}$ for a constant c. Repeating this algorithm $O((1/c)^k)$ times and outputting a solution if either iteration does then proves the theorem.

In each round, the probability that Algorithm 3 succeeds is at least p for some constant p. Thus the probability that Algorithm 3 succeeds in all its rounds before it terminates (after at most $k+2$ rounds) is at least p^{k+2}. If the algorithm succeeds in all rounds and outputs a solution then this solution is optimal and hence has size at most k if (G,k) is a yes instance. Finally, if the algorithm succeeds for $k+2$ rounds then $OPT(G_1) > OPT(G_2)\ldots OPT(G_{k+2})$ and so $OPT(G_1) \geq k+1$. Hence, if (G,k) is a "yes" instance and the Algorithm 3 succeeds in all of its rounds then it will output a solution of size at most k before terminating. This concludes the proof. □

7 Parameterized Algorithms for p-$\mathcal{F}$-Deletion

We now give a deterministic $O(c^k n \log^2 n)$ time FPT Algorithm for p-$\mathcal{F}$-Deletion for all connected $\mathcal{F} \in \mathscr{F}$.

Lemma 18. *For every connected $\mathcal{F} \in \mathscr{F}$, there exist constants ρ, r, s, $c < 1$ such that if we run the DFPR with parameters r, s on an instance (G,k) such that G has a $\mathcal{F}$ deletion S which is not a ρ-cover, then the output instance (G',k') satisfies $|V(G)| - |V(G')| \geq \frac{c|V(G)|}{\log |V(G)|}$.*

Proof. If G has a $\mathcal{F}$-deletion set S' which is not an ρ-cover, it also has a inclusion minimal $\mathcal{F}$-deletion set S which is not an ρ-cover. Such a minimal S contains no isolated vertices and hence satisfies $N[S] \leq 2\sum_{v \in S} d(v) \leq 2\rho m$.

By Proposition 1, there exists a constant b such that $\mathbf{tw}(G \setminus S) \leq b$. By Lemma 5, G has a $(4(b+1)|N[S]|, 2(b+1))$-protrusion decomposition. Set $\beta = 2(b+1)$, $r = 3(\beta+1)$ and r' to be the smallest integer such that the protrusion replacer will replace r-protrusions of size at least r'. Set $s = 2^r \cdot r'$. The protrusion decomposition of G is a $(4(b+1)|N[S]|, \beta)$-protrusion decomposition. By Theorem 8 there exist constants $0 < c < 1$ and $0 < \gamma$ such that if we run the DRFPR on G and $4(b+1)|N[S]| \leq \frac{n}{244s}$ then the output graph G' satisfies $|V(G)| - |V(G')| \geq \frac{c|V(G)|}{\log n}$. We show that there is a constant $\rho < \frac{1}{3}$ such that if S is not a ρ-cover, then $|N[S]| \leq \frac{n}{1000(b+1)s}$.

Since $\mathbf{tw}(G \setminus S) \leq b$ we have that $G \setminus S$ is $(b+1)$-degenerate. If S is not a ρ-cover then $m \leq n(b+1) + \sum_{v \in S} d(v) \leq n(b+1) + 2\rho m$. Rearranging yields that $N[S] \leq 2\rho m \leq n\frac{2\rho(b+1)}{1-2\rho} \leq n\rho 6(b+1)$. Choosing $\rho = 6000(b+1)^2 s$ yields that $|N[S]| \leq \frac{n}{1000(b+1)s}$. Hence, if S is not a ρ-cover then the output instance G' of the RFPR satisfies $|V(G)| - |V(G')| \geq \frac{c|V(G)|}{\log n}$. □

Lemma 19. *For every connected $\mathcal{F} \in \mathscr{F}$ there is an algorithm that given an instance (G,k), takes $O((n+m)\log^2 n)$ time and outputs an equivalent instance graph (G',k') such that $V(G') \leq V(G)$ and $OPT(G') \leq OPT(G)$. Furthermore there exist a constant $0 < \rho < 1$ such that every $\mathcal{F}$-deletion set S' of G' is a ρ-cover of G.*

24

Proof. By Lemma 18 there exist constants ρ, r, s, $c < 1$ such that if we run the DFPR with parameters r, s on an instance (G, k) such that G has a $\mathcal{F}$ deletion S which is not a ρ-cover, then the output instance (G', k') satisfies $|V(G)| - |V(G')| \geq \frac{c|V(G)|}{\log |V(G)|}$. We set these constanst as guaranteed by Lemma 18.

The algorithm sets $(G_1, k_1) := (G, k)$, $i = 1$ and enters a loop that proceeds as follows. The algorithm runs the DFPR on (G_i, k_i) with parameters r and s, let the output of the DFPR be (G_{i+1}, k_{i+1}). If $|V(G_i)| - |V(G_{i+1})| < \frac{c|V(G)|}{\log |V(G)|}$ the algorithm halts and outputs G_i. Otherwise, the algorithm increments i and returns to the beginning of the loop.

One iteration of the loop takes time $O((|V(G_i)| + |E(G_i)|) \log |V(G_i)|)$. Furthermore, every $\log n$ consecutive iterations of the loop reduces the number of vertices by a linear fraction. Hence the total running time us bounded by $O(n \log^2 n)$. Let (G', k') be the instance we output. By Lemma 18 we have that every $\mathcal{F}$-deletion set S' of G' is an ρ-cover of G. $\qquad\square$

We will say that an instance (G, k) is *irreducible* if running the algorithm of Lemma 19 when run on (G, k) just outputs (G, k) unchanged. Observe that if we run the algorithm of Lemma 19 when run on an instance (G, k), the instance (G', k') output by the algorithm is irreducible. A direct consequence of Lemma 19 is that in an irreducible instance (G, k) every $\mathcal{F}$-deletion set S in G is a ρ-cover.

We now give a deterministic algorithm for p-$\mathcal{F}$-DELETION. for connected $\mathcal{F} \in \mathscr{F}$. The intuition behind this algorithm is that vertices of high degree seem more useful for a solution than the vertices of low degree. Towards this we introduce the notion of buckets. We partition the vertex set of G into sets that we refer to as *buckets*, in the following fashion. For every $j \geq 1$ define

$$B_j = \left\{ v \in V(G) \ \Big| \ \frac{n}{2^j} < d(v) \leq \frac{n}{2^{j-1}} \right\}.$$

We set constants $\eta > 0$ and $d > 0$ such that $\frac{4d + 3\eta}{2} < \rho$. For the presentation of the algorithm we fix a $\mathcal{F}$-deletion set set X of size at most k. Next we define a notion of big and good for buckets.

Definition 4. *A bucket B_i is said to be* big *if $|B_i| > i\eta$ and it is said to be* good *if $|B_i \cap X| \geq d|B_i|$.*

The next lemma says that if (G, k) is a irreducible yes instance to p-$\mathcal{F}$-DELETION then it has a bucket that is both big and good simulatenouly.

Lemma 20. *For any connected $\mathcal{F} \in \mathscr{F}$, let (G, k) be a irreducible yes instance to p-$\mathcal{F}$-DELETION. Then G has a bucket that is both big and good.*

Proof. Since (G, k) a irreducible yes instance to p-$\mathcal{F}$-DELETION every optimal $\mathcal{F}$-hitting set X is a ρ-cover for G, that is, $\sum_{v \in V(G)} d(v) \leq \rho \sum_{v \in X} d(v)$. For a contradiction, assume that G does not have a bucket that is both big and good.

$$\sum_{v \in X} d(v) = \sum_{i=1}^{\log n} \sum_{v \in B_i \cap X} d(v)$$

$$= \sum_{\{i | B_i \text{ is not good}\}} \sum_{v \in B_i \cap X} d(v) + \sum_{\{i | B_i \text{ is not big}\}} \sum_{v \in B_i \cap X} d(v)$$

$$\leq d \cdot 4m + \sum_{\{i | B_i \text{ is not big}\}} i\eta \cdot \left(\frac{n}{2^i} \right)$$

$$\leq d \cdot 4m + 3\eta n = 2m \frac{4d + 3\eta}{2} < 2m\rho$$

Which contradicts that X is a ρ-cover. $\qquad\square$

Algorithm-FPT-Det(G,k)

Step 1: Check whether G is $\mathcal{F}$-free, if yes then **return**(true). Else if $k \leq 0$ and G is not $\mathcal{F}$-free return that G does not have a k-sized $\mathcal{F}$-hitting set.

Step 2: Apply Lemma 19 on (G, k) and obtain an equivalent irreducible instance (G^*, k^*).

Step 3: Let B_j, $j \in \{a, b, \ldots, \ell\}$, be the good buckets for G^*. For every good bucket B_j, and for every subset $S \subseteq B_j$ of size at least $d|B_j|$ check whether **Algorithm-FPT-Det**($G^* \setminus \{S\}$,$k - |S|$) returns true. If any of these calls return true then **return**(true) else **return**(false).

Figure 4: A $2^{O(k)} n \log^2 n$ deterministic FPT algorithm for p-$\mathcal{F}$-DELETION.

Theorem 14. *Let $\mathcal{F} \in \mathscr{F}$ be a connected obstruction set. There exists a determintistic algorithm for p-$\mathcal{F}$-DELETION running in time $O(c_h^k n \log^2 n)$ on a n vertex graph. The constant c_h only depends on $\mathcal{F}$.*

Proof. The deterministic algorithm for p-$\mathcal{F}$-DELETION is described in details in Figure 4. Given a graph G, the algorithm essentially applies Lemma 19 to obtain G^* and then recursively tries to compute the solution to the problem by branching on all large subsets of all the good buckets. The correctness follows directly from Lemma 20. Next we analyze the running time of the algorithm. Suppose for the sake of analysis that all buckets are big, and let a_i be the size of bucket i. Then we have that

$$T(k) \leq \sum_{i=1}^{\log n} \binom{a_i}{k} T(k - da_i)$$

$$T(k) \leq \sum_{i=1}^{\log n} 2^{a_i} T(k - da_i)$$

Assuming $T(k) = x^k$, substitute recursively to get:

$$T(k) \leq \sum_{i=1}^{\log n} 2^{a_i} x^{(k - da_i)}$$

$$T(k) \leq x^k \sum_{i=1}^{\log n} \left(\frac{2}{x^d} \right)^{a_i}$$

If $\frac{2}{x^d} < 1$ then each term of the sum is maximized when the exponent is as small as possible. We will choose x (based on d) such that $\frac{2}{x^d} < 1$ holds. Since $a_i \geq \eta i$ for any big bucket we have that

$$T(k) \leq x^k \sum_{i=1}^{\log n} \left(\frac{2}{x^d} \right)^{\eta i}$$

The sum above is a geometric series and converges to a value that is at most 1 for $x = c$, for a suitably small choice of c depending only on d and η, which depended only on $\mathcal{F}$. This bounds the running time by c^k. Further, if not all buckets are big the sum above should only be done over the big buckets, yielding the same result. $\qquad \square$

26

8 Kernelization for p-$\mathcal{F}$-DELETION

In this section we give polynomial kernel for p-$\mathcal{F}$-DELETION problem. We start with some combinatorial results on folios and Well Quasi Ordering (WQO) of t-Boundaried Graphs and then move to the kernelization steps of our algorithm.

8.1 Minors, Folios and Well Quasi Ordering of t-Boundaried Graphs

Minors and folios of t-boundaried graphs. We can define a minor relation for t-boundaried graphs just as for normal graphs. We say that a t-boundaried graph H is a minor of a t-boundaried graph G if (a t-boundaried graph isomorphic to) H can be obtained from G by deleting vertices or edges or contracting edges, but never contracting edges with both endpoints being boundary vertices. Here, when we contract an edge between a boundary vertex u and a non-boundary vertex v the resulting vertex is a boundary vertex with label $\ell_G(u)$. If H is a minor of G we say that $H \leq_m G$.

For a t-boundaried graph G the *folio* of G is the set $\mathbf{folio}(G) = \{H : H \leq_m G\}$. The δ-*folio* of G is the set δ-$\mathbf{folio}(G) = \{H : |V(H)| \leq \delta \text{ and } H \leq_m G\}$. The following lemma states that the δ-folio of a sum can be determined by examining the δ-folios of its terms.

