

# Protrusion Decompositions Revisited: Uniform Lossy Kernels for Reducing Treewidth and Linear Kernels for Hitting Disconnected Minors

Roohani Sharma ✉ 

Discrete Mathematics Group, Institute for Basic Science (IBS), Daejeon, South Korea

Michał Włodarczyk ✉ 

Institute of Informatics, University of Warsaw, Poland

---

## Abstract

---

Let  $\mathcal{F}$  be a finite family of graphs. In the  $\mathcal{F}$ -DELETION problem, one is given a graph  $G$  and an integer  $k$ , and the goal is to find  $k$  vertices whose deletion results in a graph with no minor from the family  $\mathcal{F}$ . This may be regarded as a far-reaching generalization of VERTEX COVER and FEEDBACK VERTEX SET. In their seminal work, Fomin, Lokshtanov, Misra & Saurabh [FOCS 2012] gave a polynomial kernel for this problem when the family  $\mathcal{F}$  contains a planar graph. As the size of their kernel is  $g(\mathcal{F}) \cdot k^{f(\mathcal{F})}$ , a natural follow-up question was whether the dependence on  $\mathcal{F}$  in the exponent of  $k$  can be avoided. The answer turned out to be negative: Giannopoulou, Jansen, Lokshtanov & Saurabh [TALG 2017] proved that this is already inevitable for the special case of the TREEWIDTH- $\eta$ -DELETION problem.

In this work, we show that this non-uniformity can be avoided at the expense of a small loss. First, we present a simple 2-approximate kernelization algorithm for TREEWIDTH- $\eta$ -DELETION with a kernel size  $g(\eta) \cdot k^6$ . Next, we show that the approximation factor can be made arbitrarily close to 1, if we settle for a kernelization protocol with  $\mathcal{O}(1)$  calls to an oracle that solves instances of size bounded by a uniform polynomial in  $k$ . We extend the above results to general  $\mathcal{F}$ -DELETION, whenever  $\mathcal{F}$  contains a planar graph, as long as an oracle for TREEWIDTH- $\eta$ -DELETION is available for small instances. Notably, all our constants are computable functions of  $\mathcal{F}$  and our techniques work also when some graphs in  $\mathcal{F}$  may be disconnected.

Our results rely on two novel techniques. First, we transform so-called “near-protrusion decompositions” into true protrusion decompositions by sacrificing a small accuracy loss. Secondly, we show how to optimally compress such a decomposition with respect to general  $\mathcal{F}$ -DELETION. Using our second technique, we also obtain linear kernels on sparse graph classes when  $\mathcal{F}$  contains a planar graph, whereas the previously known theorems required all graphs in  $\mathcal{F}$  to be connected. Specifically, we generalize the kernelization algorithm by Kim, Langer, Paul, Reidl, Rossmanith, Sau & Sikdar [TALG 2015] on graph classes that exclude a topological minor.

**2012 ACM Subject Classification** Theory of computation → Graph algorithms analysis

**Keywords and phrases** kernelization, graph minors, treewidth, uniform kernels, minor hitting

**Digital Object Identifier** 10.4230/LIPIcs.STACS.2026.78

**Related Version** *Full Version*: <https://arxiv.org/abs/2601.08424>

**Funding** *Roohani Sharma*: Supported by the Young Scientist Fellowship of the Institute for Basic Science (IBS-R029-Y8).

*Michał Włodarczyk*: Supported by the Polish National Science Centre SONATA-19 grant 2023/51/D/ST6/00155.

**Acknowledgements** We thank the anonymous reviewer for pointing out a way to derive a finer bound for Lemma 26.



© Roohani Sharma and Michał Włodarczyk;

licensed under Creative Commons License CC-BY 4.0

43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026).

Editors: Meena Mahajan, Florin Manea, Annabelle McIver, and Nguyễn Kim Thăng

Article No. 78; pp. 78:1–78:21



Leibniz International Proceedings in Informatics

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



## 1 Introduction

Consider a finite family  $\mathcal{F}$  of graphs. In the  $\mathcal{F}$ -DELETION problem we are given a graph  $G$  and a positive integer  $k$  and we ask whether one can remove  $k$  vertices from  $G$  to obtain a graph that is  $\mathcal{F}$ -minor-free, that is, it does not contain any graph  $F \in \mathcal{F}$  as a minor. This captures a wide variety of graph problems such as VERTEX COVER, FEEDBACK VERTEX SET, DIAMOND HITTING SET, or  $d$ -PATH TRANSVERSAL. It is known that  $\mathcal{F}$ -DELETION is fixed-parameter tractable (FPT) [28] for every family  $\mathcal{F}$  and it admits a polynomial kernel<sup>1</sup> whenever  $\mathcal{F}$  contains at least one planar graph [11]. This condition is equivalent to the existence of a constant  $\eta(\mathcal{F}) \in \mathbb{N}$  such that every  $\mathcal{F}$ -minor-free graph has treewidth [5, §7] bounded by  $\eta(\mathcal{F})$  [26]. A canonical problem that fits into this category is TREEWIDTH- $\eta$ -DELETION: remove  $k$  vertices from a graph to reduce its treewidth to  $\eta$ . The arguably simplest case for which  $\mathcal{F}$  does not contain a planar graph is VERTEX PLANARIZATION; here the existence of a polynomial kernel remains a major open problem [18].

What may seem like a downside of the kernelization algorithm by Fomin et al. [11] is its non-uniformity: the size of the kernel is of the form  $\mathcal{O}_{\mathcal{F}}(k^{f(\mathcal{F})})$  for some function  $f$ . **This turns out to be inevitable:** TREEWIDTH- $\eta$ -DELETION **does not admit a kernel of size  $\mathcal{O}_{\eta}(k^{o(\eta)})$  unless  $\text{NP} \subseteq \text{coNP/poly}$**  [14]. On the positive side, uniform kernels are known for TREEDEPTH- $\eta$ -DELETION with  $\mathcal{O}_{\eta}(k^6)$  vertices [14] and for families  $\mathcal{F}$  of connected graphs that include  $K_{2,p}$  (for any  $p \in \mathbb{N}$ ) with  $\mathcal{O}_{\mathcal{F}}(k^{10})$  vertices [22] (cf. [10]).

Another way to obtain uniform kernelization is to restrict the input graph  $G$  to some well-behaving graph class. There are several meta-theorems that yield linear kernels for miscellaneous problems over the class of planar graphs [4], for every proper minor-closed graph class [12], and even for every graph class that excludes some graph as a topological minor [13, 19]. These approaches suffer from a different weakness though: they require the problem in question to satisfy a certain separation condition. **This translates to a restriction that every graph in the family  $\mathcal{F}$  must be connected.** Consequently, we could not handle deletion to graph classes such as “planar graphs of bounded face cover” or “bounded-treewidth graphs embeddable on a torus”. In the second example, observe that the family of minimal minor obstructions to torus embeddability includes  $K_5 + K_5$ .

**Lossy kernelization.** In order to overcome known barriers against polynomial kernels [3, 8], researchers started to study a lossy variant of kernelization [23]. Intuitively, an  $\alpha$ -approximate kernel reduces the task of finding an approximate solution to an instance of size  $\text{poly}(k)$  where  $k$  bounds the optimum. Here, the approximation factor  $\alpha$  measures how much is “lost in translation” between the two instances: a  $c$ -approximate solution is translated to an  $(\alpha \cdot c)$ -approximate solution. Such lossy kernels have been studied for VERTEX PLANARIZATION [18] and for  $\mathcal{F}$ -DELETION with a connectivity constraint on solution [7, 24]. To capture the difficulty of an approximation task by parameter  $k$ , solutions larger than  $k$  are treated as equally bad and we define their cost as  $k + 1$  [23].

Analogously to exact kernelization, one can allow the algorithm to repeatedly call an oracle that (approximately) solves an instance of bounded size. Assuming that the oracle processes instances of size  $\text{poly}(k)$ , such an algorithm is called an (approximate) polynomial Turing kernelization [16, 21]. A restricted model in which the number of oracle calls is bounded by  $\text{poly}(k)$  (so, in particular, it does not depend on the input size) has been dubbed

<sup>1</sup> A polynomial kernel for a parameterized problem  $L$  is a polynomial-time algorithm that given an instance  $(I, k)$  of  $L$  returns an equivalent one  $(I', k')$  such that  $|I'| + k' \leq \text{poly}(k)$ . See the book [8].

a *lossy kernelization protocol* [9]. In fact, allowing just two oracle calls unlocks technical tools unavailable for currently known one-shot lossy kernels [9]. Whereas there are problems that admit an exact polynomial Turing kernel and are unlikely to admit a regular polynomial kernel [17], the current lower bound techniques do not distinguish whether the oracle is called  $\text{poly}(k)$  times or just once [8, §21]. It is therefore an intriguing question if and how (lossy) kernelization protocols can outperform one-shot (lossy) kernels.

**Our results.** In this paper we address the two raised issues that trouble the existing kernelization algorithms for  $\mathcal{F}$ -DELETION. First, we show that a uniform polynomial kernelization for TREEWIDTH- $\eta$ -DELETION is possible if we settle for 2-approximation. This should be compared to polynomial-time  $\mathcal{O}(\log \eta)$ -approximation algorithm for TREEWIDTH- $\eta$ -DELETION [15] whereas  $c$ -approximation with an absolute constant  $c$  remains elusive. Our result extends to  $\mathcal{F}$ -DELETION for any family  $\mathcal{F}$  containing a planar graph, modulo the fact that we still need an oracle solving TREEWIDTH- $\eta$ -DELETION. As the oracle problem differs from the original one, we obtain a slightly weaker result, namely a *lossy polynomial compression*.

► **Theorem 1** (Uniform lossy kernel). *Let  $\mathcal{F}$  be a finite family of graphs containing at least one planar graph. Then  $\mathcal{F}$ -DELETION admits a 2-lossy compression with  $\mathcal{O}_{\mathcal{F}}(k^5)$  vertices and of size  $\mathcal{O}_{\mathcal{F}}(k^6)$ . Moreover, TREEWIDTH- $\eta$ -DELETION admits a 2-lossy kernel with  $\mathcal{O}_{\mathcal{F}}(k^5)$  vertices and of size  $\mathcal{O}_{\mathcal{F}}(k^6)$ .*

We can improve the approximation factor from 2 to  $(1 + \epsilon)$  for any  $\epsilon > 0$  by calling the oracle in several rounds. The number of rounds, as well as the degree in the kernel size, depend only on  $\epsilon$ . Hence we obtain a uniform lossy compression protocol with approximation factor arbitrary close to 1 and with constant number of rounds.

► **Theorem 2** (Uniform lossy protocol). *Let  $\mathcal{F}$  be a finite family of graphs containing at least one planar graph. For any  $\epsilon > 0$ ,  $\mathcal{F}$ -DELETION admits a  $(1 + \epsilon)$ -lossy compression protocol of call size  $\mathcal{O}_{\mathcal{F}}(k^6 + k^{3r+1})$ , and at most  $1 + r$  rounds, where  $r = 1 + \lceil \log_{1+\epsilon}(\frac{1}{\epsilon}) \rceil$ .*

*For the special case of TREEWIDTH- $\eta$ -DELETION, there is a  $(1 + \epsilon)$ -lossy kernelization protocol with same call size and number of rounds.*

The proofs of Theorems 1 and 2 rely on “protrusion techniques” [8, §16]. An  $r$ -*protrusion* is a vertex subset  $X \subseteq V(G)$  such that (i) treewidth of  $G[X]$  is at most  $r$  and (ii)  $X$  contains at most  $r$  vertices with a neighbor outside  $X$ . A graph  $G$  admits an  $(\alpha, \beta)$ -*protrusion decomposition* if there is a vertex subset  $P_0 \subseteq V(G)$  of size  $\leq \alpha$  so that  $G - P_0$  can be covered by  $\alpha$  many  $\beta$ -protrusions (see Section 2 for formal definitions). In the proofs we transform so-called *near-protrusion decompositions* [11] into true protrusion decompositions with bounded parameters  $\alpha, \beta$  by sacrificing small accuracy loss. The known techniques can compress such structures in a uniform fashion as long as all graphs in the family  $\mathcal{F}$  are connected. To tackle the remaining cases, we prove the following lemma. Here,  $\text{opt}_{\mathcal{F}}(G)$  denotes the minimum size of a solution to  $\mathcal{F}$ -DELETION in  $G$ , i.e., an  $\mathcal{F}$ -deletion set.