Lemma 21. *Let G_1 and G_2 be t-boundaried graphs and $G = G_1 \oplus_\delta G_2$. Then $H \leq_m G$ if and only if there exist $H_1 \leq_m G_1$ and $H_2 \leq_m G_2$ such that $H_1 \oplus_\delta H_2 = H$.*

Proof. For the reverse direction, observe that $H \leq_m H_1 \oplus H_2 \leq_m G_1 \oplus G_2 = G$ and so $H \leq_m G$. It remains to prove the forward direction, so suppose H is a minor of G. Consider a model $(P_1, P_2, \ldots P_{|V(H)|})$ of H in G. Notice that each P_i intersects $\delta(G)$ at most once. Furthermore, every P_i that does not intersect $\delta(G)$ lies either entirely in G_1 or entirely in G_2. Now, consider the collection $\mathcal{M}_1$ of sets $P_i \cap V(G_1)$ such that $P_i \cap V(G_1) \neq \emptyset$ and the collection $\mathcal{M}_2$ of sets $P_i \cap V(G_2)$ such that $P_i \cap V(G_2) \neq \emptyset$. Each set in $\mathcal{M}_1$ and $\mathcal{M}_2$ induces a connected subgraph in G_1 and G_2 respectively. All sets in $\mathcal{M}_1$ are pairwise disjoint, and the same holds for $\mathcal{M}_2$. Thus $\mathcal{M}_1$ and $\mathcal{M}_2$ are models of graphs $H_1 \leq_m G_1$ and $H_2 \leq_m G_2$ respecitvely. Finally $H_1 \oplus_\delta H_2 = H$ because for every i, j there is an edge between $G[P_i]$ and $G[P_j]$ if and only if there is an edge between $G[P_i \cap V(G_1)]$ and $G[P_j \cap V(G_1)]$ or between $G[P_i \cap V(G_2)]$ and $G[P_j \cap V(G_2)]$. $\square$

Lemma 22. *Let G' be a t-boudaried graph, $S \subseteq \delta(G')$ and $G = \mathbf{forget}(G', S)$. Let H be a t-boundaried graph. Then $H \leq_m G$ if and only if there is a t-boundaried $H' \leq_M G'$ such that $H \leq_m \mathbf{forget}(H', S)$ and $|V(H')| \leq |V(H)| + |S|$.*

Proof. For the reverse direction observe that since $H' \leq_M G'$ and $G = \mathbf{forget}(G', S)$ we have $\mathbf{forget}(H', S) \leq_m \mathbf{forget}(G', S) = G$. In the forward direction, consider a model $(P_1, P_2, \ldots P_{|V(H)|})$ of H in G. Start from G' and delete all vertices that are not in a set P_i. Then, as long as there is an edge with both endpoints in the same set P_i and at least one endpoint is not boundary vertex of G' contract this edge. Modify the set P_i to contain the vertex resulting from the contraction, rather than the two endpoints. The resulting t-boundaried graph H' is a minor of G'. Let P_i' denote the vertices corresponding to P_i in H'. Note that each P_i' is a single non-boundary vertex and some subset of the boundary vertices. Since $H \leq_m G$, notice that no P_i' contains two boundary vertices from $\delta(G)$. Note that $\delta(G) = \delta(G') \setminus S$, so when we consider the set P_i' in $\mathbf{forget}(H', S)$, it contains at most one boundary vertex. Thus, at this point, $(P_1, P_2, \ldots P_{|V(H)|})$ is a model of H in H' so $H \leq_m H'$.

We now show that if $|V(H')| > |V(H)| + |S|$ then some P_i contains at least two vertices such that at least one of them is not a border vertex. Suppose not, then every P_i contains at most one vertex that is not a border vertex of G'. No P_i contains vertices from $\delta(G') \setminus (\delta(H) \cup S)$ since $(P_1, \ldots, P_{|V(H)|})$ is a minor model of H in G and we can't contract edges between border
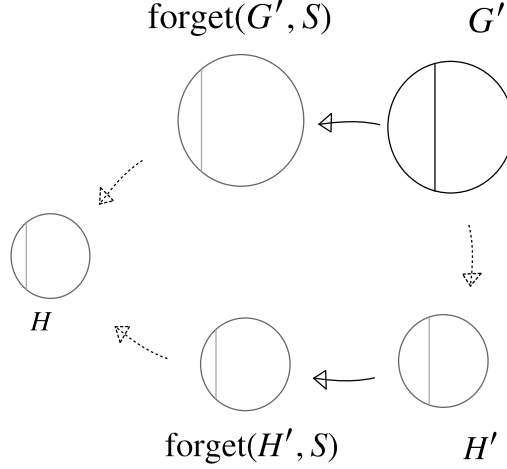
Figure 5: The above is a schematic for Lemma 22. The dotted lines represent minor operations, and the solid arrows represent discardig labels. The lemma shows that a minor H obtained after discarding some set S of labels from G' can be realized by a minor H' of G', whose size is no more than $|H| + |S|$. The graph H can be obtained as a minor of $\mathbf{forget}(H', S)$.

vertices of G. Then every P_i that contains at least one vertex from $\delta(G')$ is a subset of $\delta(H) \cup S$. At most $|V(H)| - |\delta(H)|$ V_i's contain no vertex from $\delta(H) \subseteq \delta(G')$ and each of these P_i's has size 1. So the total size of the P_i's is at most $|\delta(H)| + |S| + |V(H)| - |\delta(H)| = |V(H)| + |S|$. But every vertex in H' is in some P_i, contradicting $|V(H')| > |V(H)| + |S|$.

Suppose now that $|V(H')| > |V(H)| + |S|$ and consider a P_i that contains at least two vertices such that at least one of them is not a border vertex. Since $H'[P_i]$ is connected there is at least one edge in $H'[P_i]$ that has at least one endpoint which is not on the border of G'. This edge should have been contracted, contradicting the construction of H' this proves $|V(H')| \le |V(H)| + |S|$, concluding the proof. $\qquad \square$

The following lemma follows directly from Lemmata 21 and 22. It states that if a graph G has a separator of size t, then we can determine whether G contains H as a minor by examining the $(t + |V(H)|)$-folios of the two sides of the separator.

Lemma 23. *Let G_1 and G_2 be t-boundaried graphs and $G = G_1 \oplus G_2$. A graph H is a minor of G if and only if there exist $H_1 \le_m G_1$ and $H_2 \le_m G_2$ such that $|V(H_1)| \le |V(H)| + t$, $|V(H_2)| \le |V(H)| + t$ and $H \le_m H_1 \oplus H_2$.*

Proof. The reverse direction follows since $H \le_m H_1 \oplus H_2 \le_m G_1 \oplus G_2 = G$. For the forward direction let $G^* = G_1 \oplus_\delta G_2$. We have that $G = \mathbf{forget}(G^*, \delta(G^*))$ and so by Lemma 22 there exists an $H^* \le_m G^*$ such that $|V(H^*)| \le |V(H)| + t$ and $H \le_m \mathbf{forget}(H^*, \delta(H^*))$. By Lemma 21 there exist $H_1 \le_m G_1$ and $H_2 \le_m G_2$ such that $H_1 \oplus_\delta H_2 = H^*$. But then $|V(H_1)| \le |V(H^*)| \le |V(H)| + t$ and identically $|V(H_2)| \le |V(H^*)| \le |V(H)| + t$. Finally, since $H_1 \oplus_\delta H_2 = H^*$ we have $H_1 \oplus H_2 = \mathbf{forget}(H^*, \delta(H^*))$ and since $H \le_m \mathbf{forget}(H^*, \delta(H^*))$ the proof follows.

$\qquad \square$

Well-quasi-ordering t-boundaried graphs. A partial order of a (possibly infinite) set S is a relation $\le$ on S that is reflexive, transitive and antisymmetric. Two elements a and b of S
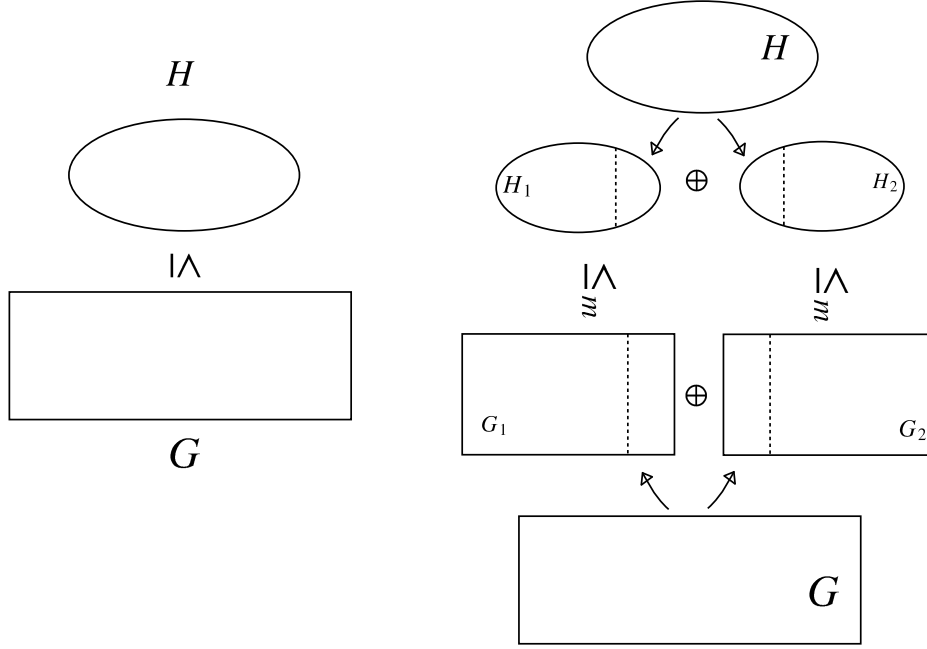
Figure 6: A schematic representation of Lemma 23.

are *comparable* with respect to $\leq$ if $a \leq b$ or $b \leq a$ and *incomparable* otherwise. A subset S' of S is an *antichain* (of $\leq$) if no two distinct elements of S' are comparable. A partial order $\leq$ is a *well-quasi-order* of S if every antichain $S' \subset S$ is finite. The famous graph minor theorem states that graphs are well-quasi-ordered by the minor relation [74]. We will need a weaker version of this theorem translated to t-boundaried graphs.

Lemma 24. *For every t and w the set of t-boundaried graphs with treewidth at most w is well-quasi-ordered.*

Proof. We prove the lemma by giving an injection f from the set of t-boundaried graphs of treewidth at most w to graphs such that for any two uncomparable t-boundaried graphs H and G of treewidth at most w, $f(H)$ and $f(G)$ are uncomparable as well. Hence an infinite antichain of t-boundaried graphs of treewidth at most w would yield an infinite antichain of graphs, contradicting the graph minor theorem.

For a t-boundaried graph G, let $L(G)$ denote the set of labels used by the boundary vertices of G, that is, $L(G) := \{l(v) \mid v \in \delta(G)\}$. We begin by noticing that it suffices to show the injection f from the set of t-boundaried graphs G for which $L(G) = X$ for some arbitrary but fixed set $X \subseteq [t]$. This is because there are only finitely many subsets of $[t]$, and therefore any infinite antichain of t-boundaried graphs of treewidth at most w contains an infinite antichain of t-boundaried graphs whose boundary sets use the same set of labels.

Let G be a t-boundaried graph. The graph $f(G)$ is obtained from G by adding some cliques to the vertices of $\delta(G)$. In particular, let $v \in \delta(G)$ and let $l(v) = l_v$. For every $v \in \delta(G)$, in $f(G)$ we introduce a clique C_v on $w + l_v + 1$ vertices and make v adjacent to all vertices in C_v. We use G^* to denote $f(G)$.

Recall that X is an arbitrary but fixed subset of $[t]$. Let H and G be two t-boundaried graphs of treewidth at most w such that X is the set of labels used by the boundary vertices of H and G. We claim that if $H^* \leq_m G^*$, then $H \leq_m G$.
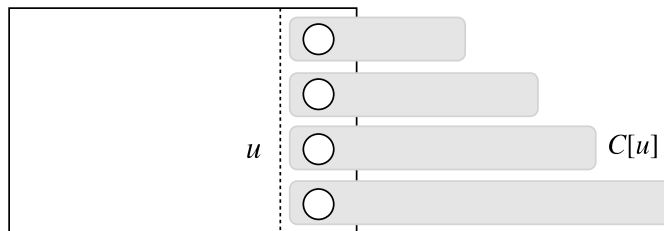
Figure 7: A representation of the function f. The grey rectangles represent cliques.

Let $H^* \leq_m G^*$. Let $(P_1, \ldots, P_{V(H^*)})$ be a model of H^* in G^*. We abuse notation and use P_v to denote the component of G^* that the vertex $v \in H^*$ is mapped to by the model under consideration. Now, let $u \in \delta(H)$ with label l_u. Let u_g be the vertex in $\delta(G)$ that has label l_u. Since H is a subgraph of H^*, $u \in H^*$, and similarly, $u_g \in G^*$. Let $C[u]$ and $C_g[u]$ denote the vertices of the clique attached to the vertices u and u_g in H^* and G^* (respectively). Further, let B_G denote the set $\delta(G)$ in G^*, let V_G denote the vertices of $G \setminus B$ in G^*, and let C_G denote the set of all vertices that participate in the cliques introduced in G^* (by construction).

We first show that for every vertex z in $C[u]$, P_z contains some vertex of $C_g[u]$. We prove this by reverse induction on the set of labels X used in G. Let r denote $|X|$, and assume, without loss of generality, that the labels used are from $[r]$. The base case is that $l_u = r$. Consider some $z \in C[u]$, and suppose, for the sake of contradiction, that $P_z \cap C_g[u] = \emptyset$. Note that P_z must contain a vertex whose neighborhood in G^* contains a clique on at least $r + w + 1$ vertices. The only such vertices in G^* are $C_g[u] \cup \{u_g\}$. Given our assumption, this implies that $u_g \in P_z$. But now consider P_u, which must also contain a vertex whose neighborhood in G^* contains a clique on at least $r + w + 1$ vertices, and whose degree is at least $r + w + 2$ (since there are no isolated boundary vertices in H). Since $u_g \in P_z$, the only remaining candidates for P_u are vertices in $C_g[u]$, but all such vertices have degree exactly $r + w + 1$, and therefore, there is no valid choice for P_u. This is a contradiction.

The induction hypothesis states that if $l_u = s + 1$, for all $z \in C[u]$, P_z contains some vertex of $C_g[u]$. We now prove that when $l_u = s$, for all $z \in C[u]$, P_z contains some vertex of $C_g[u]$. It is clear that P_z must contain a vertex whose neighborhood in G^* contains a clique on at least $s + w + 1$ vertices. The vertices which satisfy this property include $\{C_g[v] \cup \{v_g\}\}$ for all v for which $l_v \geq s$. Notice that all vertices in $\cup_{\{v \mid l_v > s\}} C_g[v]$ already belong to P_z for some $z \in C[v]$. Since the P_i's are disjoint, the only remaining candidate vertices for P_z are $\{v_g \mid l_v \geq s\} \cup C_g[u]$. For reasons similar to the ones described in the proof of the base case, P_z cannot contain the vertex v_g for any $v \in \delta(G)$. Therefore, P_z must contain some vertex from $C_g[u]$, as desired. This proves the claim.

We now argue that for any $u \in \delta(H)$, we have that the corresponding vertex in H^* is mapped to a set that contains the boundary vertex with the same label as u in G^*. Formally, we claim that $u_g \in P_u$. It is clear that P_u must contain a vertex whose neighborhood in G^* has a clique on at least $l_u + w + 1$ vertices. Since the treewidth of G is at most w, we have that $G^*[V_G \cup B_G]$ does not contain any cliques of size $w + 2$. Notice also that all the vertices in C_G are shown to belong distinct sets P_z. This follows from the claim above and the fact that the set of labels of H and G are identical — which implies that the sizes of the corresponding cliques introduced in H^* and G^* are the same. Thus, it is clear that only the vertices from B_G have sufficiently large cliques in their neighborhood. It is easy to see that a reverse induction argument similar to the one used in the claim above establishes that $u_g \in P_u$.

Thus, we have that in any minor model of H^* in G^*, the clique vertices introduced in H^* are mapped to the corresponding clique vertices in G^*, and the vertices of $\delta(H)$ in H^* are mapped

30

to the corresponding vertices in $\delta(G)$ in G^*. Now we simply consider the minor model of H^* in G^* after disregarding the P_z's that contain vertices from C_G. This is evidently a minor model of H in G, and therefore, we have that $H \leq_m G$, as desired. This concludes the proof.

$\square$

8.2 Decomposition into near-Protrusions

In this section we prove that any yes-instance G to $\mathcal{F}$-DELETION has a set D of $O(k^3)$ vertices such that every connected component C of $G \setminus D$ is a *near-protrusion*. Recall that a r-protrusion C in a graph G is a vertex set such that $|\delta(C)| \leq r$ and $\mathbf{tw}(G[C]) \leq r$. The components of $G \setminus D$ will not necessarily be protrusions, however we will prove that there is a constant r such that if (G, k) is a yes instance, then for any $\mathcal{F}$-deletion set S of size at most k, $C \setminus S$ is a r-protrusion of $G \setminus S$. The decomposition algorithm starts by finding a constant factor approximation of the smallest $\mathcal{F}$-deletion set of G with high probability. This is achieved by running the algorithm given by Theorem 1 n times, so as to boost the probability of success to $(1 - 1/2^n)$.

By Proposition 1 there exists a constant η such that every $\mathcal{F}$-minor-free graph G has treewidth at most η. Having computed a c-approximate $\mathcal{F}$-deletion set X we check whether $|X| \leq c(k + 1)$, and if not[1], then the algorithm concludes that G has no $\mathcal{F}$-deletion set of size k. We now compute a set Y disjoint from X as follows. Initially $Y = \emptyset$. For a pair u and v of distinct vertices in X, define $G_{u,v}$ to be $(G \setminus (X \cup Y \setminus u, v)) \setminus uv$. In other words $G_{u,v}$ is obtained from G by removing all vertices in X and Y except for u and v, and removing the edge uv if it exists. We check using maximum flow whether there exists a pair of distinct vertices $u, v \in X$ such that the number of vertex disjoint paths from u to v in $G_{u,v}$ is less than $\eta + k + 3$, but at least one. If such a pair exists then by Menger's theorem there is a set $S \subseteq V(G) \setminus (X \cup Y)$ of size at most $\eta + k + 2$ such that there are no paths from u to v in $G_{u,v}$. We add the set S to Y. Observe that $|Y| \leq |X|^2 \cdot (\eta + k + 2)$ since we add a set of $\eta + k + 2$ vertices at most once for each pair of vertices in X.
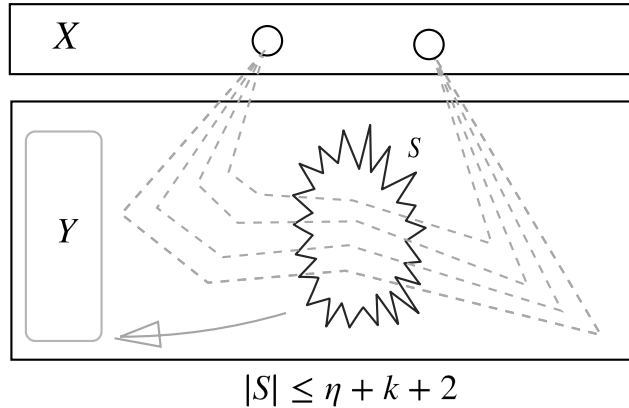

$$|S| \leq \eta + k + 2$$

Figure 8: This figure demonstrates the construction of Y based on small separating sets for pairs of vertices in X.

Furthermore, for every connected component C of $G \setminus (X \cup Y)$ it holds that for any two vertices u and v in $N(C) \cap X$ there are at least $\eta + k + 3$ vertex disjoint paths from u to v.

---

[1] The kernelization algorithm only requires to check if $|X| \leq ck$, but we choose to use the bound $c(k + 1)$ to allow for a more general consequence.

Indeed, if this was not the case then a vertex set separating u and v in $G_{u,v}$ would have been added to Y, contradicting the construction of Y.
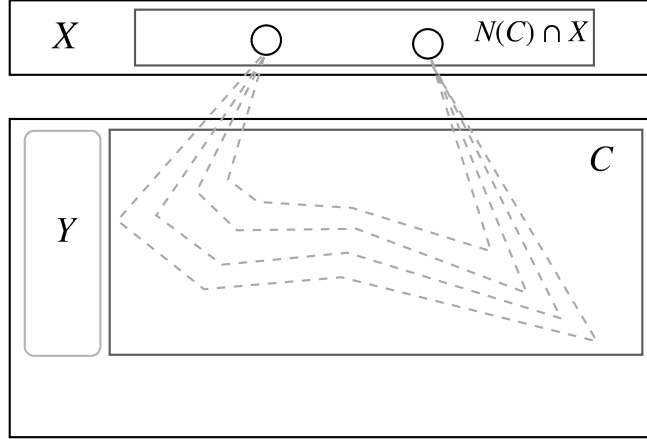

Figure 9: After the construction of Y, for every component C outside $(X \cup Y)$, every pair of vertices in X in $N(C)$ has a large flow through $G \setminus (X \cup Y)$

We will now construct a set $Z \supseteq Y$ from Y and X as follows (see Figure 10). The graph $G \setminus X$ is $\mathcal{F}$-minor-free and so the treewidth of $G \setminus X$ is at most η. We compute a nice tree-decomposition $(T, \mathcal{B} = \{B_1, B_2, \ldots, B_{|V(T)|}\})$ of $G \setminus X$ of width at most η in linear time using Bodlaender's algorithm [11]. For each vertex $v \in Y$ we put one node t of T such that $v \in B_t$ into the set M'. Let M be the LCA-closure of M'. We refer to M as the set of *marked* nodes. By Lemma 1 $|M| \leq 2|M'| \leq 2|Y|$. Set $Z = \bigcup_{t \in M} B_t$. Thus $Y \subseteq Z$, $|Z| \leq 2|Y|(\eta + 1)$ and Z and X are disjoint.

We now prove that for any connected component C of $G \setminus (X \cup Z)$ only sees a constant number of vertices in Z. For any connected component C of $G \setminus (X \cup Z)$ there is a component P of $T \setminus M$ such that $C \subseteq (\bigcup_{t \in P} B_t) \setminus Z$. By Lemma 1 we have that in T, $|N(P)| \leq 2$. But then $N(C) \cap Z \subseteq \bigcup_{t \in N(P)} B_t$, and hence $|N(C) \cap Z| \leq 2(\eta + 1)$. We can summarize the above discussion in the following result.

Lemma 25. *There is a randomized polynomial time algorithm that given an instance (G, k) of $\mathcal{F}$-Deletion, either returns that (G, k) is a no instance, or computes a vertex set X and a set Z disjoint from X such that*

- $|X| = O(k)$ and $|Z| = O(k^3)$.

- X is an $\mathcal{F}$-Deletion of G.

- For every connected component C of $G \setminus (X \cup Z)$, $|N(C) \cap Z| \leq 2(\eta + 1)$.

- For every connected component C of $G \setminus (X \cup Z)$, and $u, v \in N(C) \cap X$ there are at least $k + \eta + 3$ vertex disjoint paths from u to v in G.

If (G, k) is a yes instance, the algorithm outputs (X, Z) with probability $(1 - 1/2^n)$, and a no instance is rejected with probability 1.