► **Lemma 3** (Handling disconnected forbidden minors). *Let  $\mathcal{F}$  be a finite family of graphs. There is an algorithm that, given a graph  $G$  with an  $(\alpha, \beta)$ -protrusion decomposition and integer  $k$ , runs in time  $\mathcal{O}_{\mathcal{F}, \beta}(|V(G)|)$  and outputs a graph  $G'$  with  $\mathcal{O}_{\mathcal{F}, \beta}(\alpha + k)$  vertices such that  $\min(\text{opt}_{\mathcal{F}}(G), k + 1) = \min(\text{opt}_{\mathcal{F}}(G'), k + 1)$ .*

We remark that the factor hidden in the  $\mathcal{O}(\cdot)$ -notation is a computable function of  $(\mathcal{F}, \beta)$ . See Lemma 14 for a full statement tailored for lossy kernelization. As a corollary of Lemma 3 we obtain linear kernels on graph classes where one can compute an  $(\mathcal{O}(k), \mathcal{O}(1))$ -protrusion

decomposition. The most general cases where such a construction is known are classes with excluded topological minor [19]. Whereas the known linear kernels for  $\mathcal{F}$ -DELETION require all the graphs in  $\mathcal{F}$  to be connected [4, 12, 13, 19], we are able to drop this assumption.

► **Theorem 4** (Linear kernel on sparse graphs). *Let  $H$  be a graph and  $\mathcal{F}$  be a finite family of graphs containing at least one planar graph. Then  $\mathcal{F}$ -DELETION admits a linear kernel on  $H$ -topological-minor-free graphs.*

**Organization of the paper.** We begin with an informal exposition of our main technical ideas in Section 3. The most of the remaining space is devoted to Lemma 3 (Section 4) which is too technical to be covered in the overview. We prove Theorems 1 and 2 in Sections 5 and 6. Since Theorem 4 follows from Lemma 3 and previous results, it is given in the full version of the article. The full version also contains the proofs of lemmas marked with (★) as well as background on tree decompositions and lossy kernelization.

## 2 Preliminaries

For any object  $F$ , the notation  $\mathcal{O}_F(\cdot)$  hides factors depending on  $F$ . For  $p \in \mathbb{N}$  we denote  $[p] = [1, p] = \{1, \dots, p\}$ .

► **Definition 5.** *Let  $T$  be a rooted tree and  $S \subseteq V(T)$  be a set of vertices in  $T$ . We define the least common ancestor of (not necessarily distinct)  $u, v \in V(T)$ , denoted as  $\text{LCA}(u, v)$ , to be the deepest node  $x$  which is an ancestor of both  $u$  and  $v$ . The LCA closure of  $S$  is the set  $\overline{\text{LCA}}(S) = \{\text{LCA}(u, v) : u, v \in S\}$ .*

► **Lemma 6** ([8, Lem. 9.26, 9.27, 9.28]). *Let  $T$  be a rooted tree,  $S \subseteq V(T)$  be non-empty, and  $L = \overline{\text{LCA}}(S)$ . All of the following hold.*

1. *Each connected component  $C$  of  $T - L$  satisfies  $|N_T(C)| \leq 2$ .*
2.  *$|L| \leq 2 \cdot |S| - 1$ .*
3.  *$\overline{\text{LCA}}(L) = L$ .*

**Protrusions.** For any  $\beta \in \mathbb{N}$ , a  $\beta$ -protrusion in a graph  $G$  is a vertex set  $X \subseteq V(G)$  such that the treewidth of  $G[X]$  is at most  $\beta$  and  $|\partial_G(X)| \leq \beta$ , where  $\partial_G(X) = N_G(V(G) \setminus X)$ . When  $G$  is clear from the context, we drop the subscript.

► **Definition 7.** *For  $\alpha, \beta \in \mathbb{N}$ , an  $(\alpha, \beta)$ -protrusion decomposition of a graph  $G$  is a partition of the vertex set  $V(G) = (P_0, P_1, \dots, P_\ell)$  such that:*

1.  *$\max\{|P_0|, \ell\} \leq \alpha$ , and*
2. *for each  $i \in [1, \ell]$ ,  $N[P_i]$  is a  $\beta$ -protrusion, and*
3. *for each  $i \in [1, \ell]$ ,  $N(P_i) \subseteq P_0$ .*

*We refer to  $P_0$  as the root bag of the decomposition. A protrusion decomposition is called nice if for each  $i \in [1, \ell]$  it holds that  $|N(P_i)| \leq \beta$ .*

► **Lemma 8.** *An  $(\alpha, \beta)$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of  $G$  can be transformed in linear time into a nice  $(\alpha, \beta)$ -protrusion decomposition  $(P'_0, P'_1, \dots, P'_\ell)$  such that for each  $i \in [1, \ell]$  it holds that  $P_i \subseteq P'_i$ .*

**Proof.** Fix  $i \in [1, \ell]$ . By definition, we have that  $|\partial(N[P_i])| = |N(V(G) \setminus N[P_i])| \leq \beta$ . Suppose that  $v \in N(P_i) \setminus \partial(N[P_i])$ . Then  $N(v) \subseteq N[P_i]$  and moving  $v$  from  $P_0$  to  $P_i$  does not alter the set  $N[P_i]$ , hence  $N[P_i] \cup \{v\}$  remains a  $\beta$ -protrusion. After performing such an operation exhaustively for each  $i \in [1, \ell]$  and  $v \in N(P_i) \setminus \partial(N[P_i])$ , we obtain a nice  $(\alpha, \beta)$ -protrusion decomposition. ◀

**Boundaried graphs.** A  $t$ -boundaried graph is a triple  $\mathbf{H} = (H, B, \lambda)$  where  $H$  is a graph,  $B \subseteq V(H)$  is of size  $t$  and  $\lambda: [t] \rightarrow B$  is a bijection representing a labeling function. We refer to the set  $B$  as  $\partial\mathbf{H}$  and call it the *boundary* of  $\mathbf{H}$ . By  $V(\mathbf{H})$  we refer to  $V(H)$ . Given two  $t$ -boundaried graphs  $\mathbf{H}_1 = (H_1, B_1, \lambda_1)$  and  $\mathbf{H}_2 = (H_2, B_2, \lambda_2)$ ,  $\mathbf{H}_1 \oplus \mathbf{H}_2$  denotes the graph  $H$  obtained by first taking the disjoint union of  $H_1$  and  $H_2$  and then identifying each vertex  $u_1$  in  $B_1$  with  $u_2 \in B_2$  such that  $\lambda_1^{-1}(u_1) = \lambda_2^{-1}(u_2)$ . For  $h \in \mathbb{N}$  and a graph  $G$ ,  $h$ -folio( $G$ ) is the set of all graphs on at most  $h$  vertices which appear in  $G$  as a minor.

► **Lemma 9 (★).** *There is a computable function  $\gamma: \mathbb{N} \times \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$  so that the following holds. Let  $h, r, d \in \mathbb{N}$  and  $\mathcal{F}$  be a family of graphs where each graph has at most  $h$  vertices. Next, for a graph  $G$  and an  $r$ -protrusion  $X$ , let us denote  $\mathbf{H} = (G[X], \partial_G(X), \lambda)$ , where  $\lambda$  is some labeling function, and  $\mathbf{H}^* = (G[N[V(G) \setminus X]], \partial_G(X), \lambda)$ . Note that  $G = \mathbf{H} \oplus \mathbf{H}^*$ .*

*Then there is an algorithm that, given  $G$  and  $X$ , runs in time  $\mathcal{O}_{h,r,d}(|X|)$  and returns a graph  $G'$  with the following properties.*

1.  $G'$  is obtained from  $G$  by replacing the boundaried graph  $\mathbf{H}$  with a  $|\partial_G(X)|$ -boundaried graph  $\widehat{\mathbf{H}}$ , that is  $G' = \widehat{\mathbf{H}} \oplus \mathbf{H}^*$  such that:
  - $h\text{-folio}(\mathbf{H} - \partial\mathbf{H}) = h\text{-folio}(\widehat{\mathbf{H}} - \partial\widehat{\mathbf{H}})$
  - $|V(\widehat{\mathbf{H}})| \leq \gamma(h, r, d)$ .
2. Given an  $\mathcal{F}$ -deletion set  $S'$  in  $G'$  such that  $|S' \cap V(\widehat{\mathbf{H}})| \leq d$ , one can in polynomial time find an  $\mathcal{F}$ -deletion set  $S$  in  $G$  of size at most  $|S'|$ . And vice versa, an  $\mathcal{F}$ -deletion set  $S$  in  $G$  such that  $|S \cap V(\mathbf{H})| \leq d$  can be lifted to an  $\mathcal{F}$ -deletion set  $S'$  of size at most  $|S|$  in  $G'$ .

*Additionally, suppose that  $Q$  is a fixed graph and  $G$  is  $Q$ -topological-minor free. Then the algorithm outputs a graph  $G'$  that is also  $Q$ -topological-minor free in time  $\mathcal{O}_{h,r,d,Q}(|X|)$ .*

The proof of Lemma 9 appears in the full version for completeness and follows the proof of [13, Theorem 1.1]. Note that the key highlight of Lemma 9 is that the function  $\gamma$  is computable and the family  $\mathcal{F}$  is arbitrary (and not necessarily of connected graphs).

### 3 Technical overview

We sketch the main ideas in Theorems 1 and 2 for the simplest case of TREEWIDTH- $\eta$ -DELETION. The starting point is a near-protrusion decomposition from [11]: one can assume that graph  $G$  is equipped with disjoint sets  $X, Z$  so that  $|X| = \mathcal{O}_\eta(k)$ ,  $|Z| = \mathcal{O}_\eta(k^3)$ ,  $\text{tw}(G - X) \leq \eta$ , each component  $C$  of  $G - (X \cup Z)$  has  $\mathcal{O}_\eta(1)$  neighbors in  $Z$  and each  $u, v \in N(C) \cap X$  are highly connected. The last condition implies that for any solution  $S$  of size  $\leq k$ , if  $\{u, v\} \cap S = \emptyset$  then  $u, v$  must share a bag in any tree decomposition of  $G - S$  of width  $\leq \eta$ . Therefore, it is safe to insert the edge  $uv$  to  $G$ . We can thus assume that  $N(C) \cap X$  is a clique. If this clique has size  $\geq 2(\eta + 1)$  then we know that any solution must remove at least half of its vertices. And so we can remove the entire clique by paying 2 in the approximation factor. After applying such a reduction rule, each component  $C$  of  $G - (X \cup Z)$  has  $\mathcal{O}_\eta(1)$  neighbors in total. This implies that  $N[C]$  forms an  $\mathcal{O}_\eta(1)$ -protrusion, having both its treewidth and boundary bounded. The framework of protrusion replacement [4, 13] allows us to replace  $G[N[C]]$  by a subgraph of constant size that exhibits the same behavior with respect to the problem in question. The caveat is that the number of such protrusions may be as large as  $k^{\Omega(\eta)}$  which is prohibitive for constructing a uniform kernel.

**Uniform lossy kernel.** We would like to transform the current decomposition into a  $(\mathcal{O}_\eta(k^c), \mathcal{O}_\eta(1))$ -protrusion decomposition, where  $c$  does not depend on  $\eta$ . We call a component  $C$  of  $G - (X \cup Z)$  *simplicial* if  $N(C)$  forms a clique. Because we can insert edges between highly connected vertices in  $X \cup Z$ , each non-edge may appear only in a bounded number

of sets  $N(C)$ . We use this argument to bound the number of non-simplicial components by  $\mathcal{O}_\eta(k^5)$ . Let us neglect the simplicial components for a moment and let  $G'$  denote the graph without them. Then  $G'$  admits a protrusion decomposition of desired form and we can compress  $G'$  to  $G''$  on  $\mathcal{O}_\eta(k^5)$  vertices. This is the output of the kernelization algorithm.

It remains to provide a mechanism to lift a solution  $S''$  from  $G''$  to  $G$ . First, we note that a solution  $S''$  can be lifted to a solution  $S'$  of  $G'$  using standard tools. The interesting step is lifting  $S'$  to a solution  $S$  in  $G$ . This is the part where the lifting step of the algorithm does a nontrivial (yet, polynomial-time) computation exploiting a treewidth bound. Consider a tree decomposition  $\mathcal{T}$  of  $G' - S'$  of width  $\eta$  and a simplicial component  $C$  of  $G - (X \cup Z)$ . Since  $N(C) \setminus S'$  is a clique, it must be covered by some bag in  $\mathcal{T}$ . Therefore, we can plug  $G[N[C]]$  into  $\mathcal{T}$  by merging a tree decomposition of  $G[N[C]]$  of width  $2\eta + 1$  and the tree decomposition  $\mathcal{T}$  alongside  $N(C) \setminus S'$ . This argument bounds the treewidth of  $G - S'$  by  $2\eta + 1$  and on such a graph we can solve TREEWIDTH- $\eta$ -DELETION exactly in polynomial time. We output the union of  $S'$  and the optimal solution in  $G - S'$ , resulting in 2-approximation.

**Uniform lossy protocol.** Improving the approximation factor below 2 requires a different approach. We can still start with the sets  $X \cup Z$  as described in the beginning of this section, and make each connected component  $C$  of  $G - (X \cup Z)$  a protrusion, this time by removing a clique in the neighbourhood of  $C$  whose size is at least  $(\eta + 1)(1 + \epsilon)/\epsilon$ . The main challenge is to bound the number of simplicial protrusions. Let  $Y_0 = X \cup Z$ . Since we now aim at a lossy kernelization protocol, we can send  $G[Y_0]$  to the oracle which outputs  $Y_1 \subseteq Y_0$  for which  $\text{tw}(G[Y_0 \setminus Y_1]) \leq \eta$ . Assume for simplicity that the oracle always returns an optimal solution. Then  $|Y_1| = \text{opt}(G[Y_0]) \leq \text{opt}(G)$ . And once again, we ask the oracle for a solution  $Y_2$  in  $G[Y_1]$ .

Suppose first that  $|Y_2| > (1 - \epsilon) \cdot |Y_1|$ . Observe that any optimal solution  $S$  in  $G$  satisfies  $|S \cap Y_1| \geq \text{opt}(G[Y_1])$  so  $\text{opt}(G - Y_1) \leq \text{opt}(G) - (1 - \epsilon) \cdot |Y_1|$ . A crucial observation is that removing  $Y_1$  from  $G$  allows us to construct an  $(\mathcal{O}_\eta(k^5), \mathcal{O}_\eta(1))$ -protrusion decomposition. For each simplicial component  $C$ , the set  $N(C) \setminus Y_1$  forms a clique in the graph  $G[Y_0 \setminus Y_1]$  of treewidth  $\leq \eta$ . But the number of cliques in a bounded-treewidth graph is linear so we can group the simplicial components into  $\mathcal{O}_\eta(|Y_0 \setminus Y_1|)$  protrusions according to their neighborhoods in  $Y_0 \setminus Y_1$ . Together with  $\mathcal{O}_\eta(k^5)$  non-simplicial components, this allows us to compress the graph  $G - Y_1$  into size  $\mathcal{O}_\eta(k^5)$ . Let  $S_1$  be the solution in  $G - Y_1$  computed with the help of the oracle. Assuming  $|S_1| = \text{opt}(G - Y_1)$  we get  $|S_1| + |Y_1| \leq \text{opt}(G) - (1 - \epsilon) \cdot |Y_1| + |Y_1| = \text{opt}(G) + \epsilon \cdot |Y_1| \leq (1 + \epsilon) \cdot \text{opt}(G)$ .

Suppose now that  $|Y_2| \leq (1 - \epsilon) \cdot |Y_1|$ . We continue asking the oracle for solution  $Y_{i+1}$  in  $G[Y_i]$ . Let  $j$  be the first iteration for which  $|Y_{j+1}| > (1 - \epsilon) \cdot |Y_j|$ . We repeat the same trick but this time the set  $Y_0 \setminus Y_j$  is partitioned into  $j$  subsets  $(Y_0 \setminus Y_1), (Y_1 \setminus Y_2), \dots, (Y_{j-1} \setminus Y_j)$ , each inducing a graph of treewidth  $\leq \eta$ . The number of cliques in  $G[Y_0 \setminus Y_j]$  can be bounded by  $f(\eta, j) \cdot |Y_0|^j$ , allowing us to construct an  $(\mathcal{O}_{\eta,j}(k^{\mathcal{O}(j)+5}), \mathcal{O}_{\eta,j}(1))$ -protrusion decomposition.

This procedure yields a uniform kernelization protocol as long as  $j$  can be bounded. But suppose we keep getting outcome  $|Y_{j+1}| \leq (1 - \epsilon) \cdot |Y_j|$ . Then for  $r = 1 + \lceil \log_{1-\epsilon}(\epsilon) \rceil$  we obtain  $|Y_r| \leq (1 - \epsilon)^{r-1} \cdot |Y_1| = \epsilon \cdot |Y_1| \leq \epsilon \cdot \text{opt}(G)$ . In this case we can again use the oracle to compute a solution in  $G - Y_r$ , bounding the number of protrusions by  $|Y_0|^r$ , and merge it with  $Y_r$ , achieving  $(1 + \epsilon)$ -approximation. See Figure 3 on page 17 for an illustration.

In order to handle  $\mathcal{F}$ -DELETION, we need to be more careful because inserting an edge between highly connected vertices may no longer be safe. To this end, we introduce an auxiliary graph which collects these inserted edges and exploit the fact that the optimum to  $\mathcal{F}$ -DELETION can be lower bounded by the optimum to TREEWIDTH- $\eta$ -DELETION, for some  $\eta = \eta(\mathcal{F})$ . We need to ask the oracle for solutions to TREEWIDTH- $\eta$ -DELETION over the

auxiliary graph as well, and this is the reason why in general we end up with compression protocol, rather than kernelization protocol. Whereas the standard protrusion replacement technique suffices to finish the construction when all the graphs in  $\mathcal{F}$  are connected, dealing with the general case requires one more tool.

**Handling disconnected forbidden minors.** There are two known ways to compress protrusions in the presence of disconnected graphs in  $\mathcal{F}$ . First, one can consider an annotated version of  $\mathcal{F}$ -DELETION, where some vertices are marked as undeletable [4], but this requires the oracle to solve a significantly harder task. Secondly, one can take advantage of a certain *well-quasi-order* over boundaried graphs [11] but this involves a non-constructive argument and results in factors that a priori may be an uncomputable function of  $\mathcal{F}$ . Besides, both approaches are insufficient to construct a linear kernel in Theorem 4.

To visualize the issue, consider  $F = K_4 + C_5$  and an  $(\alpha, \beta)$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of a graph  $G$ . It may be the case that  $G[P_i]$  is  $K_4$ -free for some  $i \in [1, \ell]$  but it contains a lot of minor models of  $C_5$ . Depending on whether a solution to  $F$ -DELETION hits all the copies of  $K_4$  outside  $P$ , it may need to remove either none or a very large number of vertices in  $P$ . In fact, such a problem does not admit *finite integer index* [19], a property crucial for protrusion replacement [4, 13].

Observe however that if  $G[P_i]$  contains a disjoint packing of  $k + 1$  minor models of  $C_5$ , then we already know that no solution of size  $\leq k$  can make the graph  $C_5$ -minor-free, so it should “focus” on hitting the  $K_4$ -minors. There is a caveat though: it might be the case that the only minor model of  $K_4$  in  $G$  intersects each model of  $C_5$  and so  $K_4 + C_5$  already does not appear as a minor. In order to circumvent such error-prone corner cases, we refine an  $(\alpha, \beta)$ -protrusion decomposition to one with *dichotomy property*. It states that for every connected graph  $F$  on at most  $h$  vertices (where  $h$  is the maximum graph size in  $\mathcal{F}$ ) either  $F$  does not appear as a minor in  $G - P_0$  or  $F$  has a large “private” collection of protrusions in which it appears. This property implies that any optimal solution  $S$  can use only  $\mathcal{O}_{\mathcal{F}, \beta}(1)$  vertices from each protrusion, mimicking the missing separation property. Intuitively, the aforementioned collections justify that any solution  $S$  of size  $\leq k$  must focus on hitting minors that do not appear in  $G - P_0$ , and so it does not make sense to remove too many vertices from a single protrusion. Consequently, the constructive protrusion replacement by Garnero et al. [13] can be adapted to compress protrusions in a decomposition with the dichotomy property.

To construct a decomposition with dichotomy property we generalize the packing/covering duality for connected minor models in bounded-treewidth graphs [25]. In our case however we do not pack models of a single graph  $F$  but we consider a family of connected graphs  $\mathcal{F}'$  and in each step of a greedy procedure we need to choose one graph  $F \in \mathcal{F}'$  to be packed, without spoiling the invariants for the remaining graphs from  $\mathcal{F}'$ . The details are technical and we omit them here.

## 4 Protrusion replacement for disconnected forbidden minors

In this section we collect ingredients to prove Lemma 14 which is a generalization of Lemma 3. We begin with two crucial definitions. We write  $2^{[1, \ell]}$  to denote the power set of  $[1, \ell]$ .

► **Definition 10** (Minor packing). *Let  $c \in \mathbb{N}$  and  $\mathcal{F}$  be a finite family of graphs. We say that a protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of  $G$  admits an  $(\mathcal{F}, c)$ -minor packing if there is a mapping  $I: \mathcal{F} \rightarrow 2^{[1, \ell]}$  such that:*

1. for each  $F \in \mathcal{F}$ ,  $|I(F)| = c$ ,
2. for each  $F \in \mathcal{F}$ ,  $i \in I(F)$ , it holds that  $F \leq_m G[P_i]$ ,
3. for each distinct  $F_1, F_2 \in \mathcal{F}$ , the sets  $I(F_1), I(F_2)$  are disjoint.

► **Definition 11** (Dichotomy property). *Let  $k, h \in \mathbb{N}$  and  $\mathcal{H}_h^{\text{conn}}$  denote the family of all connected graphs on at most  $h$  vertices. We say that a protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of graph  $G$  has  $(k, h)$ -dichotomy property if it admits an  $(\mathcal{F}, k + h)$ -minor packing, where  $\mathcal{F} \subseteq \mathcal{H}_h^{\text{conn}}$  is the family of connected graphs that belong to  $h$ -folio( $G - P_0$ ).*

We prove that any protrusion decomposition can be modified to satisfy  $(k, h)$ -dichotomy property. After initializing  $\mathcal{F} = \mathcal{H}_h^{\text{conn}}$ , we consider each graph from  $\mathcal{F}$  and as long as  $G - P_0$  contains an  $F$ -deletion set  $S_F$  of size  $\mathcal{O}_{h,\beta}(k)$  for some  $F \in \mathcal{F}$ , we insert  $S_F$  into  $P_0$  and remove  $F$  from  $\mathcal{F}$ . More precisely, we first compute the LCA-closure for  $S_F$  in a tree decomposition of  $G - P_0$  to preserve the invariants of a protrusion decomposition. When we cannot proceed, each remaining graph  $F \in \mathcal{F}$  must have a large packing of disjoint minor models in  $G - P_0$ . Note that these models can intersect for different  $F$ . We give a greedy procedure that scans a tree decomposition of  $G - P_0$  bottom-up and marks bags that allow us to separate a new protrusion containing a model of  $F$  (see Figure 2 on page 12). We also care to keep the set of marked bags closed under LCA, so that we end up with a protrusion decomposition of  $G - P_0$  with an  $(\mathcal{F}, k + h)$ -minor packing. Finally, we merge it with the protrusion decomposition of  $G$ .

► **Lemma 12.** *Suppose  $G$  admits an  $(\alpha, \beta)$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$ . Then it also admits a nice  $(\alpha + \mathcal{O}_{h,\beta}(k), \mathcal{O}_{h,\beta}(1))$ -protrusion decomposition with  $(k, h)$ -dichotomy property. Furthermore, this new decomposition, together with the corresponding minor packing, can be constructed in time  $\mathcal{O}_{h,\beta}(|V(G)|)$  given the old one.*

The proof of Lemma 12 is technical and we defer it to Section 4.1 to keep the focus of the current section. We now explain how dichotomy property ensures that an optimal  $\mathcal{F}$ -deletion set can remove only a bounded number of vertices from each protrusion.