In the remainder of the paper the sets X and Z will always refer to the sets X and Z as guaranteed by Lemma 25. We now observe that the connected components of $G \setminus (X \cup Z)$ are near-protrusions.
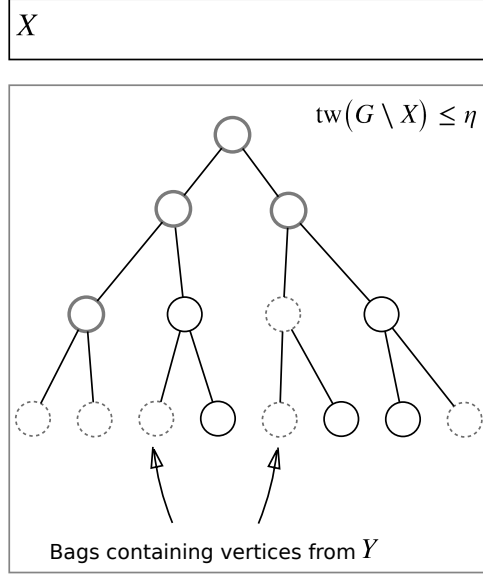
Figure 10: This is a schematic for the construction of Z based on Y and a tree decomposition of $G \setminus X$.

Lemma 26. *For any $\mathcal{F}$-DELETION S of G of size at most $k+1$ and any connected component C of $G \setminus (X \cup Z)$, $|(N(C) \cap X) \setminus S| \leq \eta + 1$.*

Proof. Let $P = (N(C) \cap X) \setminus S$. Let $(T, \mathcal{B})$ be a tree-decomposition of $G \setminus S$ of width at most η. We prove that there is a bag of this tree decomposition that contains P, thereby proving $|P| \leq \eta + 1$. Since tree-decompositions enjoy the Helly property, it is sufficient to show that for every pair of vertices u and v in P there is a bag of the tree-decomposition that contains both u and v [59]. The proof proceeds by contradiction.

Assume that no bag of T contains both u and v. Observe that for every pair of vertices u and v in P, there are at least $k + \eta + 3$ disjoint paths from u to v in G and hence there are at least $\eta + 2$ disjoint paths from u to v in $G \setminus S$. But since no bag of T contains both u and v then $G \setminus S$ has a u-v separator of size at most η, contradicting the existence of $\eta + 2$ disjoint paths from u to v. $\qquad\square$

8.3 Reducing the size of near-Protrusions

We now consider the sets X and Z as guaranteed by Lemma 25, and a single component C of $G \setminus (X \cup Y)$. The aim of this section is to prove that if C is "too large", then it is "not doing its job efficiently". In particular we prove that there exists a constant α depending only on $\mathcal{F}$ such that if $|C| \geq \alpha \cdot k^\alpha$ then there exists an edge e with at least one endpoint in C, an edge e' with both endpoints in C, or a vertex $v \in C$ such that deleting e, contracting e' or deleting v does not change whether G has an $\mathcal{F}$-deletion set of size at most k. Let G' be the graph obtained from G by doing this minor operation. If G does have an $\mathcal{F}$-deletion set of size at most k then G' does as well, since minor operations can not increase the size of the minimum $\mathcal{F}$-deletion set . Thus it is sufficient to prove that if G does not have an $\mathcal{F}$-deletion set of size at most k, then neither does G'. We prove this by showing that for any set S on at most k vertices, if $G \setminus S$ contains a copy of a graph H in $\mathcal{F}$ as a minor, then one of the following two things must
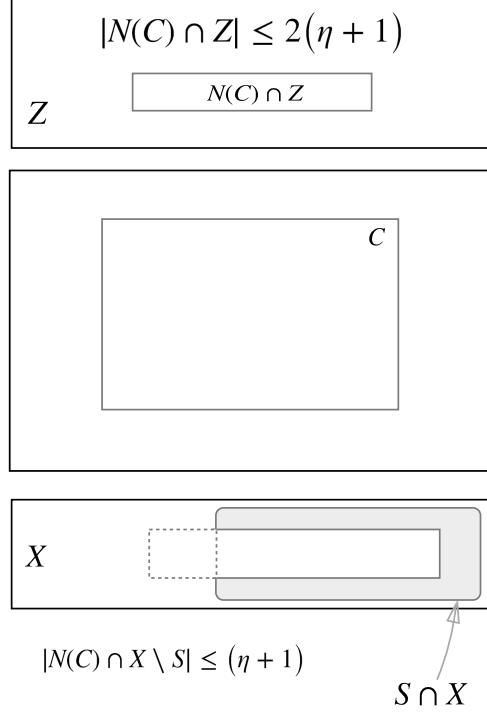
Figure 11: This figure shows, for a connected component C in $G \setminus (Z \cup X)$, the bounds on the set $N(C)$ in $G \setminus S$. The neighborhood in Z is bounded based on the construction of Z. The fact that the neighborhood in $X \setminus S$ is bounded relies on the design of C, which ensured large flows between pairs of vertices in X via $G \setminus (Z \cup X)$ (see Lemma 25).

happen in G'. Either the treewidth of $G' \setminus S$ is more than η, or the model of H in $G \setminus S$ can be modified such that the minor operation used to obtain G' from G does not destroy it. In both cases this yields a proof that S is not a $\mathcal{F}$-deletion set in G'. The reduction rule we give in this section is designed so that the plan for the proof of correctness sketched above will go through.

We will focus our attention on a particular component C of $G \setminus (X \cup Y)$. This component has treewidth at most η, and so we can compute in linear time a nice tree decomposition $(T, \mathcal{B})$ of C of width at most η. For every vertex $v \in C$ let T_v be the subtree of T corresponding to bags that contain v. We will say that a set P of vertices in C is *interesting* if there exist a subtree T' of T such that $|N(T')| \leq 2$, and $P = \{v \in C \ : \ T_v \subseteq T'\}$. There are at most $O(n^2)$ interesting sets, and all interesting sets can be listed in polynomial time. Every interesting set P satisfies that $|N(P) \setminus X| \leq 4(\eta + 1)$.

For a vertex set P in $V(G) \setminus (X \cup Y)$, the *border collection* of P is the collection $\mathcal{B}_P$ of all sets B such that $B \setminus X \subseteq N(P) \setminus X$ and $|B \cap X| \leq \eta + 1$. (See Figure 12.) The *signature* of a vertex set P of $G \setminus (X \cup Y)$ is a function σ_P. The domain of σ_P is the set of pairs $(B, \mathcal{Q})$ where $B \in \mathcal{B}_P$ while $\mathcal{Q}$ is a collection of t-boundaried graphs on at most $|B| + h$ vertices, and with B as their boundary. Here h is the maximum number of vertices of a graph in $\mathcal{F}$.

For such a pair $(B, \mathcal{Q})$, $\sigma_P(B, \mathcal{Q})$ outputs the size of a minimum cardinality set $S \subset P \cup B$ such that $\mathbf{folio}(G_C^B \setminus S) \cap \mathcal{Q} = \emptyset$. (See Figure 13.) Recall that G_P^B is the t-boundaried graph $G[P \cup B]$ with boundary B. The signature of a vertex set P is taken with respect to the graph G - we will sometimes consider the signature of P with respect to G', a graph obtained from G by a single minor operation. For an integer r the r-truncated signature of P is a function σ_P^r
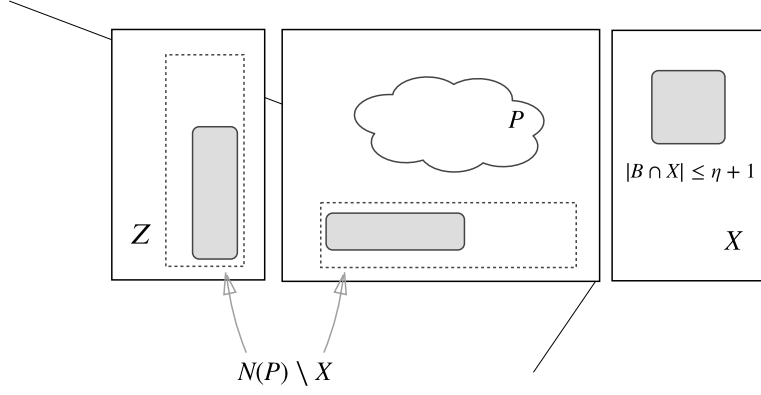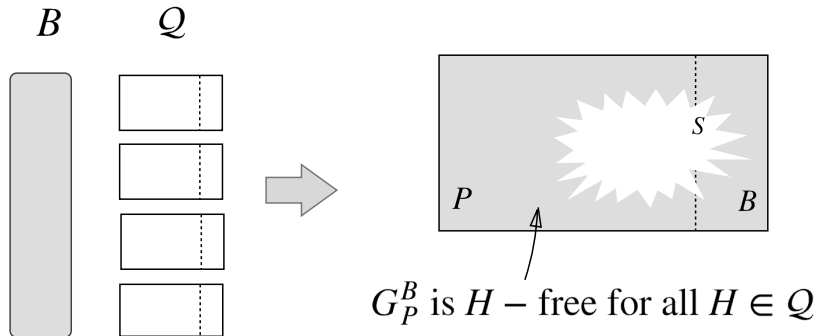
Figure 12: A representation of a set in the border collection of P. The gray areas show a subset B that would be an element of $\mathcal{B}_P$. It satisfies that outside X, B is contained in $N(P)$, and the intersection of B with X is at most $\eta + 1$.

with the same domain as σ_P. The function $\sigma_P^r(B, \mathcal{Q})$ evaluates to $\sigma_P(B, \mathcal{Q})$ if $\sigma_P(B, \mathcal{Q}) \leq r$ and $\sigma_P^r(B, \mathcal{Q}) = \infty$ otherwise.

$$\sigma_P(B, \mathcal{Q}) = \min\{|S|\}$$

Figure 13: The signature function. For a boundary set B and a family $\mathcal{Q}$, the signature function σ_P considers the graph G_P^B, that is, the graph with boundary B and P as the internal set. It returns the size of the smallest subset S of $P \cup B$ whose removal makes the graph H-minor free for all $H \in \mathcal{Q}$.

Lemma 27. *The signature of an interesting set P can be computed in time $O(n^{O(\eta)})$.*

Proof. Observe that the border collection $\mathcal{B}_P$ has size $O(k^{O(\eta)}) = O(n^{O(\eta)})$, since $|N(P) \setminus X| \leq 4(\eta + 1)$. For each choice of $B \in \mathcal{B}_P$, the size of $\mathcal{Q}$ is given by the number of possible collections of t-boundaried graphs with border B and at most $|B| + h$ vertices, which is at most $2^{2^{O(|B|+h)^2}}$, a function of $\mathcal{F}$. Thus we need to show that we can in polynomial time evaluate $\sigma(B, \mathcal{Q})$ for any choice of B and $\mathcal{Q}$. This amounts to finding a minimum cardinality set $S \subseteq P \cup B$ such

that **folio**$(G_P^B \setminus S) \cap \mathcal{Q} = \emptyset$. This problem can be expressed by the following MSO-optimization problem:

$$\text{Minimize } S \subseteq V(G)[\bigwedge_{H \in \mathcal{Q}} \neg\phi(G_P^B \setminus S, H, \delta(G_P^B \setminus S), \delta(H))] \tag{2}$$

Recall that ϕ is as defined in equation (1) (Section 2.5), has length depending only on h and $|B|$. Clearly, the length of this formula depends only on $\mathcal{F}$. Since G_C^B has treewidth at most $5(\eta + 1)$ and MSO-optimization problems can be solved in linear time on graphs of bounded treewidth [20], the proof follows. $\qquad \square$

Lemma 28. *Let P be an interesting set and let G' and P' be obtained from G and P respectively by deleting a vertex of P, deleting an edge with at least one endponint in P, or contracting an edge with both endpoints in P. Suppose the k-truncated signatures σ_P^k of P and $\sigma_{P'}^k$ of P' are identical, that is $\sigma_P^k(B, \mathcal{Q}) = \sigma_{P'}^k(B, \mathcal{Q})$ for every $(B, \mathcal{Q})$ with $B \in \mathcal{B}_P$ and $\mathcal{Q}$ being a family of t-boundaried graphs of size at most $|B| + h$. Then G has a $\mathcal{F}$-deletion set of size k if and only if G' does.*