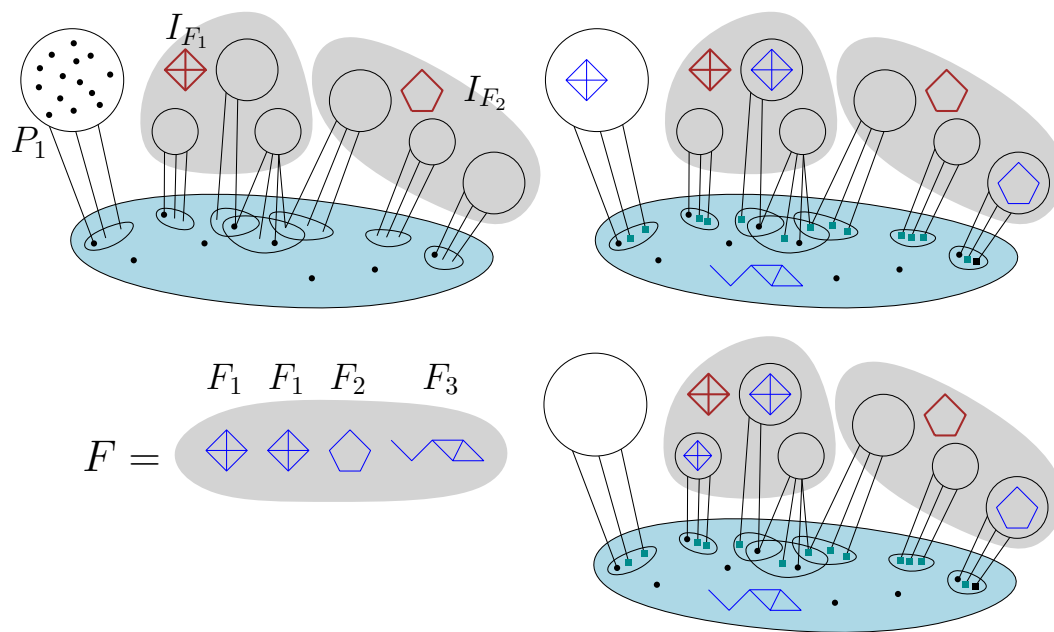
► **Lemma 13.** *Suppose  $G$  admits a nice  $(\alpha, \beta)$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  with  $(k, h)$ -dichotomy property and let  $\mathcal{F}$  be a family of graphs of maximum size  $h$ . Next, suppose that  $G$  has an  $\mathcal{F}$ -deletion set  $S$  of size  $\leq k$ . Then  $G$  has an  $\mathcal{F}$ -deletion set  $\widehat{S}$  with the property that for each  $i \in [1, \ell]$  we have  $|\widehat{S} \cap P_i| \leq \beta \cdot (h \cdot |\mathcal{H}_h^{\text{conn}}| + 1)$  and  $|\widehat{S}| \leq |S|$ .*

*Furthermore, one can compute  $\widehat{S}$ , given  $G, S$  and  $(P_0, P_1, \dots, P_\ell)$ , in polynomial time. Moreover, every minimum  $\mathcal{F}$ -deletion set in  $G$  has the described property.*

**Proof.** Let  $\mathcal{H}_G \subseteq \mathcal{H}_h^{\text{conn}}$  be the family of those connected graphs on at most  $h$  vertices that appear as minors in  $G - P_0$ . Due to  $(k, h)$ -dichotomy property, for each  $H \in \mathcal{H}_G$  there is a collection  $I(H) \subseteq [1, \ell]$  of size  $k + h$ , such that for  $i \in I(H)$  we have  $H \leq_m G[P_i]$  and the sets  $I(H)$  are disjoint for distinct graphs  $H$  from  $\mathcal{H}_G$ . Since  $|S| \leq k$ , for some  $h$  indices  $i \in I(H)$  the set  $P_i$  is disjoint from  $S$ . Let us denote this subset as  $J_H \subseteq I(H)$ . Consequently, for each  $H \in \mathcal{H}_G$  we have  $|J_H| = h$ , the sets  $J_H$  are pairwise disjoint, and for each  $i \in J_H$  it holds that  $H \leq_m G[P_i]$  and  $P_i \cap S = \emptyset$ . Note that  $\sum_{H \in \mathcal{H}_G} |J_H| \leq h \cdot |\mathcal{H}_h^{\text{conn}}|$ .

Suppose that there exists  $j \in [1, \ell]$  for which  $|S \cap P_j| > \beta \cdot (h \cdot |\mathcal{H}_h^{\text{conn}}| + 1)$ . W.l.o.g. we will assume that  $j = 1$ . Let  $\widehat{S}$  be obtained from  $S$  by (1) removing  $S \cap P_1$ , (2) inserting  $N(P_1)$ , (3) inserting  $N(P_i)$  for every  $i \in J_H, H \in \mathcal{H}_G$  (see Figure 1). Note that  $|\widehat{S}| < |S|$  because the given protrusion decomposition is nice.

We will now prove that  $\widehat{S}$  is also an  $\mathcal{F}$ -deletion set. Suppose not, and let  $F \in \mathcal{F}$  be a graph appearing in  $G - \widehat{S}$  as a minor. Let  $F_1, \dots, F_c, c \leq h$ , be the connected components of  $F$ . Note that some of them may be isomorphic. Consider a minor model  $\Phi$  of  $F$  in  $G - \widehat{S}$  for which the number of  $F_j$  appearing in  $P_1$  is minimized. More precisely, we minimize the number  $c'$  of indices  $j \in [c]$  for which  $\Phi(V(F_j)) \subseteq P_1$ . Observe that  $P_1$  is a union of connected components of  $G - \widehat{S}$  so if  $\Phi(V(F_j))$  intersects  $P_1$  then it must be fully contained in  $P_1$ . If  $c' = 0$  then  $F$  is a minor of  $G - (P_1 \cup \widehat{S})$ . But this is an induced subgraph of  $G - S$ , which is  $\mathcal{F}$ -minor-free by assumption, so this is impossible.



■ **Figure 1** Illustration to Lemma 13 for  $F$  being a disjoint union  $F_1 + F_1 + F_2 + F_3$  and  $\mathcal{F} = \{F\}$ . Top left: An  $\mathcal{F}$ -deletion set  $S$  is marked with black discs. The gray areas correspond to those  $P_i$  which belong to  $I(F_1), I(F_2)$ . Their neighborhoods in  $P_0$  are sketched as well. Each  $P_i$  in the corresponding family contains a minor model of the graph drawn in brown and is disjoint from  $S$ . Top right:  $\widehat{S}$  is obtained from  $S$  by removing  $S \cap P_1$  and adding the vertices marked with squares. A potential minor model  $\Phi$  of  $F$  in  $G - \widehat{S}$  is drawn in blue. It places one copy of  $F_1$  inside  $P_1$ . Bottom right: A new model  $\Phi'$  of  $F$  is obtained from  $\Phi$  by moving the image of  $F_1$  to an unused component in  $I(F_1)$ . But this model is also present in  $G - S$  which leads to a contradiction.

Consequently, the minor model  $\Phi$  places  $c' \geq 1$  connected components of  $F$  inside  $P_1$ . Assume w.l.o.g. that this is the case for  $F_1$ . Aiming at contradiction, we want to modify  $\Phi$  to place  $F_1$  outside  $P_1$ , without affecting the models of remaining  $F_j, j > 1$ . Since  $F_1 \leq_m G[P_1]$  it follows that  $F_1 \in \mathcal{H}_G$ . Recall that for each  $i \in J_{F_1}$  it holds that  $\widehat{S} \cap P_i = \emptyset$  (because  $S \cap P_i = \emptyset$ ) and  $N(P_i) \subseteq \widehat{S}$ . Since each  $F_j$  is connected,  $\Phi(F_j)$  can intersect at most one  $P_i$  for  $i \in J_{F_1}$ . Next,  $|J_{F_1}| = h$  and  $c \leq h$ , so there is some  $i \in J_{F_1}$  for which  $P_i$  is disjoint from all  $\Phi(F_j), j > 1$ . Therefore, we can move the model of  $F_1$  from  $P_1$  to  $P_i$ , obtaining a model  $\Phi'$  of  $F$  in  $G - \widehat{S}$  with  $c' - 1$  components contained in  $P_1$ . This contradicts the choice of the model  $\Phi$  and, consequently, contradicts the assumption that  $F \leq_m G - \widehat{S}$ . Hence  $\widehat{S}$  is a valid  $\mathcal{F}$ -deletion set.

We can repeat this process to reduce the intersection of  $\widehat{S}$  with each  $P_i$  because the described modification to  $S$  never adds new vertices from any  $P_i$ . A single refinement step can be implemented in polynomial time using any polynomial-time algorithm for minor testing, for example [27, 20], which also yields an algorithm to construct  $\widehat{S}$ . Observe that if  $G$  has an  $\mathcal{F}$ -deletion set of size at most  $k$ , then every minimum  $\mathcal{F}$ -deletion set  $S$  in  $G$  must satisfy  $|S \cap P_i| \leq \beta \cdot (h \cdot |\mathcal{H}_h^{\text{conn}}| + 1)$  as otherwise we could find a smaller  $\mathcal{F}$ -deletion set. ◀

We can now combine Lemma 13 with Lemma 9 to compress each  $P_i$  in a protrusion decomposition to a constant size. It is crucial that Lemma 9 preserves the  $h$ -folio of  $G[P_i]$  therefore the dichotomy property is maintained after the replacement. The last point of the lemma statement refers to the solution cost capped at  $k + 1$ , which will be convenient in the design of a lossy kernelization protocol.

► **Lemma 14.** *Let  $k, h \in \mathbb{N}$  and  $\mathcal{F}$  be a family of graphs of maximum size  $h$ . There is an algorithm that, given a graph  $G$  with an  $(\alpha, \beta)$ -protrusion decomposition, runs in time  $\mathcal{O}_{h,\beta}(|V(G)|)$  and returns a graph  $G'$  on  $\mathcal{O}_{h,\beta}(\alpha + k)$  vertices such that  $\min(\text{opt}_{\mathcal{F}}(G), k + 1) = \min(\text{opt}_{\mathcal{F}}(G'), k + 1)$ .*

*Furthermore, given an  $\mathcal{F}$ -deletion set  $S'$  in  $G'$ , one can in polynomial time lift it to an  $\mathcal{F}$ -deletion set  $S$  in  $G$ , satisfying*

$$\frac{\text{val}_{(G,k)}^{\mathcal{F}\text{-DEL}}(S)}{\text{opt}_{\mathcal{F}\text{-DEL}}((G,k))} \leq \frac{\text{val}_{(G',k)}^{\mathcal{F}\text{-DEL}}(S')}{\text{opt}_{\mathcal{F}\text{-DEL}}((G',k))}.$$

**Proof.** We apply Lemma 12 to construct a nice  $(\widehat{\alpha}, \widehat{\beta})$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  with the  $(k, h)$ -dichotomy property, where  $\widehat{\alpha} = \alpha + \mathcal{O}_{h,\beta}(k)$  and  $\widehat{\beta} = \mathcal{O}_{h,\beta}(1)$ . Let  $d := \widehat{\beta} \cdot (h \cdot |\mathcal{H}_h^{\text{conn}}| + 1)$ . We will iteratively replace each protrusion induced by  $N_G[P_i]$  with one of constant size using Lemma 9. Below we describe this process for  $P_1$ .

We fix some labeling  $\lambda$  of  $\partial_G(N_G[P_1])$  and replace the boundaried graph  $\mathbf{H} = (G[N[P_1]], N(P_1), \lambda)$  with a boundaried graph  $\widehat{\mathbf{H}}$  of size  $\mathcal{O}_{h,\widehat{\beta},d}(1) \in \mathcal{O}_{h,\beta}(1)$ . Denote the new graph  $G'$  and let  $P'_1$  be the set of vertices put in place of  $P_1$ .

► **Claim 15.** Suppose that  $\text{opt}_{\mathcal{F}}(G) \leq k$ . Then  $\text{opt}_{\mathcal{F}}(G') \leq \text{opt}_{\mathcal{F}}(G)$ .

**Proof.** By Lemma 13 there exists an optimal  $\mathcal{F}$ -deletion set  $S$  in  $G$  such that  $|S \cap P_1| \leq d$ . Lemma 9 guarantees that such an  $S$  can be translated to an  $\mathcal{F}$ -deletion set in  $G'$  of size at most  $|S|$ . ◀

► **Claim 16.** The protrusion decomposition  $(P_0, P'_1, P_2, \dots, P_\ell)$  of  $G'$  admits  $(k, h)$ -dichotomy property.

**Proof.** Lemma 9 ensures that  $h\text{-folio}(\mathbf{H} - \partial\mathbf{H}) = h\text{-folio}(\widehat{\mathbf{H}} - \partial\widehat{\mathbf{H}})$  which can be rewritten as  $h\text{-folio}(G[P_1]) = h\text{-folio}(G'[P'_1])$ . If  $P_1$  contributed a minor model to the minor packing corresponding to  $(k, h)$ -dichotomy property, it can be replaced by a minor model of the same graph in  $G'[P'_1]$ . ◀

► **Claim 17.** Suppose that  $\text{opt}_{\mathcal{F}}(G') \leq k$ . Then  $\text{opt}_{\mathcal{F}}(G) \leq \text{opt}_{\mathcal{F}}(G')$ . Furthermore, given an  $\mathcal{F}$ -deletion set  $S'$  in  $G'$  of size  $\leq k$ , one can in polynomial time lift it to an  $\mathcal{F}$ -deletion set  $S$  in  $G$  of size at most  $|S'|$ .

**Proof.** The previous claim allows us to apply Lemma 13 to graph  $G'$  and  $S'$ . Given  $S'$  we can in polynomial time find an  $\mathcal{F}$ -deletion set  $\widehat{S}$  in  $G'$  of size  $\leq |S'|$  for which  $|\widehat{S} \cap P'_1| \leq d$ . By applying Lemma 9 we can lift  $\widehat{S}$  to a solution in  $G$  of size at most  $|\widehat{S}| \leq |S'|$ . By taking  $S$  to be an optimal  $\mathcal{F}$ -deletion set in  $G$ , we infer that  $\text{opt}_{\mathcal{F}}(G) \leq \text{opt}_{\mathcal{F}}(G')$ . ◀

Claims 15 and 17 imply that  $\min(\text{opt}_{\mathcal{F}}(G), k + 1) = \min(\text{opt}_{\mathcal{F}}(G'), k + 1)$  and provide a mechanism to lift a bounded-size solution from  $G'$  to  $G$  while preserving its size. Claim 16 ensures that we preserve the  $(k, h)$ -dichotomy property in the new protrusion decomposition of  $G'$ , so we can repeat this process to replace each  $P_i$ . Application of Lemma 9 takes time linear in the size of the protrusion being replaced, so the total running time is linear.