Proof. Since G' is a minor of G it follows that when G has a $\mathcal{F}$-deletion set of size k then G' does as well. Suppose now that G' has a $\mathcal{F}$-deletion set S' of size at most k. G' was obtained from G by deleting a vertex $v \in P$ or deleting or contracting some edge e with an endpoint $v \in P$. Now $S^* = S' \cup \{v\}$ is a $\mathcal{F}$-deletion set of G of size at most $k + 1$, and so by Lemma 26 $|N(C) \cap (X \setminus S^*)| \leq \eta + 1$. Recall that C is the component of $G \setminus (X \cup Y)$ that contains P. Hence $|N(P) \cap (X \setminus S^*)| \leq \eta + 1$ as well. Since $S^* \cap X = S' \cap X$ it follows that $|N(P') \cap (X \setminus S')| \leq \eta + 1$.
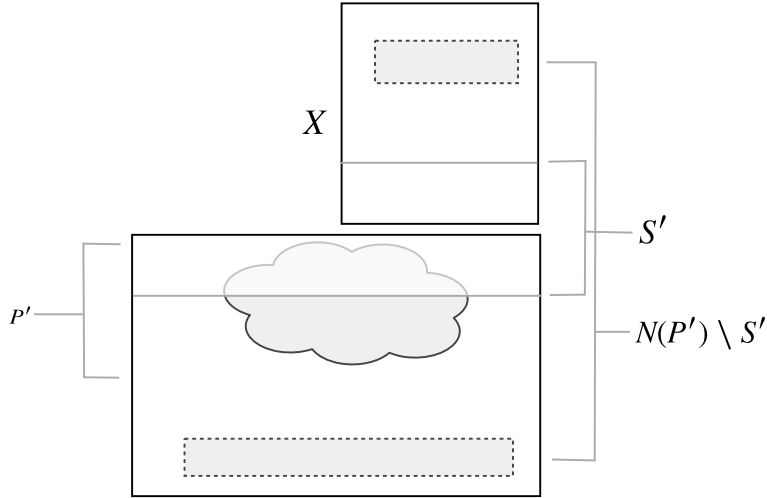

Figure 14: A schematic representation of the various subsets involved in the proof of Lemma 28.

Set $B = N_{G' \setminus S'}(P' \setminus S')$ and observe that $B \in \mathcal{B}_P$. Now define $G_{P' \setminus S'}^B$ to be the t-boundaried graph $G'[(P' \setminus S') \cup B]$ with boundary B. Set $\mathcal{Q}$ to be the set of all t-boundaried graphs of size at most $(5(\eta + 1) + h)$ that are *not* minors of $G_{P' \setminus S'}^B$. Observe that $\sigma_{P'}'(B, \mathcal{Q}) \leq |S' \cap P'|$. Thus there exists a set S_P of size at most $|S' \cap P'|$ such that $G_P^B \setminus S_P$ excludes every graph $\mathcal{Q}$ as a minor. Set $S = (S' \setminus P') \cup S_P$. Observe that since $|S_P| \leq |S' \cap P'|$ it follows that $|S| \leq |S'| \leq k$. We argue that S is a $\mathcal{F}$-deletion set of G.

Define G_R^B to be the t-boundaried graph $G' \setminus (S' \cup P')$ with boundary B. Observe that G_R^B is also equal to $G \setminus (S \cup P)$ with boundary B. Thus we have

$$G' \setminus S' = G_R^B \oplus (G_{P'}^B \setminus S') \qquad G \setminus S = G_R^B \oplus (G_P^B \setminus S_P).$$

Since $(5(\eta + 1) + h)$-**folio**$(G_P^B \setminus S) \subseteq (5(\eta + 1) + h)$-**folio**$G_{P' \setminus S'}^B$ and $G_R^B \oplus (G_{P'}^B \setminus S')$ is $\mathcal{F}$-free, Lemma 23 implies that $G_R^B \oplus (G_P^B \setminus S_P) = G \setminus S$ is $\mathcal{F}$-free as well. Hence S is a $\mathcal{F}$-deletion set of G of size at most k. □

Lemmata 27 and 28 naturally lead to the following reduction rule. In particular Lemma 28 guarantees correctness, while Lemma 27 ensures that we can check in polynomial time whether the rule applies.

Reduction Rule 1. *Let P be an interesting set and let G' and P' be obtained from G and P respectively by deleting a vertex of P, deleting an edge with at least one endpoinnt in P, or contracting an edge with both endpoints in P. If the k-truncated signatures σ_P^k of P and $\sigma_{P'}^k$ of P' are identical, that is $\sigma_P^k(B, \mathcal{Q}) = \sigma_{P'}^k(B, \mathcal{Q})$ for every $(B, \mathcal{Q})$, then reduce (G, k) to (G', k).*

We now turn to the analysis of graphs to which Rule 1 can not be applied. Over the rest of this section we will show the following lemma.

Lemma 29. *There exists a constant γ depending only on $\mathcal{F}$ such that for any graph G on which Rule 1 does not apply, every component of $G \setminus X \cup Y$ has at most $\gamma \cdot k^\gamma$ vertices.*

We say that an interesting vertex set $P \subseteq V(G) \setminus (X \cup Y)$ has a *signature gap* if the following holds: for every set $B \subseteq \mathcal{B}_P$ and collection $\mathcal{Q}$ of t-boundaried graphs of size at most $|B| + h$ whose boundary is a subset of B, either $\sigma_P(B, \mathcal{Q}) \leq |B|$ or $\sigma_P(B, \mathcal{Q}) \geq k + 2$. We will prove that any large enough interesting set P contains a large interesting subset P' that has a signature gap. To that end we first prove a preliminary lemma. For a graph H and vertex set $P \subseteq C$, define $\zeta_P(H)$ to be the size of the smallest vertex set $S \subseteq P$ such that $G[P \setminus S]$ excludes H as a minor.

Lemma 30. *There exist constants λ and γ depending only on $\mathcal{F}$ such that for any interesting set P of size at least γk^γ there exists an interesting set $P' \subseteq P$ of size at least $\frac{|P|}{\lambda \cdot k^\lambda}$ and for every graph H of size at most $5(\eta + 1) + h$, either $\zeta_P(H) = 0$ or $\zeta_P(H) \geq k + 2$.*

Proof. Consider the nice tree-decomposition $(T, \mathcal{B} = \{B_t : t \in V(T)\})$ of width at most η of $G[C]$, where C is the component of $G \setminus (X \cup Z)$ that contains P. Let Q be the subtree of T such that $P = \{v \in V(G) : T_v \subseteq Q\}$. Since P is an interesting set such a subtree Q exists, and $|N(T_P)| \leq 2$. Now, set $M = N(Q)$. The set M refers to *marked* nodes in T. We run and analyze an algorithm that will modify Q and M. For a Q define $P(Q)$ to be $\{v \in V(G) : T_v \subseteq Q\}$. We will maintain the following invariants: $N_T(Q) = M$, $|M| \leq 2$ and Q induces a connected subtree of T. It is easy to verify that initially Q and M satisfy these invariants.

The algorithm proceeds as follows. If there does not exist a graph H on at most $5(\eta + 1) + h$ vertices such that $0 < \zeta_{P(Q)}(H) < k + 2$ output $P(Q)$. Otherwise there exists a set $S \subseteq P(Q)$ of size at most $k + 1$ such that $G[P(Q)] \setminus S$ excludes H as a minor. We build the set M' by starting from M and then for each vertex $v \in S$ select a node $t \in Q$ such that $v \in B_t$ and add it to M'. Then, set $M^* = $ **LCA-closure**(M'). Let Q' be the component of $Q \setminus M^*$ that maximizes $|P(Q')|$. Clearly Q' is a connected subtree of T and by Lemma 1, $|N(Q')| \leq 2$. The algorithm changes Q to Q' and $M = N(Q')$, maintaining the invariants. Since the size of Q drops in every iteration the algorithm eventually terminates, and when it terminates it satisfies that $P' = P(Q)$ is an interesting set and that for every graph H of size at most $5(\eta + 1) + h$, either $\zeta_{P'}(H) = 0$ or $\zeta_{P'}(H) \geq k + 2$. Thus it remains to show that $|P'| \geq \frac{|P|}{\lambda \cdot k^\lambda}$.

For every iteration of the algorithm the set $P(Q)$ shrinks to a subset of itself. Hence if $\zeta_{P(Q)}(H) = 0$ for a graph H at some step it will remain 0 in all future steps. Furthermore the graph H which had $0 < \zeta_{P(Q)}(H) < k + 2$ before the iteration satisfies $\zeta_{P(Q)}(H) = 0$ after. Thus in every iteration the number of graphs H of size at most $5(\eta + 1) + h$ with $\zeta_P(H) = 0$ increases by at least one. Since the number of graphs on at most $5(\eta+1)+h$ vertices is at most $2^{(5(\eta+1)+h)^2}$, the algorithm must terminate after this many iterations.

Finally, in every iteration of the algorithm the size of the set M' is at most $k + 4$ and hence by Lemma 1 M^* is at most $2k + 8$. Since every vertex in T has degree at most 3 the number of components of $T[Q \setminus M^*]$ is at most $6k + 24$. Hence

$$P(Q') \geq \frac{|P(Q)| - (\eta + 1)(2k + 8)}{6k + 24}.$$

As long as $|P(Q)| \geq 2(\eta + 1)(2k + 8)$ and $k \geq 1$ this is lower bounded by $\frac{|P(Q)|}{60k}$. Hence $|P(Q)|$ drops at most by a factor $60k$ in every iteration. Thus there exist constants λ and γ depending only on $\mathcal{F}$ such that for any interesting set P of size at least $\gamma \cdot k^{\gamma}$, the size of P' is at least $\frac{|P|}{\lambda \cdot k^{\lambda}}$. $\square$

Lemma 31. *There exist constants λ and γ depending only on $\mathcal{F}$ such that for any interesting set P of size at least γk^{γ} there exists an interesting set $P' \subseteq P$ of size at least $\frac{|P|}{\lambda \cdot k^{\lambda}}$ such that P' has a signature gap.*

Proof. Set the constants γ and λ as guaranteed by Lemma 30, and let $P' \subseteq P$ be a set of size at least $\frac{|P|}{\lambda \cdot k^{\lambda}}$ as guaranteed by Lemma 30. That is, for every graph H of size at most $5(\eta + 1) + h$, either $\zeta_{P'}(H) = 0$ or $\zeta_{P'}(H) \geq k + 2$. We prove that P' has a signature gap.

If there is at least one graph $H \in \mathcal{Q}$ that has no boundary and $\zeta_{P'}(H) \geq k + 2$ then $\sigma(\mathcal{Q}, B) \geq k + 2$ since any set S of size less than $k + 2$ can't hit all copies of H as a minor in $G[P']$, and hence can't hit all copies of H as a t-boundaried minor of $G^B_{P'}$. On the other hand if no such graph H exists then every $H \in \mathcal{Q}$ that has no boundary does not appear in $G^B_{P'}$ as a minor because H is not a minor of $G[P']$ and a model of H in $G^B_{P'}$ can not contain boundary vertices. Furthermore, the set B hits all t-boundaried graphs H that contain at least one boundary vertex. Hence $G^B_{P'} \setminus B$ is $\mathcal{Q}$-free and so P' has a signature gap. $\square$

We now show that if P is an interesting set with a signature gap and some vertex $x \in X$ has too many neighbours in X, then there exists an edge e from x to P such that deleting e does not affect the k-truncated signature of P. We first observe that we only need care about preserving the signature value for the pairs $(B, \mathcal{Q})$ such that $\sigma_P(B, \mathcal{Q}) \leq |B|$.

Lemma 32. *Let G be a graph and P be an interesting set in G with a signature gap. Let G' be obtained from G by deleting an edge e with at least one endpoint in P, contracting an edge e' with both endpoints in P or deleting a vertex in P. Let χ_P and χ_P^k be the signature and the k-truncated signature of P in G' respectively. If $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$ whenever $\sigma_P(B, \mathcal{Q}) \leq |B|$, then $\sigma_P^k(B, \mathcal{Q}) = \chi_P^k(B, \mathcal{Q})$ for all $(B, \mathcal{Q})$.*

Proof. Cleary $\sigma_P^k(B, \mathcal{Q}) \geq \chi_P^k(B, \mathcal{Q})$ for all $(B, \mathcal{Q})$ so it suffices to show that $\sigma_P(B, \mathcal{Q}) > |B|$ implies $\sigma_P^k(B, \mathcal{Q}) \leq \chi_P^k(B, \mathcal{Q})$. If $\sigma_P(B, \mathcal{Q}) > |B|$ we have that $\sigma_P(B, \mathcal{Q}) \geq k + 2$ since P has a signature gap. Then $\sigma_P^k(B, \mathcal{Q}) = \infty$ and $\chi_P(B, \mathcal{Q}) \geq k + 1$ so $\chi_P^k(B, \mathcal{Q}) = \infty$ as well, concluding the proof. $\square$

Now we need a few definitions. For a pair of integers t, β and finite set $\mathcal{Q}$ of t-boundaried graphs, define the class $\Pi_{\mathcal{Q}}^{(t,\beta)}$ to be the set of all t-boundaried graphs of treewidth at most t, such that every graph $G \in \Pi_{\mathcal{Q}}^{(t,\beta)}$ has a vertex set S of size at most β such that $G \setminus S$ is $\mathcal{Q}$-free.

For every choice of t, β and $\mathcal{Q}$ the corresponding class $G \in \Pi_{\mathcal{Q}}^{(t,\beta)}$ is closed under minors, since taking minors can not increase treewidth nor increase the size of the smallest $\mathcal{Q}$-deletion set in G. Thus, by Lemma 24 for each such class $\Pi_{\mathcal{Q}}^{(t,\beta)}$ there is a finite set $\mathcal{F}_{\mathcal{Q}}^{(t,\beta)}$ such that a t-boundaried graph G is in $\Pi_{\mathcal{Q}}^{(t,\beta)}$ if and only if it is $\mathcal{F}_{\mathcal{Q}}^{(t,\beta)}$-free.

We also introduce the notion of a vital set in a minor model. Let $\mathcal{P}_H := \{P_1, \ldots, P_{|V(H)|}\}$ be a minor model of H in G, and let $h := |V(H)|$. The vital trees of $\mathcal{P}_H$ are given by $\{T_1, \ldots, T_{|V(H)|}\}$, where $T_i \subseteq E(P_i)$ induces a spanning tree of P_i. Let $\mathcal{E}_H := \{e_i \mid 1 \leq i \leq |E(H)|\}$ denote a set of edges in G which witness the adjacencies of H in $\mathcal{P}_H$, that is, if $(i,j) \in E(H)$, then e_i represents some $uv \in E(G)$ such that $u \in P_i$ and $v \in P_j$. be The vital edges of P_H are defined as $\bigcup_{1 \leq i \leq |V(H)|} T_i \cup \mathcal{E}_H$. The following observation is easy to make:

Observation 5. *Let* $\mathcal{P}_H := \{P_1, \ldots, P_{|V(H)|}\}$ *be a minimal minor model of H in G, where* $h := |V(H)|$. *Let* $\{T_1, \ldots, T_{|V(H)|}\}$ *be the vital trees of $\mathcal{P}_H$, and let $\mathcal{E}_H$ be the set of vital edges. Then every T_i has at most h leaves, and no vertex in any P_i is incident to more than h vital edges.*

Proof. We first show that the vital trees of a minimal minor model have at most h leaves. Suppose not, and let T_i be the vital tree with more than h leaves. This implies that there is at least one leaf vertex v which is not adjacent to any vertex outside P_i. Note that $\{P_1, \ldots, P_i \setminus \{v\}, \ldots, P_{|V(H)|}\}$ continues to be a minor model of H, since deleting a leaf vertex from a spanning tree of P_i does not affect the connectivity of P_i and the adjacencies of H are maintained since v was not adjacent to any vertex outside P_i. This contradicts the assumption that $\mathcal{P}_H$ was a minimal minor model of H.

Since T_i has at most h leaves, the degree of the internal vertices of T_i (and hence, P_i) is bounded by h. A leaf vertex in P_i has one neighbor in P_i and is adjacent at most $(h-1)$ vertices outside P_i (note that in a minimal minor model, a leaf of T_i is never adjacent to more than one vertex in P_j, $j \neq i$). Therefore, every vertex in P_i is adjacent to at most h vital edges. $\qquad\square$

We are now ready to show that in a irreducible graph a vertex in X can't have too many neighbours in an interesting set with a signature gap.

Lemma 33. *There exists a constant μ depending only on $\mathcal{F}$ such that if $x \in X$ has at least $\mu \cdot k^\mu$ neighbors in an interesting set P that has a signature gap, then there is an edge e from x to P such that the k-truncated signature σ_P remains unchanged when e is deleted from G.*

Proof. Recall that σ_P is the signature and σ_P^k is the k-truncated signature of P respectively. For an edge e let $G' = G \setminus e$ and let χ_P be the signature of P in G'. Similarly, let χ_P^k be the k-truncated signature of P in G'. By Lemma 32 it is sufficient to find an edge e incident to x such that $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$ whenever $\sigma_P(B, \mathcal{Q}) \leq |B|$.

For every pair $(B, \mathcal{Q})$ such that $\sigma_P^k(B, \mathcal{Q}) \leq |B|$, set $\beta = \sigma_P^k(B, \mathcal{Q})$. We have that $G_P^B \in \Pi_{\mathcal{Q}}^{(\eta,\beta)}$, but also that $G_P^B \notin \Pi_{\mathcal{Q}}^{(\eta,\beta-1)}$. Hence G_P^B contains a graph $H \in \mathcal{F}_{\mathcal{Q}}^{(\eta,\beta-1)}$ as a minor. Mark a vital set of edges of a minimal model of H in G_P^B. Do this marking for every pair $(B, \mathcal{Q})$ such that $\sigma_P^k(B, \mathcal{Q}) \leq |B|$. The size of any graph H in $\mathcal{F}_{\mathcal{Q}}^{(\eta,\beta-1)}$ is upper bounded by a function depending only on $\mathcal{F}$. Thus, for each pair $(B, \mathcal{Q})$, by Observation 5 the number of edges incident to x that are marked is bounded by a constant depending only on $\mathcal{F}$. The number of pairs $(B, \mathcal{Q})$ is bounded by $O(k^{5(\eta+1)})$ and hence, the exists a constant μ depending only on $\mathcal{F}$ such that if $x \in X$ has at least $\mu \cdot k^\mu$ neighbors P, then at least one edge e is left unmarked by the process above. We claim that $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$ whenever $\sigma_P(B, \mathcal{Q}) \leq |B|$.

Suppose for contradiction that $\sigma_P(B, \mathcal{Q}) \leq |B|$ but $\sigma_P(B, \mathcal{Q}) > \chi_P(B, \mathcal{Q})$ for some pair $(B, \mathcal{Q})$. Set $\beta = \sigma_P^k(B, \mathcal{Q})$. We have that $G_P^B \notin \Pi_{\mathcal{Q}}^{(\eta,\beta-1)}$. Hence G_P^B contains a graph $H \in \mathcal{F}_{\mathcal{Q}}^{(\eta,\beta-1)}$ as a minor. Furthermore, all vital edges of a model of H in G_P^B are marked. Since the edge e is unmarked H is also a minor of $G_P^B \setminus e$, contradicting that $\chi_P^k(B, \mathcal{Q}) \leq \beta - 1$. $\qquad\square$

Lemma 31 allows us to find a large interesting set P with a signature gap. Lemma 33 shows that in a graph where Reduction Rule 1 does not apply, every vertex in X has few neighbours inside P. We now that if P is a large interesting set and X has few neoghbours in P then P contains a large interesting subset P' that has a signature gap and contains no neighbours of X.

Lemma 34. *There exist constants ρ and τ such that for any interesting set P of size at least $\rho k^\rho \cdot (|P \cap N(X)|)$ there is an interesting set $P' \subseteq P$ of size at least $\frac{|P|}{\tau \cdot k^\tau \cdot |P \cap N(X)|}$, such that P' has a signature gap and $P' \cap N(X) = \emptyset$.*

Proof. Since P is interesting there is a subtree Q of T such that $|N(Q)| \leq 2$ and $P = \{v \in C : T_v \subset Q\}$. Initially, set $M = N(Q)$ and then, for each $v \in N(X) \cap P$ we insert a node in T_v into M. We let $M^* = \textbf{LCA-closure}(M)$ and let Q^* be the component of $Q \setminus M$ that maximizes $|\{v \in C : T_v \subseteq Q^*\}|$. Set $P^* = \{v \in C : T_v \subseteq Q^*\}$. We have that $|M| \leq |P \cap N(X)|$ and by Lemma 1 $M^* \leq 2|M|$. Thus the number of connected components of $Q \setminus M$ is at most $6|P \cap N(X)|$. From this is follows that $|P^*| \geq \frac{|P| - (\eta+1)2|P \cap N(X)|}{6|P \cap N(X)|} \geq \frac{|P|}{12|P \cap N(X)|}$. Now, P^* is an interesting set, as evidenced by Q^* and the fact that $|N(Q^*)| \leq 2$. Furthermore, $P^* \cap N(X) = \emptyset$. If ρ is chosen appropriately then we can invoke Lemma 31 and obtain an interesting set $P' \subseteq P$ with a signature gap. The size of P' is lower bounded by $\frac{|P|}{\tau \cdot k^\tau \cdot |P \cap N(X)|}$ for a constant τ. $\square$

Now we show that Reduction Rule 1 will reduce large enough interesting sets with a signature gap and no neighbors in X.

Lemma 35. *There exists a constant ω depending only on $\mathcal{F}$ such that for any interesting set P such that $|P| > \omega$, P has a signature gap and $P \cap N(X) = \emptyset$, there is an edge e with both endpoints in P such that deleting or contacting e does not change the k-truncated signature of P, or a vertex v such that deleting v does not change the k-truncated signature of P.*

Proof. We show the existence of a minor operation inside P that produces a graph G' with the following properties. Let χ_P and χ_P^k BE the signature and the k-truncated signature of P in G' respectively. Then, if $\sigma_P(B, \mathcal{Q}) \leq |B|$ then $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$. By Lemma 32 this implies that the minor operation in fact preserves the k-truncated signature of P.

For each pair $(B, \mathcal{Q})$ such that $\sigma_P(B, \mathcal{Q}) \leq |B| \leq 5(\eta + 1)$, let $\beta = \sigma_P(B, \mathcal{Q})$. We have that G_P^B contains a boundaried graph $H_{B,\mathcal{Q}} \in \mathcal{F}_\mathcal{Q}^{(t,\beta-1)}$ as a minor. Thus $G_P^{N(P)}$ contains $H_{B,\mathcal{Q}}$ as a minor as well. Let $\mathcal{R}$ be the collection of all graphs $H_{B,\mathcal{Q}}$ for pairs $(B, \mathcal{Q})$ such that $\sigma_P(B, \mathcal{Q}) \leq |B|$. If $G_P^{N(P)}$ contains all graphs in $\mathcal{R}$ even after the minor operation, then $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$ for all $(B, \mathcal{Q})$ such that $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$.

Consider the collection $\mathcal{S}$ of $5(\eta + 1)$ boundaried graphs of treewidth at most η, such that each graph $J \in \mathcal{S}$ contains all graphs in $\mathcal{R}$ as a minor, but no minor of J contains all graphs in $\mathcal{R}$ as a minor. No two graphs in $\mathcal{S}$ are minors of each other and hence by Lemma 24 $\mathcal{S}$ is finite. Let ω be the size of the largest graph in $\mathcal{S}$. Since ω only depends on the values of $\sigma_P(B, \mathcal{Q})$, each value of $\sigma_P(B, \mathcal{Q})$ is bounded by $5(\eta + 1)$ and the number of choices for $(B, \mathcal{Q})$ is a function of η only, ω is upper bounded by a function of $\mathcal{F}$. Furthermore, $G_P^{N(P)}$ contains all graphs in $\mathcal{R}$ as a minor, and if $|P| > \omega$, $G_P^{N(P)}$ has a minor that also contains all graphs in $\mathcal{R}$ as minors. Thus there is a minor opreration that ensures that if $\sigma_P(B, \mathcal{Q}) \leq |B|$ then $\sigma_P(B, \mathcal{Q}) = \chi_P(B, \mathcal{Q})$, proving the lemma. $\square$

We are now in position to tie the results of the section together and prove Lemma 29.