We now justify the final inequality for lifting solutions. Recall that when  $S$  is a feasible  $\mathcal{F}$ -deletion set in  $G$  then  $\text{val}_{(G,k)}^{\mathcal{F}\text{-DEL}}(S) = \min\{|S|, k + 1\}$  and  $\text{opt}_{\mathcal{F}\text{-DEL}}((G, k))$  is the minimum over  $\text{val}_{(G,k)}^{\mathcal{F}\text{-DEL}}(S)$ . In these terms, we have  $\text{opt}_{\mathcal{F}\text{-DEL}}((G, k)) = \text{opt}_{\mathcal{F}\text{-DEL}}((G', k))$ . Suppose first that  $|S'| > k$  so  $\text{val}_{(G',k)}^{\mathcal{F}\text{-DEL}}(S') = k + 1$ . Then we simply return  $S = V(G)$  for which  $\text{val}_{(G,k)}^{\mathcal{F}\text{-DEL}}(S) = k + 1$  and the inequality holds. Otherwise,  $|S'| \leq k$ . We use the lifting algorithm described in Claim 17 to compute a solution  $S$  in  $G$  so that  $|S| \leq |S'|$  and we have

$$\frac{|S|}{\text{opt}_{\mathcal{F}\text{-DEL}}((G,k))} \leq \frac{|S'|}{\text{opt}_{\mathcal{F}\text{-DEL}}((G',k))}. \quad \blacktriangleleft$$

## 4.1 Proof of Lemma 12

We first summon the known packing-covering duality for bounded-treewidth graphs.

► **Proposition 18** ([25, Lemma 3.10]). *Let  $\beta, \ell \in \mathbb{N}$  and  $G, F$  be graphs, such that  $\text{tw}(G) \leq \beta$  and  $F$  is connected. Then either  $G$  contains a packing of  $\ell$  disjoint minor models of  $F$  or there is a set  $S \subseteq V(G)$  of size at most  $(\beta + 1) \cdot \ell$  such that  $G - S$  is  $F$ -minor-free.*

*Furthermore, for fixed  $F$  there is a linear algorithm that outputs one of these two objects, when given a corresponding tree decomposition of  $G$ .*

The following lemma utilizes the LCA-closure in a tree decomposition to cover a given vertex set  $S$  by the root bag of a protrusion decomposition.

► **Lemma 19** (★). *Let  $\beta \geq 1$ ,  $G$  be a graph of treewidth at most  $\beta$ , and  $S \subseteq V(G)$ . Then there exists a nice  $(2(\beta + 1) \cdot |S|, 2(\beta + 1))$ -protrusion decomposition  $(P_0, P_1, \dots, P_p)$  of  $G$  with  $S \subseteq P_0$ . Furthermore,  $(P_0, P_1, \dots, P_p)$  can be computed in linear time when a corresponding tree decomposition of  $G$  is given.*

Next, we will need a mechanism to refine a protrusion decomposition by another one, while preserving an  $(\mathcal{F}, c)$ -minor packing.

► **Lemma 20** (★). *Let  $\mathcal{F}$  be a family of connected graphs,  $c \in \mathbb{N}$ ,  $(P_0, P_1, \dots, P_p)$  be a nice  $(\alpha_1, \beta_1)$ -protrusion decomposition of a graph  $G$ , and  $(Q_0, Q_1, \dots, Q_q)$  be a nice  $(\alpha_2, \beta_2)$ -protrusion decomposition of  $G - P_0$  with an  $(\mathcal{F}, c)$ -minor packing. Then  $G$  admits a nice  $(\alpha_1 + 2\alpha_2 \cdot \beta_2, \beta_1 + \beta_2)$ -protrusion decomposition with an  $(\mathcal{F}, c)$ -minor packing, whose root bag is  $P_0 \cup Q_0$ . Furthermore, the new decomposition and its minor packing can be computed in linear time.*

The following lemma shows that if we have a sufficiently large packing of  $F$ -minors in a bounded-treewidth graph, for every  $F \in \mathcal{F} \subseteq \mathcal{H}_h^{\text{conn}}$ , then we can choose a large subset from each packing, so that together all the picked minor models are disjoint.

► **Lemma 21**. *Let  $h \in \mathbb{N}$ ,  $\mathcal{F} \subseteq \mathcal{H}_h^{\text{conn}}$ ,  $G$  be a graph such that  $\text{tw}(G) \leq \beta$  and for each  $F \in \mathcal{F}$  there is a disjoint packing of  $6(\beta + 1) \cdot |\mathcal{H}_h^{\text{conn}}| \cdot (k + h)$  minor models of  $F$  in  $G$ . Then there exists a nice  $(\mathcal{O}_{h,\beta}(k), 2\beta + 2)$ -protrusion decomposition  $(Q_0, Q_1, \dots, Q_\ell)$  of  $G$  with an  $(\mathcal{F}, k + h)$ -minor packing. Furthermore,  $(Q_0, Q_1, \dots, Q_\ell)$  and the corresponding minor packing can be constructed in time  $\mathcal{O}_{h,\beta}(|V(G)|)$ .*

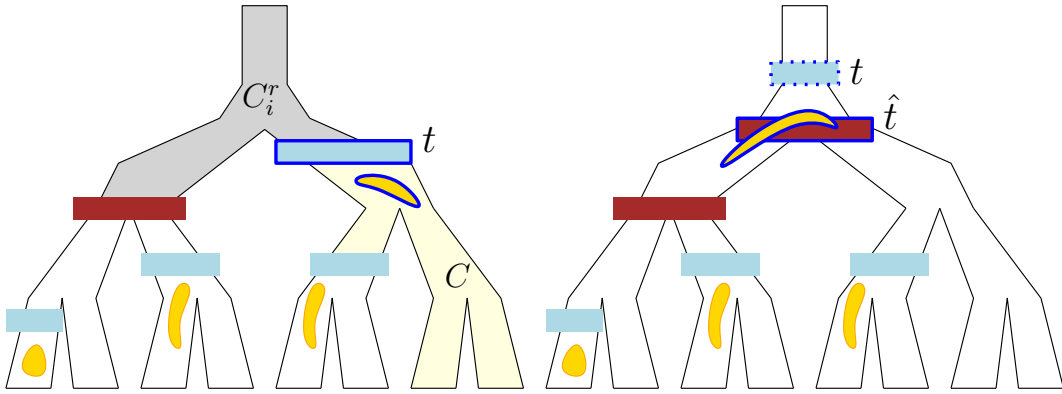
**Proof.** Consider a binary tree decomposition  $(T, \chi)$  of  $G$  of width  $\leq \beta$ , rooted at some node  $r$ . It can be constructed in linear time [2]. For a subset  $C \subseteq V(T)$  let  $\bar{\chi}(C) = \chi(C) \setminus \chi(V(T) \setminus C)$  be the set of vertices appearing only in the bags from  $C$ . We will describe a procedure that scans  $T$  bottom-up and marks two disjoint sets  $M, L \subseteq V(T)$ , both initialized as empty. Intuitively, a node will be marked (and added to  $M$ ) when it is a deepest node below which we can still find a minor model that can be added to the minor packing, and  $L$  will form the LCA closure of  $M$ . When we add a new node to  $M$  or  $L$ , we will create new connected components  $C$  of  $T - (M \cup L)$  not containing  $r$ . To some of these components we will assign  $F \in \mathcal{F}$  such that  $F \leq_m G[\bar{\chi}(C)]$ . We will refer to the number of components assigned to  $F$  as the *score* of  $F$ . We say that  $F$  is *completed* when its score reaches  $k + h$ . After the  $i$ -th step we shall maintain the following invariants.

1. The set  $M_i \cup L_i$  comprises  $i$  nodes and  $L_i$  is contained in  $\overline{\text{LCA}}(M_i)$ .
2. The sum of all scores equals  $|M_i|$ .
3. Let  $C_i^r$  denote the connected component of  $T - (M_i \cup L_i)$  that contains  $r$ . For every connected component  $C$  of  $T - (M_i \cup L_i)$  different from  $C_i^r$ , it holds that  $|N(C)| \leq 2$ .
4. For each  $F \in \mathcal{F}$  either  $F$  is already completed or  $G[\bar{\chi}(C_i^r)]$  contains a packing of  $3(\beta + 1) \cdot (2|\mathcal{H}_h^{\text{conn}}| \cdot (k + h) - i)$  minor models of  $F$ .

Clearly, the invariants hold for  $i = 0$ , before the first iteration. We will never increase a score of  $F$  that has been already completed so the algorithm terminates when  $|M_i| = |\mathcal{F}| \cdot (k + h)$ . As  $|M_i \cup L_i| \leq |\overline{\text{LCA}}(M_i)| \leq 2|M_i|$  (Lemma 6), it follows that the algorithm will perform at most  $2|\mathcal{F}| \cdot (k + h) \leq 2|\mathcal{H}_h^{\text{conn}}| \cdot (k + h)$  iterations.

Consider the  $i$ -th iteration,  $1 \leq i < 2|\mathcal{F}| \cdot (k + h)$ , and suppose there are still some not completed graphs in  $\mathcal{F}$ . Consider  $t \in V(T)$  that is not equal or descendant of any  $t' \in M_i \cup L_i$ . For such  $t$  let  $C_i^t \subseteq V(T)$  denote the set of those nodes which are descendants of  $t$  but not equal to or a descendant of any  $t' \in M_i \cup L_i$ . We look for the deepest node  $t \in V(T)$  for which  $G[\bar{\chi}(C_i^t)]$  contains a minor model of some  $F \in \mathcal{F}$  that is not yet completed. By invariant (4), such a node  $t$  must exist because for each not completed  $F$  there is a packing of at least  $3(\beta + 1)$  models of  $F$  in  $\bar{\chi}(C_i^t)$  whereas  $\chi(r)$  can intersect at most  $\beta + 1$  of them, so  $t = r$  satisfies the criteria.

We consider two scenarios. First suppose that there are some  $t_1, t_2 \in M_i$  for which  $\hat{t} = \text{LCA}(t_1, t_2)$  is descendant of  $t$  and  $\hat{t}$  does not belong to  $L_i \cup M_i$ . Choose such  $\hat{t}$  that is deepest and update  $L_{i+1} = L_i \cup \{\hat{t}\}$ ,  $M_{i+1} = M_i$ . In the second scenario, when there is no such  $\hat{t}$ , we set  $M_{i+1} = M_i \cup \{t\}$ ,  $L_{i+1} = L_i$ . By the choice of  $t$ , in the latter scenario we have created a new component  $C$  of  $T - (M_i \cup L_i)$ , not containing  $r$ , for which  $F \leq_m G[\bar{\chi}(C)]$ . Note that  $C$  can be chosen as a single component of  $T - (M_i \cup L_i)$  because  $F$  is connected. Then we assign  $F$  to  $C$ . See Figure 2 for an example.



■ **Figure 2** Illustration to Lemma 21. The bags corresponding to nodes from  $M_i$  are drawn in blue and those from  $L_i$  are drawn in brown. The already collected minor models, assigned to some components, are yellow. Left: A bag of  $t$  selected in the  $i$ -th iteration and a minor model of some  $F \in \mathcal{F}$  below  $t$  have blue borders. The first scenario does not trigger so  $t$  is added to  $M_i$ . This creates a component  $C$  (light yellow) of  $T - (M_i \cup L_i)$  to which we assign  $F$ . The component  $C_i^r$  has gray background. Right: In this situation there is  $\hat{t}$  below  $t$  that is LCA of two nodes from  $M_{i-1}$ . We do not add  $t$  to  $M_i$  but instead  $\hat{t}$  is inserted to  $L_i$ . We do not increase any score in this scenario.

We need to justify that all the invariants are preserved. The first two are clear.

To show invariant (3), we inspect the two scenarios. In first one, we have chosen the  $\hat{t} = \text{LCA}(t_1, t_2)$  for some  $t_1, t_2 \in M_i$  that does not yet belong to  $L_i \cup M_i$ . Suppose that one of the created components of  $T - (M_i \cup L_i)$  below  $\hat{t}$  has more than two neighbors. Then it has at least two neighbors different from  $\hat{t}$ . Let  $s_1, s_2 \in M_i \cup L_i$  denote these neighbors and let  $\hat{s} = \text{LCA}(s_1, s_2)$ . Note that  $\hat{s}$  must be a descendant of  $\hat{t}$ . By invariant (1), there are  $s'_1, s'_2 \in M_i$  such that  $\hat{s} = \text{LCA}(s'_1, s'_2)$  as well. This contradicts the choice of  $\hat{t}$  because  $\hat{s}$  satisfies the same condition and is located deeper. The same argument works in the second scenario. If some component of  $T - (M_i \cup L_i)$  created below  $t$  has more than two neighbors, then  $t$  must have a descendant  $\hat{s}$  with the same property, hence it is the first scenario that must have occurred.