Proof of Lemma 29. If $|C| \geq \gamma \cdot k^\gamma$ then by Lemma 31 there is an interesting set P of size $\frac{\gamma}{\lambda} \cdot k^{\gamma-\lambda}$ with a signature gap. If some vertex $x \in X$ has at least $\mu \cdot k^\mu$ neigbours in P then by Lemma 33, Rule 1 applies to P. If no vertex $x \in X$ has at least $\mu \cdot k^\mu$ neigbours in P then by

40

Lemma 34 P has an interesting subset P' with a signature gap such that $|P'| \geq \frac{\gamma}{\lambda \cdot \mu \cdot \tau} \cdot k^{\gamma - \lambda - \mu - \tau}$ and $P \cap N(X) = \emptyset$. For a sufficiently large choice of γ (still depending only on $\mathcal{F}$), we have that $|P'| > \omega$ and hence by Lemma 35, Rule 1 applies on P'. $\qquad\square$

8.4 Reducing the number of near-Protrusions

If Rule 1 can not be applied but the input graph G is still large, the number of components of $G \setminus X$ must be large. We now show that if the number of components is too large then we can remove a vertex from one of them. For a component C, set $B \in \mathcal{B}_C$ and a boundaried graph H with B as boundary, we say that C *realizes* (B, H) if $H \leq_m G_C^B$. We say that a pair (B, H) is *rich* if there are at least $|X| + |Z| + k + (h + 3(\eta + 1))^2 + 2$ components. We prove the following lemma, which naturally leads to a reduction rule.

Lemma 36. *Let C be a component of $G \setminus (X \cup Z)$ such that every pair (B, H) that C realizes is rich. Then, for any vertex $v \in C$, $G \setminus v$ has an F-deletion set of size at most k if and only if G does.*

Proof. If G has a $\mathcal{F}$-deletion set S of size at most k, then $S \setminus \{v\}$ is a $\mathcal{F}$-deletion set of G, hence it suffices to show the reverse direction. Assume now that $G \setminus v$ has an $\mathcal{F}$-deletion set S of size at most k. We argue that S is in fact a $\mathcal{F}$-deletion set of G as well. Suppose not, then $S \cap v$ is an $\mathcal{F}$-deletion set of G of size at most $k + 1$ and hence by Lemma 26, $|(N(C) \cap X) \setminus S| \leq \eta + 1$. Set $B = N(C) \setminus S$ and $R = V(G) \setminus (C \cup S \cup B)$. We have that $|B| \leq 3(\eta + 1)$ (and hence $B \in \mathcal{B}_C$) and that $G \setminus S = G_C^B \oplus G_R^B$. Since $G \setminus S$ is not $\mathcal{F}$-free, by Lemma 23 there are boundaried graphs H_1 and H_2 on at most $h + 3(\eta + 1)$ vertices each such that $H \leq_m H_1 \oplus H_2$, $H_1 \leq_m G_C^B$ and $H_2 \leq_m G_R^B$.

Suppose that we show the existence of a component $C' \neq C$ of $G \setminus (X \cup Z)$ such that $C' \subseteq R$ such that $C' \cap S = \emptyset$, $H_1 \leq_m G_{C'}^B$ and $H_2 \leq_m (G_R^B \setminus C')$. Then Lemma 23 would yield that H is a minor of $G_{C'}^B \oplus (G_R^B \setminus C') = G \setminus (S \cup C)$, contradicting that $S \cup v$ is an $\mathcal{F}$-deletion set of G. It remains to find such a component C'. We prove the lemma assuming the following claim.

Claim 1. *Any minimal minor model of H_2 in G_R^B has a non-empty intersection with at most $|X| + |Z| + (h + 3(\eta + 1))^2$ components of $G \setminus (X \cup Z)$.*

Since C realizes (B, H_1), we have that (B, H_1) is rich and so there are at least $|X| + |Z| + k + (h + 3(\eta + 1))^2 + 2$ components of $G \setminus (X \cup Z)$ that realize (B, H_1). Consider any minimal minor model of H_2 in G_R^B. By Claim 1 there are at least $k + 2$ components of $G \setminus (X \cup Z)$ that realize (B, H_1) and are disjoint from this model of H_2. At least two of these are disjoint from S, and at least one of these two is not C. Let this component be C'. We have that $C' \neq C$, $C' \cap S = \emptyset$, $H_1 \leq_m G_{C'}^B$ and $H_2 \leq_m (G_R^B \setminus C')$. This completes the proof of the lemma, up to proving Claim 1.

Proof. Proof of Claim 1. Consider a minimal model $(P_1, P_2, \ldots, P_\ell)$ of H_2 in G_R^B. Here $\ell = |V(H_2)| \leq h + 3(\eta + 1)$. For every $i \leq \ell$ select a spanning tree T_i of $G[P_i]$. Since $(P_1, P_2, \ldots, P_\ell)$ is a minimal model, each tree T_i has at most ℓ leaves. Consider the connected components of $T_i \setminus (X \cup Z)$. Each component has at least one edge to $X \cup Z$ and the components that do not contain leaves have at least two edges to $X \cup Z$. Counting the total number of vertices and edges incident to each component and using that $E(T_i) = V(T_i) + 1$ yields that the number of components of $T_i \setminus (X \cup Z)$ is at most $|V(T_i) \cap (X \cup Z)| - 1 + \ell - 1$. Since the intersection of T_i with different components of $G \setminus (X \cup Z)$ must be in different components of $T_i \setminus (X \cup Z)$ it follows that the number of components that intersect with T_i is at most $|V(T_i) \cap (X \cup Z)| - 1 + \ell - 1$ as well. Hence the number of components with non-empty intersection with the model of H_2 is at most $\sum_{i \leq \ell} |V(T_i) \cap (X \cup Z)| + \ell - 2 \leq |X \cup Z| + \ell^2$. Since $\ell \leq h + 3(\eta + 1)$ this proves the claim. $\qquad\square$

$\square$

For each component C of $G \setminus (X \cup V)$ and set B of size at most $3(\eta + 1)$ the boundaried graph G_C^B has constant treewidth. For every boundaried graph H on at most $h + 3(\eta + 1)$ vertices we can encode whether G_C^B contains H as a minor in monadic second order logic. Thus for each choice of C, $B \in \mathcal{B}_C$ and H we can test whether C realises a pair (B, H) in linear time. Since the number of pairs $B \in \mathcal{B}_C$ is bounded by a polynomial and the number of boundaried graphs on at most $h + 3(\eta + 1)$ vertices is constant this means we can test in polynimial time whether there exists a component C of $G \setminus (X \cup Z)$ such that every pair (B, H) that C realizes is rich, and find such a component if it does. This, together with Lemma 36 yields the following reduction rule.

Reduction Rule 2. *If there is a component C of $G \setminus (X \cup Z)$ such that every pair (B, H) that C realizes is rich, remove an arbitrary vertex v from C.*

Lemma 36 shows that Rule 2 is safe, it remains to show that if the number of components of $G \setminus (X \cup Y)$ is too large, then Rule 2 applies.

Lemma 37. *There exists a constant θ depending only on $\mathcal{F}$ such that if the number of components of $G \setminus (X \cup Z)$ is at least $\theta \cdot k^\theta$ then Rule 2 applies.*

Proof. If Rule 2 does not apply then every component C can realize at least one pair (B, H) that is not rich. The number of choices for B is at most $|X \cup Z|^{3(\eta+1)}$ while the number of choices for H is at most $2^{\binom{3(\eta+1)+h}{2}}$. For each choice of (B, H) that is poor, at most $|X| + |Z| + k + (h + 3(\eta + 1))^2 + 1$ components realize that pair. Thus by the pigeon hole principle the number of components is at most $\theta \cdot k^\theta$ for some constant θ depending only on $\mathcal{F}$. $\square$

Lemmata 25, 29 and 37 together imply that every p-$\mathcal{F}$-DELETION problem has a polynomial kernel. Specifically, Lemma 25 proves that in a reduced graph $|X \cup Z| \leq O(k^3)$, Lemma 29 shows that the size of each component of $G \setminus (X \cup Z)$ is $k^{O(1)}$ while Lemma 37 shows that the number of components of $G \setminus (X \cup Z)$ is $k^{O(1)}$. This proves the following theorem.

Theorem 15. *Every p-$\mathcal{F}$-DELETION problem has a randomized polynomial kernel.*

We now turn to two interesting consequences of Theorem 15. The first concerns a bound on the obstructions for the family $\mathcal{G}_{\mathcal{F},k}$, which contains all yes instances of the p-$\mathcal{F}$-DELETION problem with parameter k. We first make the following observation:

Observation 6. *For a graph $G \notin \mathcal{G}_{\mathcal{F},k}$, let H be some graph in the obstruction set of $\mathcal{G}_{\mathcal{F},k}$, and let H^* be a minimal minor model of H in G. Then, $H^* \in \mathcal{G}_{\mathcal{F},k+1}$.*

Proof. Clearly, $H^* \notin \mathcal{G}_{\mathcal{F},k}$. If $H^* \notin \mathcal{G}_{\mathcal{F},k+1}$, then $H^* \in \mathcal{G}_{\mathcal{F},j}$ for some $j > k + 1$, and this contradicts the minimality of H^*. $\square$

The first step in the kernelization algorithm used the approximation algorithm to reject all instances where the size of the optimal solution was at least $(k + 2)$. Therefore, by Observation 6, all minimal minor models of graphs in the obstruction set of $\mathcal{G}_{\mathcal{F},k}$ are present in any non-trivial reduced instance. The other two reduction rules performed by the kernelization algorithm involve only minor operations, which clearly do not destroy minimal minor models of the obstruction graphs. Therefore, we have the following corollary.

Corollary 1. *When $\mathcal{F}$ contains a planar graph, every minimal obstruction for $\mathcal{G}_{\mathcal{F},k}$ is of size polynomial in k.*

By standard arguments that translate a kernelization algorithm into a FPT algorithm, we also have the following consequence.

Corollary 2. *Every p-$\mathcal{F}$-DELETION problem admits an $O(k^k)$ FPT algorithm.*

Finally a note about computability; the implementation of the decomposition algorithm and the two rules is entirely constructive. That is, there is an algorithm that given G, k and $\mathcal{F}$, runs in time $O(n^c)$ and outputs a reduced instance. Here the constant c depends on $\mathcal{F}$ and is upper bounded by $2^{2^{h^{10}}}$, where h is the size of the largest graph in $\mathcal{F}$. A slightly tighter analysis reveals that the running time is actually $n^2 \cdot k^c$ which of course is still horrendous, but at least it is quadratic for fixed k. The *size* of the kernel, however, is not explicit. Several of the constants that go into the proof of Lemma 29 depend on the size of the largest graph in certain antichains in a well-quasi-order and thus we dont know what the (constant) exponent bounding the size of the kernel is. We leave it to future work to make also the size of the kernel explicit.

9   Conclusions and open problems

The techniques of fast protrusion reductions developed for p-$\mathcal{F}$-DELETION have a broader spectrum of applications which we mention briefly. By combining results from [50] with fast protrusion reducers, we have that kernelization algorithms on apex-free and H-minor free graphs for *all* bidimensional problems from [50] can be implemented in linear time if we use randomized protrusion reducer and in time $O(n \log^2 n)$ when we use deterministic reducer. This gives randomized linear time linear kernels for a multitude of problems.

In the framework for obtaining EPTAS on H-minor-free graphs in [48], the running time of approximation algorithms for many problems is $f(1/\varepsilon) \cdot n^{O(g(H))}$, where g is some function of H only. The only bottleneck for improving polynomial time dependence in [48] is Lemma 4.1, which gives a constant factor approximation algorithm for TREEWIDTH η-DELETION or η-TRANSVERSAL of running time $n^{O(g(H))}$. Now instead of that algorithm, we can use the algorithm from Theorem 1, which runs in time $O(n^2)$. Therefore each EPTAS from [48] runs in time $O(f(1/\varepsilon) \cdot n^2)$. For the same reason, PTAS for many problems on unit disc and map graphs from [49] become EPTAS.