To advocate invariant (4) we show that for every not-yet-completed  $F \in \mathcal{F}$  the number of disjoint minor models of  $F$  in  $\bar{\chi}(C_i^r)$  can decrease only moderately. Let  $\mathcal{M}^F$  denote the initial disjoint packing of  $6(\beta + 1) \cdot |\mathcal{H}_h^{\text{conn}}| \cdot (k + h)$  minor models of  $F$  in  $G$ . Let  $\mathcal{M}_i^F \subseteq \mathcal{M}^F$  denote the subfamily of those minor models which are entirely contained in  $\bar{\chi}(C_i^r)$ . We aim to show that  $|\mathcal{M}_{i+1}^F| \geq |\mathcal{M}_i^F| - 3(\beta + 1)$ ; this will imply invariant (4).

Let us fix  $F \in \mathcal{F}$  and let  $t \in V(T)$  be the node chosen in the  $i$ -th iteration, in any of the two scenarios. Recall that we work on a binary tree decomposition so  $t$  has either one or two children; w.l.o.g. assume there are two of them:  $t_1, t_2$ . Furthermore, let  $C_1, C_2$  be the connected components of  $C_i^r - \{t\}$  that contains  $t_1, t_2$ , respectively. Let  $U \subseteq V(G)$  induce a minor model from  $\mathcal{M}_i^F \setminus \mathcal{M}_{i+1}^F$ . Suppose that  $M \subseteq \bar{\chi}(C_1)$ . Because we have not chosen  $t_1$  in the  $i$ -th iteration,  $U$  must intersect  $\chi(t_1)$ . Similarly, if  $U \subseteq \bar{\chi}(C_2)$  then  $U$  must intersect  $\chi(t_2)$ . If neither of these cases hold,  $U$  must intersect  $\chi(t)$  because  $G[U]$  is connected. Since  $|\chi(t) \cup \chi(t_1) \cup \chi(t_2)| \leq 3(\beta + 1)$  and the considered minor models are disjoint, we infer that  $|\mathcal{M}_i^F \setminus \mathcal{M}_{i+1}^F| \leq 3(\beta + 1)$ . Hence invariant (4) is preserved.

Having completed all  $F \in \mathcal{F}$ , the process stops. Let  $j$  denote the number of the last iteration, let  $M = M_j, L = \overline{\text{LCA}}(M)$ . Invariant (2) implies that  $|M| = |\mathcal{F}| \cdot (k + h)$ . By invariant (3) we know that  $L \setminus L_j \subseteq C_j^r$ , that is, we do not add any new vertices to  $L$  that would intersect a component of  $T - (M_j \cup L_j)$  to which we have assigned some  $F \in \mathcal{F}$ . We define  $Q_0 = \bigcup_{t \in L} \chi(t)$ ; this set has at most  $2(\beta + 1) \cdot |M| \leq 2(\beta + 1) \cdot |\mathcal{H}_h^{\text{conn}}| \cdot (k + h)$  vertices. By Lemma 6 the neighborhood of each connected component of  $G - Q_0$  is contained in at most two bags of  $(T, \chi)$ , so it contains at most  $2(\beta + 1)$  vertices. For each  $F \in \mathcal{F}$  we gather the components of  $T - L$  assigned to  $F$ . For each such component  $C$  we add the set  $Q_C = \bar{\chi}(C)$  to the protrusion decomposition. This gives us the  $(\mathcal{F}, k + h)$ -minor packing. Using LCA-closure, the remaining components of  $T - L$  can be grouped into  $\leq 2|L|$  sets, each with neighborhood in at most two nodes. This concludes the proof.  $\blacktriangleleft$

We are ready to prove the Lemma 12 and thus complete the proof of Lemma 14.

**Proof of Lemma 12.** Let  $\hat{k} = 6(\beta + 1) \cdot |\mathcal{H}_h^{\text{conn}}| \cdot (k + h)$ . Due to Lemma 8 we can assume that  $(P_0, P_1, \dots, P_\ell)$  is nice, i.e., for  $i \in [1, \ell]$  we have  $|N_G(P_i)| \leq \beta$ . Initialize  $\mathcal{H}^1 = \mathcal{H}_h^{\text{conn}}$ ,  $\alpha_1 = \alpha$ ,  $\beta_1 = \beta$ , and  $\mathcal{P}^1 = (P_0, P_1, \dots, P_\ell)$ ,  $P^1 = P_0$ . In the  $i$ -th iteration we will remove one set from  $\mathcal{H}^i$  and construct a new protrusion decomposition  $\mathcal{P}^{i+1}$  with a root bag denoted as  $P^{i+1}$ . We will maintain an invariant that  $G - P^i$  is  $F$ -minor-free for each  $F \in \mathcal{H}_h^{\text{conn}} \setminus \mathcal{H}^i$ .

In the  $i$ -th iteration,  $i \geq 1$ , we apply Proposition 18 for every  $F \in \mathcal{H}^i$ , the graph  $G - P_Q^i$  and  $\ell = \hat{k}$ . Suppose that for some  $F \in \mathcal{H}^i$  this call returns a vertex set  $S_F$  of size  $\leq (\beta + 1) \cdot \hat{k}$  for which  $G - (P^i \cup S_F)$  is  $F$ -minor-free. Then we apply Lemma 19 to the graph  $G - P^i$  and set  $S_F$  to compute a nice  $(2(\beta + 1) \cdot \hat{k}, 2(\beta + 1))$ -protrusion decomposition of  $G - P^i$  whose root bag contains  $S_F$ . Next, we apply Lemma 20, disregarding the parameter  $\mathcal{F}$ , to transform it into a nice  $(\alpha_{i+1}, \beta_{i+1})$ -protrusion decomposition  $\mathcal{P}^{i+1}$  of  $G$  whose root bag  $P^{i+1}$  contains  $S_F$  and  $P^i$ . We can estimate  $\alpha_{i+1} = \alpha_i + 2(\beta + 1)^2 \cdot \hat{k}$  and  $\beta_{i+1} = \beta_i + 2(\beta + 1)$ . We set  $\mathcal{H}^{i+1} = \mathcal{H}^i \setminus \{F\}$  and the invariant is preserved.

Let  $j$  denote the iteration number at which this process stops. Let  $\mathcal{F} = \mathcal{H}^j$ . We have  $j \leq |\mathcal{H}_h^{\text{conn}}| + 1$  so  $\alpha_j = \alpha + \mathcal{O}_{h,\beta}(k)$  and  $\beta_j = \mathcal{O}_{h,\beta}(1)$ . If  $\mathcal{F} = \emptyset$  then we are done so let us assume otherwise.

Since the process has stopped, Proposition 18 guarantees that every  $F \in \mathcal{F}$  has a packing of  $\hat{k}$  disjoint minor models in  $G - P^j$ . Hence the requirements of Lemma 21 are satisfied with respect to the graph  $G - P^j$ . Applying the lemma gives us a nice  $(\mathcal{O}_{h,\beta}(k), 2\beta + 2)$ -protrusion decomposition  $\mathcal{Q}$  of  $G - P^j$  with an  $(\mathcal{F}, k + h)$ -minor packing. Finally, we apply Lemma 20 with respect to  $\mathcal{P}_j, \mathcal{Q}$ , and the constructed  $(\mathcal{F}, k + h)$ -minor packing for  $\mathcal{Q}$ . As a

result, we obtain a nice  $(\alpha + \mathcal{O}_{h,\beta}(k), \mathcal{O}_{h,\beta}(1))$ -protrusion decomposition  $\mathcal{R}$  of  $G$  with an  $(\mathcal{F}, k + h)$ -minor packing. Furthermore, the root bag of  $\mathcal{R}$  contains  $P^j$ . Therefore  $G - P^j$  is  $F$ -minor-free for each  $F \in \mathcal{H}_h^{\text{conn}} \setminus \mathcal{F}$  and so the above minor packing yields  $(k, h)$ -dichotomy property.  $\blacktriangleleft$

## 5 Uniform 2-lossy polynomial compression algorithm

In this section we prove Theorem 1. Several reduction rules from the section will also come in useful in Section 6. Throughout Sections 5 and 6, we assume that the family  $\mathcal{F}$  contains a planar graph. Fix an arbitrary  $F_{\text{planar}} \in \mathcal{F}$  which is a planar graph. The Grid Minor Theorem implies that for any  $\mathcal{F}$ -deletion set  $S$  of  $G$ , the treewidth of  $G - S$  is bounded in terms of  $|V(F_{\text{planar}})|$  [26]. We shall denote by  $\eta = \eta(\mathcal{F})$  the upper bound on the treewidth of an  $\mathcal{F}$ -minor-free graph.

Following the convention from [23], we consider solution cost capped at  $k + 1$ , i.e., we measure the cost of an  $\mathcal{F}$ -deletion set  $S$  in  $G$  as  $\text{val}_{(G,k)}(S) := \min\{|S|, k + 1\}$ . This ensures that the parameter  $k$  captures the difficulty of an instance  $(G, k)$ . We refer the reader to [23] and the full version of this article for more details on lossy kernelization and lossy reduction rules.

Our kernelization algorithm (and protocol) starts by constructing a *near-protrusion decomposition* of  $V(G)$  as done in [11, Lemma 25]. The construction in [11] is randomized because it relies on a randomized constant-factor approximation algorithm for  $\mathcal{F}$ -DELETION. This step, and hence the entire construction of [11, Lemma 25], can be made deterministic by using the deterministic constant-factor approximation for  $\mathcal{F}$ -DELETION from [15, Corollary 1.1].

► **Lemma 22** ([11, Lemma 25] together with [15, Corollary 1.1]). *There is a polynomial-time algorithm that given an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION, when  $\mathcal{F}$  contains a planar graph, either reports correctly that  $(G, k)$  is a no-instance of  $\mathcal{F}$ -DELETION, or computes  $X, Z \subseteq V(G)$ ,  $X \cap Z = \emptyset$  such that:*

- $|X| = \mathcal{O}_\eta(k)$ ,  $|Z| = \mathcal{O}_\eta(k^3)$ ,  $X$  is an  $\mathcal{F}$ -deletion set of  $G$ ,
- for each connected component  $C$  of  $G - (X \cup Z)$ ,  $|N_G(C) \cap Z| \leq 2(\eta + 1)$ , and
- for each connected component  $C$  of  $G - (X \cup Z)$ , and distinct  $u, v \in N_G(C) \cap X$ , there are at least  $k + \eta + 2$  vertex-disjoint paths from  $u$  to  $v$  in  $G$ .

Throughout the rest of this section,  $X, Z$  stand for the sets returned by Lemma 22 for input  $(G, k)$ . We now design a  $(1 + \epsilon)$ -lossy strict reduction rule that bounds the size of the neighborhood of each connected component of  $G - (X \cup Z)$ .

► **Lossy Reduction Rule 1.** *Let  $\epsilon > 0$  and let  $C$  be a connected component of  $G - (X \cup Z)$  such that  $|N_G(C) \cap X| \geq \frac{(1+\epsilon)(\eta+1)}{\epsilon}$ . In this case the reduction algorithm takes as input an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION and outputs the instance  $(G' := G - (N_G(C) \cap X), k' := k - |N_G(C) \cap X|)$  of  $\mathcal{F}$ -DELETION.*

*The solution lifting algorithm takes as input instances  $(G, k)$ ,  $(G', k')$  and a set  $S' \subseteq V(G')$  and outputs the set  $S \subseteq V(G)$ , where  $S := S' \cup (N_G(C) \cap X)$ .*

We prove that such a reduction rule is a  $(1 + \epsilon)$ -lossy strict reduction rule, that is,

$$\frac{\text{val}_{G,k}^{\mathcal{F}\text{-DEL}}(S)}{\text{opt}_{\mathcal{F}\text{-DEL}}(G,k)} \leq \max \left\{ (1 + \epsilon), \frac{\text{val}_{G',k'}^{\mathcal{F}\text{-DEL}}(S')}{\text{opt}_{\mathcal{F}\text{-DEL}}(G',k')} \right\}.$$

► **Lemma 23** (★). *Lossy Reduction Rule 1 is a  $(1 + \epsilon)$ -lossy strict reduction rule.*

After an application of Lossy Reduction Rule 1, some vertices from  $X$  may get deleted. But the remaining set  $X$ , together with  $Z$ , still satisfies the properties of Lemma 22 in the resulting graph. We keep the variable  $X$  to denote this set in the new graph.

► **Lemma 24 (★)**. *When Lossy Reduction Rule 1 is no longer applicable, then there is an  $(\alpha, \mathcal{O}(\eta/\epsilon))$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of  $G$  for some  $\alpha$ , where  $P_0 := X \cup Z$  and for each  $i \in [1, \ell]$ ,  $P_i$  is a connected component of  $G - (X \cup Z)$ . Furthermore,  $(P_0, P_1, \dots, P_\ell)$  can be computed in polynomial time.*

Let  $(P_0, \dots, P_\ell)$  be the protrusion decomposition obtained from Lemma 24. We now define the *augmented graph*  $G_{\text{flow}}$  of  $G$  which is a supergraph of  $G$  on the same vertex set as  $G$ , and which is obtained by additionally adding the edge set  $\{uv : u, v \in P_0, \text{ there exist at least } k + \eta + 2 \text{ internally vertex-disjoint paths from } u \text{ to } v \text{ in } G\}$  in  $G$ .

► **Lemma 25 (★)**. *The following two (in)equalities hold:*

1.  $\text{opt}_{\text{TW-}\eta\text{-DEL}}((G, k)) = \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}, k))$ ,
2.  $\text{opt}_{\mathcal{F}\text{-DEL}}((G, k)) \geq \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}, k))$ .

We partition  $\{P_1, \dots, P_\ell\}$  into two groups based on the neighborhood of  $P_i$  in  $G_{\text{flow}}$ . Let  $\mathcal{P}_{\text{simp}} = \{P_i : G_{\text{flow}}[N(P_i)] \text{ is a clique or } N(P_i) \text{ is empty}\}$ , which we call the set of *simplicial parts* of the protrusion decomposition. Let  $\mathcal{P}_{\text{non-simp}}$  be the set of remaining parts, referred to as *non-simplicial parts*. Note that we cannot simply discard  $P_i$  for which  $N(P_i) = \emptyset$  in the case of  $\mathcal{F}$ -DELETION when  $\mathcal{F}$  contains disconnected graphs. Observe that when  $u, v \in P_0$  and  $uv \notin E(G_{\text{flow}})$  then  $\{u, v\}$  can belong to at most  $k + \eta + 1$  sets  $N(P_i)$ .

► **Lemma 26 (★)**.  $|\mathcal{P}_{\text{non-simp}}| = \mathcal{O}_\eta(k^5)$ .

We show that ignoring the simplicial parts yields a 2-lossy compression reduction rule. To this end, we need an algorithm solving  $\mathcal{F}$ -DELETION exactly on bounded-treewidth graphs.

► **Proposition 27** ([1, Theorem 4]). *When  $\mathcal{F}$  is a finite family of graphs and contains at least one planar graph, then  $\mathcal{F}$ -DELETION can be solved in time  $2^{\mathcal{O}_{\mathcal{F}}(\text{tw} \log \text{tw})} \cdot n$  on an  $n$ -vertex graph of treewidth  $\text{tw}$ .*

► **Lossy Reduction Rule 2**. *Suppose we are given an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION with the corresponding protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$ . The reduction algorithm outputs  $(G' := G_{\text{flow}}[V(G) \setminus \bigcup_{P_i \in \mathcal{P}_{\text{simp}}} P_i], k' := k)$ .*

*The solution lifting algorithm takes a solution  $S'$  of the TREEWIDTH- $\eta$ -DELETION problem on the instance  $(G', k')$ . It computes an optimum-sized  $\mathcal{F}$ -deletion set  $S''$  of  $G - S'$  in  $\mathcal{O}_{\mathcal{F}, \eta}(n^{\mathcal{O}(1)})$  time using the algorithm of Proposition 27 (we argue in Lemma 28 that this is possible). It then outputs  $S := S' \cup S''$  as a solution on  $\mathcal{F}$ -DELETION for the instance  $(G, k)$ .*

Specifically, we prove that the simplicial parts can be added to  $G' - S'$  while keeping the treewidth bounded by  $\mathcal{O}(\eta)$ . Hence  $S''$  can be computed in polynomial time.

► **Lemma 28 (★)**. *Lossy Reduction Rule 2 is a 2-lossy compression reduction rule.*

After the exhaustive application of Lossy Reduction Rule 1 and 2, it only remains to bound the size of each part in  $\mathcal{P}_{\text{non-simp}}$ , which can be done using Lemma 14. The formal proof of Theorem 1 is given next.

► **Theorem 1** (Uniform lossy kernel). *Let  $\mathcal{F}$  be a finite family of graphs containing at least one planar graph. Then  $\mathcal{F}$ -DELETION admits a 2-lossy compression with  $\mathcal{O}_{\mathcal{F}}(k^5)$  vertices and of size  $\mathcal{O}_{\mathcal{F}}(k^6)$ . Moreover, TREEWIDTH- $\eta$ -DELETION admits a 2-lossy kernel with  $\mathcal{O}_{\mathcal{F}}(k^5)$  vertices and of size  $\mathcal{O}_{\mathcal{F}}(k^6)$ .*

**Proof.** We describe a polynomial time 2-lossy compression algorithm below. Given an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION, first compute a near-protrusion decomposition on the instance  $(G, k)$  in polynomial time using Lemma 22. This gives sets  $X, Z$  with the properties listed in Lemma 22. Set  $\epsilon = 1$  and for each connected component  $C$  of  $G - (X \cup Z)$ , it checks whether Lossy Reduction Rule 1 is applicable. Let  $(G^1, k_1)$  be the instance returned by the reduction algorithm of Lossy Reduction Rule 1 at the end of its exhaustive application. From Lemma 24,  $G^1$  has an  $(\alpha, \mathcal{O}(\eta))$ -protrusion decomposition  $(P_0, \dots, P_\ell)$  for some (unbounded)  $\alpha$  and  $|P_0| = \mathcal{O}_\eta(k^3)$ .

Let  $\mathcal{P}_{\text{non-simp}}$  be the set of non-simplicial parts of  $(P_1, \dots, P_\ell)$ . From Lemma 26,  $|\mathcal{P}_{\text{non-simp}}| = \mathcal{O}_\eta(k^5)$ . Run the reduction algorithm of Lossy Reduction Rule 2 on the instance  $(G^1_{\text{flow}}[V(G^1) \setminus \bigcup_{P_i \in \mathcal{P}_{\text{simp}}} P_i], k_1)$  of  $\mathcal{F}$ -DELETION. It returns an instance  $(G^2, k_1)$  of TREEWIDTH- $\eta$ -DELETION such that  $G^2$  has an  $(\mathcal{O}_\eta(k^5), \mathcal{O}(\eta))$ -protrusion decomposition. Let  $(G^3, k_1)$  be the instance returned by the algorithm of Lemma 14 on input  $(G^2, k_1)$ . Observe that  $k_1 \leq k$ . From Lemma 14 the number of vertices of  $G^3$  is bounded by  $\mathcal{O}_\mathcal{F}(k^5)$ . Observe that if  $G^3$  indeed has an at most  $k$ -sized vertex set whose deletion results in a graph of treewidth at most  $\eta$ , then treewidth of  $G^3$  has to be at most  $k + \eta$ . Thus, the number of edges of  $G^3$ , and hence the size of  $G^3$ , is at most  $(k + \eta)$  times the number of vertices of  $G^3$ . Thus, the size of  $G^3$  is  $\mathcal{O}_\mathcal{F}(k^6)$ , otherwise  $G^3$  has no  $k$ -sized solution.

For any  $\beta \geq 1$ , run the  $\beta$ -compression oracle for TREEWIDTH- $\eta$ -DELETION on the instance  $(G^3, k_1)$  and let  $S_3$  be the returned  $\beta$ -approximate solution of the instance  $(G^3, k_1)$  of TREEWIDTH- $\eta$ -DELETION. The algorithm of Lemma 14 provides a lifting algorithm that, in polynomial time, outputs a  $\beta$ -approximate solution  $S_2$  for the instance  $(G^2, k_1)$  of TREEWIDTH- $\eta$ -DELETION. Given the  $\beta$ -approximate solution  $S_2$  for the instance  $(G^2, k_1)$  of TREEWIDTH- $\eta$ -DELETION, from Lemma 28, the solution lifting algorithm of Lossy Reduction Rule 2, outputs a  $2 \cdot \beta$ -approximate solution  $S_1$  for the instance  $(G^1, k_1)$  of the  $\mathcal{F}$ -DELETION problem. Finally since Lossy Reduction Rule 1 is a  $(1 + \epsilon)$ -lossy strict reduction rule and  $\epsilon = 1$ , using the solution lifting algorithm of Lossy Reduction Rule 1 and  $S_1$ , one can obtain a 2-approximate solution for the instance  $(G, k)$  of  $\mathcal{F}$ -DELETION. Clearly, in the special case of TREEWIDTH- $\eta$ -DELETION the above constitutes a 2-lossy kernelization algorithm.  $\blacktriangleleft$

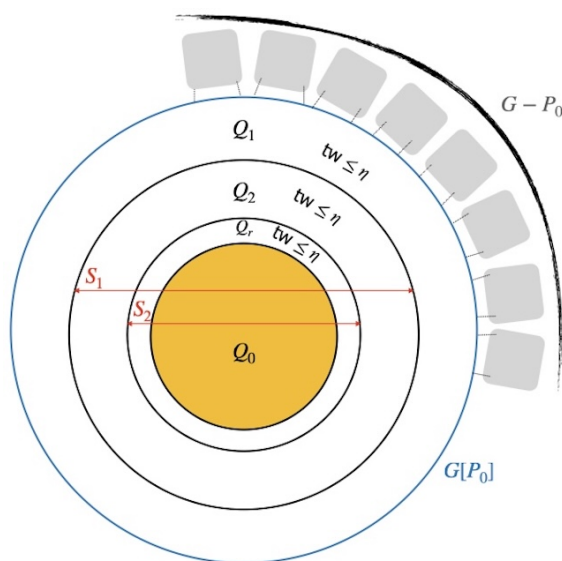
## 6 Uniform $(1 + \epsilon)$ -lossy polynomial compression protocol

This section is devoted to the proof of Theorem 2. Assume that we are given an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION,  $\epsilon > 0$ , and an  $(\alpha, \mathcal{O}(\eta/\epsilon))$ -protrusion decomposition  $(P_0, P_1, \dots, P_\ell)$  of  $G$  from Lemma 24, for some unbounded  $\alpha$ , with  $|P_0| = \mathcal{O}_\eta(k^3)$ . Recall that the parts  $\{P_1, \dots, P_\ell\}$  of this protrusion decomposition were partitioned into two sets:  $\mathcal{P}_{\text{simp}}$  (simplicial parts) and  $\mathcal{P}_{\text{non-simp}}$  (non-simplicial parts). Lemma 26 estimates  $|\mathcal{P}_{\text{non-simp}}|$  as  $\mathcal{O}_\eta(k^5)$ .

► **Lemma 29.** *Consider  $\epsilon > 0$  and  $\beta \in \mathbb{N}$ . Let  $r = 1 + \lceil \log_{1+\epsilon}(\frac{1}{\epsilon}) \rceil$ . There exists a polynomial-time protocol which has access to a  $\beta$ -approximate kernelization oracle  $\mathcal{O}$  for TREEWIDTH- $\eta$ -DELETION of capacity  $|P_0|$ , and when given  $(G_{\text{flow}}[P_0], k)$  as input, it makes at most  $r$  calls to the oracle  $\mathcal{O}$ , and outputs a partition of  $P_0 = (Q_0, \dots, Q_r)$ , such that*

1. for each  $i \in [1, r]$ , treewidth of  $G_{\text{flow}}[Q_i]$  is at most  $\eta$  ( $Q_i$  is allowed to be empty), and
2. a.  $|Q_0| \leq (1 + \epsilon) \cdot \beta \cdot \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}[Q_0], k))$  or  
b.  $|Q_0| \leq \epsilon \cdot \beta \cdot \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}[P_0], k))$ .

**Proof.** We design an iterative procedure with at most  $r$  steps. See Figure 3 for a visualization. Initialize  $Q_0 := P_0$  and for each  $i \in [1, r]$ , set  $Q_i := \emptyset$ . Let  $S_1$  be the set returned by the oracle  $\mathcal{O}$  on input  $(G_{\text{flow}}[P_0], k)$ . Then  $S_1$  is a  $\beta$ -approximate solution to the instance  $(G_{\text{flow}}[P_0], k)$  of TREEWIDTH- $\eta$ -DELETION, and so  $|S_1| \leq \beta \cdot \text{opt}_{\text{TW-}\eta\text{-DEL}}((G[P_0], k))$ .



■ **Figure 3** Illustration to Lemma 29. The protocol starts by finding a  $\beta$ -approximate solution  $S_1$  to the TREEWIDTH- $\eta$ -DELETION problem in  $G[P_0]$ . Then  $Q_1$  is the set obtained after deleting  $S_1$  from  $P_0$ ; note that  $\text{tw}(G[Q_0]) \leq \eta$ . It then repeats the process of finding a solution to the TREEWIDTH- $\eta$ -DELETION problem inside the old solution  $S_1$  and continues doing so until the obtained new solution is a large fraction of the previous solution. Afterwards, Lossy Reduction Protocol 3 removes the final solution from the graph and compresses the remaining graph using Lemma 32.