Finally, an interesting direction for further research is to investigate p-$\mathcal{F}$-DELETION when none of the graphs in $\mathcal{F}$ is planar. The most interesting case here is when $\mathcal{F} = \{K_5, K_{3,3}\}$ aka the VERTEX PLANARIZATION problem. Surprisingly, we are not aware even of a single case of p-$\mathcal{F}$-DELETION with $\mathcal{F}$ containing no planar graph admitting either constant factor approximation, or polynomial kernelization, or parameterized single-exponential algorithms. It is tempting to conjecture that the line of tractability is determined by whether $\mathcal{F}$ contains a planar graph or not.

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(3):411–430, 2007.

[2] I. Adler, M. Grohe, and S. Kreutzer. Computing excluded minors. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2008)*, pages 641–650. ACM-SIAM, 2008.

[3] N. Alon, G. Gutin, E. J. Kim, S. Szeider, and A. Yeo. Solving max-r-sat above a tight lower bound. In *SODA 2010*, pages 511–517. ACM-SIAM, 2010.

[4] S. Arnborg, B. Courcelle, A. Proskurowski, and D. Seese. An algebraic theory of graph reduction. *J. ACM*, 40(5):1134–1164, 1993.

[5] V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM Journal on Discrete Mathematics*, 12(3):289–297, 1999.

[6] R. Bar-Yehuda and S. Even. A linear-time approximation algorithm for the weighted vertex cover problem. *J. Algorithms*, 2(2):198–203, 1981.

[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 Journal on Computing*, 27(4):942–959, 1998.

[8] A. Becker, R. Bar-Yehuda, and D. Geiger. Randomized algorithms for the loop cutset problem. *J. Artificial Intelligence Res.*, 12:219–234, 2000.

[9] R. Bodendiek and K. Wagner. Solution to König's graph embedding problem. *Math. Nachr.*, 140:251–272, 1989.

[10] H. Bodlaender. Treewidth: Algorithmic techniques and results. In *MFCS'97: Mathematical Foundations of Computer Science 1997, 22nd International Symposium (MFCS)*, volume 1295 of *Lecture Notes in Computer Science*, pages 19–36. Springer, 1997.

[11] H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[12] H. L. Bodlaender. A cubic kernel for feedback vertex set. In *STACS 2007*, volume 4393 of *LNCS*, pages 320–331. Springer, 2007.

[13] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proceedings of the 4th Workshop on Parameterized and Exact Computation (IWPEC 2009)*, volume 5917 of *Lecture Notes in Computer Science*, pages 17–37, 2009.

[14] H. L. Bodlaender and B. de Fluiter. Reduction algorithms for constructing solutions in graphs with small treewidth. pages 199–208, 1996.

[15] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.

[16] H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (meta) Kernelization. In *FOCS 2009*, pages 629–638. IEEE, 2009.

[17] H. L. Bodlaender, F. V. Fomin, D. Lokshtanov, E. Penninkx, S. Saurabh, and D. M. Thilikos. (Meta) kernelization. In *Proc. of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638, 2009.

[18] H. L. Bodlaender and T. Hagerup. Parallel algorithms with optimal speedup for bounded treewidth. *SIAM J. Comput.*, 27:1725–1746, 1998.

[19] H. L. Bodlaender and B. van Antwerpen-de Fluiter. Reduction algorithms for graphs of small treewidth. 167:86–119, 2001.

[20] 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*, 7:555–581, 1992.

[21] R. B. Borie, R. G. Parker, and C. A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7(5&6):555–581, 1992.

[22] 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 IWPEC 2006*, volume 4169 of *LNCS*, pages 192–202. Springer, 2006.

[23] Y. Cao, J. Chen, and Y. Liu. On feedback vertex set new measure and new structures. In *Proc. of the 12th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2010)*, volume 6139 of *LNCS*, pages 93–104, 2010.

[24] J. Chen, B. Chor, M. Fellows, X. Huang, D. W. Juedes, I. A. Kanj, and G. Xia. Tight lower bounds for certain parameterized NP-hard problems. *Information and Computation*, 201(2):216–231, 2005.

[25] J. Chen, F. V. Fomin, Y. Liu, S. Lu, and Y. Villanger. Improved algorithms for feedback vertex set problems. *J. Comput. Syst. Sci.*, 74(7):1188–1198, 2008.

[26] J. Chen, I. A. Kanj, and W. Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41(2):280–301, 2001.

[27] J. Chen, I. A. Kanj, and G. Xia. Improved upper bounds for vertex cover. *Theor. Comput. Sci.*, 411(40-42):3736–3756, 2010.

[28] B. Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, chapter 5. World Scientific, 1997.

[29] B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theor. Comput. Sci.*, 109(1&2):49–82, 1993.

[30] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. van Rooij, and J. O. Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *Proceedings of the 52nd Annual Symposium on Foundations of Computer Science (FOCS 2011)*, pages 150–159. IEEE, 2011.

[31] M. Cygan, M. Pilipczuk, M. Pilipczuk, and J. O. Wojtaszczyk. An improved fpt algorithm and quadratic kernel for pathwidth one vertex deletion. In *IPEC*, volume 6478 of *Lecture Notes in Computer Science*, pages 95–106, 2010.

[32] B. de Fluiter. *Algorithms for Graphs of Small Treewidth*. PhD thesis, Utrecht University, 1997.

[33] F. K. H. A. Dehne, M. R. Fellows, M. A. Langston, F. A. Rosamond, and K. Stevens. An $O(2^{O(k)}n^3)$ FPT algorithm for the undirected feedback vertex set problem. *Theory Comput. Syst.*, 41(3):479–492, 2007.

[34] 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 *IWPEC 2004*, volume 3162 of *LNCS*, pages 271–280. Springer, 2004.

[35] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *STOC 2010*, 2010. To appear.

[36] M. J. Dinneen, K. Cattell, and M. R. Fellows. Forbidden minors to graphs with small feedback sets. *Discrete Mathematics*, 230(1-3):215–252, 2001.

[37] M. J. Dinneen and L. Xiong. Minor-order obstructions for the graphs of vertex cover 6. *J. Graph Theory*, 41(3):163–178, 2002.

[38] H. Djidjev and J. H. Reif. An efficient algorithm for the genus problem with explicit construction of forbidden subgraphs. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC 91)*, pages 337–347. ACM, 1991.

[39] R. G. Downey and M. R. Fellows. *Parameterized complexity*. Springer-Verlag, New York, 1999.

[40] Z. Dvořák, A. C. Giannopoulou, and D. M. Thilikos. *European Journal of Combinatorics*, (5):969.

[41] M. R. Fellows and M. A. Langston. Nonconstructive tools for proving polynomial-time decidability. *J. ACM*, 35(3):727–739, 1988.

[42] M. R. Fellows and M. A. Langston. An analogue of the myhill-nerode theorem and its use in computing finite-basis characterizations (extended abstract). In *FOCS*, pages 520–525, 1989.

[43] M. R. Fellows and M. A. Langston. An analogue of the myhill-nerode theorem and its use in computing finite-basis characterizations (extended abstract). In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS 1989)*, pages 520–525. IEEE Computer Society, 1989.

[44] M. R. Fellows and M. A. Langston. On search, decision, and the efficiency of polynomial-time algorithms. *J. Comput. Syst. Sci.*, 49(3):769–779, 1994.

[45] 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)*, volume 6080 of *Lecture Notes in Comput. Sci.*, pages 191–204. Springer, 2010.

[46] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.

[47] F. V. Fomin, D. Lokshtanov, N. Misra, G. Philip, and S. Saurabh. Hitting forbidden minors: Approximation and kernelization. In *Proceedings of the 8th International Symposium on Theoretical Aspects of Computer Science (STACS 2011)*, volume 9 of *LIPIcs*, pages 189–200. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[48] F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Bidimensionality and EPTAS. In *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2011)*, pages 748–759. SIAM, 2011.

[49] F. V. Fomin, D. Lokshtanov, and S. Saurabh. Bidimensionality and geometric graphs. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012)*, pages 1563–1575. SIAM, 2012.

[50] 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)*, pages 503–510, 2010.

[51] L. Fortnow and R. Santhanam. Infeasibility of instance compression and succinct PCPs for NP. In *STOC 08*, pages 133–142. ACM, 2008.

[52] J. Guo, J. Gramm, F. Hüffner, R. Niedermeier, and S. Wernicke. Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. *J. Comput. Syst. Sci.*, 72(8):1386–1396, 2006.

[53] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT news*, 38(1):31–45, 2007.

[54] A. Gupta and R. Impagliazzo. Computing planar intertwines. In *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science (FOCS 1991)*, pages 802–811. IEEE Computer Society, 1991.

[55] D. S. Hochbaum and J. Naor. Simple and fast algorithms for linear and integer programs with two variables per inequality. *SIAM J. Comput.*, 23(6):1179–1192, 1994.

[56] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity. *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

[57] G. Joret, C. Paul, I. Sau, S. Saurabh, and S. Thomassé. Hitting and harvesting pumpkins. In *Proceedings of the 19th Annual European Symposium on Algorithms (ESA2011)*, volume 6942 of *Lect. Notes Comp. Sc.*, pages 394–407. Springer, 2011.

[58] E. J. Kim, C. Paul, and G. Philip. A single-exponential FPT algorithm for K_4-minor cover problem. 2012.

[59] T. Kloks. *Treewidth – computations and approximations*, volume 842 of *LNCS*. Springer, 1994.

[60] S. Kratsch and M. Wahlström. Compression via matroids: a randomized polynomial kernel for odd cycle transversal. In *Proceedings of the 23d Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012)*, pages 94–103, 2012.

[61] J. Lagergren. Upper bounds on the size of obstructions and intertwines. *J. Combin. Theory Ser. B*, 73(1):7–40, 1998.

[62] A. Langer, F. Reidl, P. Rossmanith, and S. Sikdar. Linear kernels on graphs excluding topological minors. *CoRR*, abs/1201.2780, 2012.

[63] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is np-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980.

[64] C. Lund and M. Yannakakis. The approximation of maximum subgraph problems. In *Proceedings of the 20nd International Colloquium on Automata, Languages and Programming (ICALP 1993)*, volume 700 of *Lecture Notes in Comput. Sci.*, pages 40–51, 1993.

[65] M. Naor, L. J. Schulman, and A. Srinivasan. Splitters and near-optimal derandomization. In *FOCS*, pages 182–191, 1995.

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

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

[68] G. Philip, V. Raman, and S. Sikdar. Polynomial kernels for dominating set in $K_{i,j}$ -free and d-degenerate graphs, manuscript, 2009.

[69] G. Philip, V. Raman, and Y. Villanger. A quartic kernel for pathwidth-one vertex deletion. In *WG*, volume 6410 of *Lecture Notes in Computer Science*, pages 196–207, 2010.

[70] V. Raman, S. Saurabh, and C. R. Subramanian. Faster fixed parameter tractable algorithms for finding feedback vertex sets. *ACM Transactions on Algorithms*, 2(3):403–415, 2006.

[71] B. Reed, K. Smith, and A. Vetta. Finding odd cycle transversals. 32(4):299–301, 2004.

[72] N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *J. Combin. Theory Ser. B*, 41(1):92–114, 1986.

[73] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *J. Combin. Theory Ser. B*, 63:65–110, 1995.

[74] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *J. Combin. Theory Ser. B*, 63(1):65–110, 1995.

[75] S. Thomassé. A quadratic kernel for feedback vertex set. In *SODA 2009*, pages 115–119. ACM-SIAM, 2009.

[76] M. Yannakakis. The effect of a connectivity requirement on the complexity of maximum subgraph problems. *J. ACM*, 26(4):618–630, 1979.

[77] M. Yannakakis. Some open problems in approximation. In *Proceedings of the 2nd Italian Conference on Algorithms and Complexity (CIAC 1994)*, volume 778 of *Lecture Notes in Comput. Sci.*, pages 33–39, 1994.