In each iteration, we maintain the following invariants: (i)  $(Q_0, Q_1, \dots, Q_r)$  is a partition of  $P_0$ , (ii) for each  $i \in [1, r]$ , the treewidth of  $G_{\text{flow}}[Q_i]$  is at most  $\eta$ , and (iii) at the end of the  $j$ -th iteration,  $|Q_0| \leq \left(\frac{1}{1+\epsilon}\right)^{j-1} |S_1|$ .

In the  $j$ -th iteration,  $j \in [1, r]$ , the protocol calls the oracle  $\mathcal{O}$  on the instance  $(G_{\text{flow}}[Q_0], k)$  and receives a  $\beta$ -approximate solution  $S_j$  to TREEWIDTH- $\eta$ -DELETION on  $(G_{\text{flow}}[Q_0], k)$ . Then  $|S_j| \leq \beta \cdot \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}[Q_0], k))$ . It proceeds with the following case distinction.

1. If  $|Q_0| \leq (1+\epsilon) \cdot |S_j|$ , then the protocol terminates and outputs the partition  $(Q_0, \dots, Q_r)$  which satisfies condition (2a).
2. Otherwise,  $|Q_0| > (1+\epsilon) \cdot |S_j|$ . In this case, we set  $Q_j := Q_0 \setminus S_j$ . Clearly, the treewidth of  $G_{\text{flow}}[Q_j]$  is at most  $\eta$ . Next, we set  $Q_0 := S_j$ .

We now verify that the invariants are satisfied. Clearly  $(Q_0, Q_1, \dots, Q_r)$  is a partition of  $P_0$  and the treewidth of each  $G_{\text{flow}}[Q_i]$ ,  $i \in [1, r]$ , is at most  $\eta$ . If  $j = 1$  then the invariant (iii) holds trivially. Otherwise, due to case distinction, the size of  $Q_0$  drops by factor  $(1+\epsilon)$ , implying  $|Q_0| \leq \left(\frac{1}{1+\epsilon}\right)^{j-1} |S_1|$ .

If the first case occurs before the  $r$ -th iteration, then we are done. Otherwise, the protocol terminates after the  $r$ -th iteration. The invariant (iii) implies  $|Q_0| \leq \left(\frac{1}{1+\epsilon}\right)^{r-1} |S_1|$ . Substituting the definitions of  $r$  and  $S_1$  yields condition (2b):  $|Q_0| \leq \epsilon \cdot \beta \cdot \text{opt}_{\text{TW-}\eta\text{-DEL}}((G_{\text{flow}}[P_0], k))$ . ◀

We argue that one can remove  $Q_0$  from the graph while paying a small loss in accuracy.

► **Lossy Reduction Protocol 3.** Let  $(G, k)$  be an instance of  $\mathcal{F}$ -DELETION where Lossy Reduction Rule 1 has been exhaustively applied. Let  $(P_0, \dots, P_\ell)$  be the protrusion decomposition of  $(G, k)$  from Lemma 24. Let  $(Q_0, Q_1, \dots, Q_r)$  be the partition of  $P_0$  obtained from Lemma 29, on input  $(G_{\text{flow}}[P_0], k)$ . The reduction protocol outputs  $(G' := G - Q_0, k' := k - |Q_0|)$ .

The solution lifting algorithm takes as input the instances  $(G, k)$ ,  $(G', k')$  and  $S' \subseteq V(G')$  which is a  $\beta$ -approximate solution for the instance  $(G', k')$  of  $\mathcal{F}$ -DELETION and outputs  $S := S' \cup Q_0$  as a solution for  $(G, k)$  of  $\mathcal{F}$ -DELETION.

► **Lemma 30** (★). *Lossy Reduction Protocol 3 is a  $(1 + \epsilon)$ -lossy reduction protocol.*

On the other hand, discarding  $Q_0$  allows us to bound the number of simplicial parts. For this, we first need to bound the number of cliques in a graph of bounded treewidth.

► **Proposition 31.** *An  $n$ -vertex graph of treewidth  $\text{tw}$  has  $\mathcal{O}(2^{\text{tw}} \cdot \text{tw} \cdot n)$  distinct cliques.*

**Proof.** An  $n$ -vertex graph  $G$  of treewidth  $\text{tw}$  has a tree decomposition of width  $\text{tw}$  and  $\mathcal{O}(\text{tw} \cdot n)$  nodes [5, Lemma 7.4]. Since each clique of  $G$  is contained inside some bag of this tree decomposition [6, Lemma 12.3.5], the number of distinct cliques are at most  $2^{\text{tw}+1}$  times the number of nodes in a tree decomposition. ◀

► **Lemma 32.** *After the application of Lossy Reduction Protocol 3, the number of distinct cliques in  $G_{\text{flw}}[P_0]$  is  $\mathcal{O}_\eta(k^{3r})$ , where  $r$  is as defined in Lemma 29.*

**Proof.** Let  $(Q_0, Q_1, \dots, Q_r)$  be the partition of  $P_0$  obtained from Lemma 29. After the application of Lossy Reduction Protocol 3, we can assume that  $Q_0 = \emptyset$ . Recall that for each  $i \in [1, r]$ , treewidth of  $G_{\text{flw}}[Q_i]$  is at most  $\eta$ . From Proposition 31, the number of distinct cliques in  $G_{\text{flw}}[Q_i]$  is  $\mathcal{O}_\eta(|Q_i|) \leq \mathcal{O}_\eta(|P_0|) \leq \mathcal{O}_\eta(k^3)$ . Therefore, the number of distinct cliques in  $G_{\text{flw}}[P_0]$  is at most  $\prod_{i=1}^r \mathcal{O}_\eta(k^3) = \mathcal{O}_\eta(k^{3r})$ . ◀

► **Theorem 2** (Uniform lossy protocol). *Let  $\mathcal{F}$  be a finite family of graphs containing at least one planar graph. For any  $\epsilon > 0$ ,  $\mathcal{F}$ -DELETION admits a  $(1 + \epsilon)$ -lossy compression protocol of call size  $\mathcal{O}_{\mathcal{F}}(k^6 + k^{3r+1})$ , and at most  $1 + r$  rounds, where  $r = 1 + \lceil \log_{1+\epsilon}(\frac{1}{\epsilon}) \rceil$ .*

*For the special case of TREEWIDTH- $\eta$ -DELETION, there is a  $(1 + \epsilon)$ -lossy kernelization protocol with same call size and number of rounds.*

**Proof.** Fix any  $\epsilon > 0$ . We describe a polynomial time  $(1 + \epsilon)$ -lossy compression protocol of capacity  $\mathcal{O}_\eta(k^7) + \mathcal{O}_\eta(k^{3r})$  and  $1 + r$  rounds. For any  $\beta \geq 1$  it has access to  $\beta$ -kernelization oracles for  $\mathcal{F}$ -DELETION and TREEWIDTH- $\eta$ -DELETION of capacities  $\mathcal{O}_\eta(k^6) + \mathcal{O}_\eta(k^{3r+1})$ .

Given an instance  $(G, k)$  of  $\mathcal{F}$ -DELETION, first compute a near-protrusion decomposition on the instance  $(G, k)$  in polynomial time using Lemma 22. This gives sets  $X, Z$  with the properties listed in Lemma 22. For each connected component  $C$  of  $G - (X \cup Z)$ , it checks whether Lossy Reduction Rule 1 is applicable. Let  $(G^1, k_1)$  be the instance returned by the reduction algorithm of Lossy Reduction Rule 1 at the end of its exhaustive application. From Lemma 24,  $G^1$  has an  $(\alpha, \mathcal{O}(\eta/\epsilon))$ -protrusion decomposition  $(P_0, \dots, P_\ell)$  for some (unbounded)  $\alpha$  and  $|P_0| = \mathcal{O}_\eta(k^3)$ . Run Lossy Reduction Protocol 3 on the instance  $(G^1, k_1)$  to compute  $(G^2, k_2)$ .

Consider the augmented graph  $G_{\text{flw}}^2$  of  $G^2$ . The number of  $P_i$ ,  $i \in [1, \ell]$ , whose neighborhood in  $G_{\text{flw}}^2$  is non-empty and not a clique is bounded by  $\mathcal{O}_\eta(k^7)$  from Lemma 26. For the remaining  $P_i$ , we partition them into sets such that each part in a fixed set has the same neighborhood in  $G_{\text{flw}}^2$  and any two sets have distinct neighborhood in  $G_{\text{flw}}^2$ . Let the resulting sets be  $R_1, \dots, R_\rho$ . Note that for each  $i \in [1, r]$ ,  $N[R_i]$  is an  $\mathcal{O}(\eta/\epsilon)$ -protrusion in  $G$  and  $N_{G_{\text{flw}}^2}(R_i)$  is a clique in  $G_{\text{flw}}^2$ . From Lemma 32 we infer that  $\rho = \mathcal{O}_\eta(k^{3r})$ . Then  $G^2$  has a  $(\delta, \gamma)$ -protrusion decomposition, where  $\delta = \mathcal{O}_\eta(k^5) + \mathcal{O}_\eta(k^{3r})$  and  $\gamma = \mathcal{O}(\eta/\epsilon)$ .

Let  $(G^3, k_3)$  be the instance obtained from Lemma 14 on input  $(G^2, k_2)$ , with  $\mathcal{O}_{\mathcal{F}}(\delta)$  vertices. If  $G^3$  has a  $k$ -sized solution then the treewidth of  $G^3$  is at most  $k + \eta$  and therefore the number of edges of  $G^3$  is at most  $\mathcal{O}_{\mathcal{F}}(k \cdot \delta)$ . Finally run the  $\beta$ -kernelization algorithm for the problem  $\mathcal{F}$ -DELETION on the instance  $(G^3, k_3)$ . Let  $S_3$  be a  $\beta$ -approximate solution for the problem  $\mathcal{F}$ -DELETION on the instance  $(G^3, k_3)$ .

Using the polynomial-time algorithm of Lemma 14, compute a  $\beta$ -approximate solution  $S_2$  for the instance  $(G^2, k_2)$  of  $\mathcal{F}$ -DELETION. We use the solution lifting algorithm of Lemma 30 with  $S_2$ : let  $S_1$  be the corresponding  $(1 + \epsilon) \cdot \beta$ -approximation solution obtained for the instance  $(G^1, k_1)$  of  $\mathcal{F}$ -DELETION. Finally, the solution lifting algorithm of Lemma 23 takes  $S_1$  and gives a  $(1 + \epsilon) \cdot \beta$ -approximation solution for the instance  $(G, k)$ . ◀

## 7 Conclusion

We have presented new techniques to turn a near-protrusion decomposition into a protrusion decomposition (by sacrificing accuracy) and to process the latter in the presence of disconnected forbidden minors. A main follow-up question is whether one really needs multiple calls to the oracle to obtain a uniform  $(1 + \epsilon)$ -approximate kernel for TREewidth- $\eta$ -DELETION (and other cases of  $\mathcal{F}$ -DELETION). In other words, can we improve the uniform lossy kernelization protocol to uniform lossy kernelization? Secondly, our protocol requires the oracle to handle instances of size  $\mathcal{O}_{\eta, \epsilon}(k^{f(\epsilon)})$  for some function  $f$ . Can we obtain a lossy kernel/protocol that is uniform also with respect to  $\epsilon$ ?

---

## References

- 1 Julien Baste, Ignasi Sau, and Dimitrios M. Thilikos. Hitting minors on bounded treewidth graphs. I. General upper bounds. *SIAM J. Discret. Math.*, 34(3):1623–1648, 2020. doi:10.1137/19M1287146.
- 2 Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996. doi:10.1137/S0097539793251219.
- 3 Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009. doi:10.1016/J.JCSS.2009.04.001.
- 4 Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (Meta) Kernelization. *J. ACM*, 63(5):44:1–44:69, 2016. doi:10.1145/2973749.
- 5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.
- 6 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.
- 7 Eduard Eiben, Diptapriyo Majumdar, and M. S. Ramanujan. On the lossy kernelization for connected treedepth deletion set. In Michael A. Bekos and Michael Kaufmann, editors, *Graph-Theoretic Concepts in Computer Science - 48th International Workshop, WG 2022, Tübingen, Germany, June 22-24, 2022, Revised Selected Papers*, volume 13453 of *Lecture Notes in Computer Science*, pages 201–214. Springer, 2022. doi:10.1007/978-3-031-15914-5\_15.
- 8 Fedor Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.
- 9 Fedor V. Fomin, Tien-Nam Le, Daniel Lokshtanov, Saket Saurabh, Stéphan Thomassé, and Meirav Zehavi. Lossy kernelization for (implicit) hitting set problems. In Inge Li Gørtz, Martin Farach-Colton, Simon J. Puglisi, and Grzegorz Herman, editors, *31st Annual European Symposium on Algorithms, ESA 2023, September 4-6, 2023, Amsterdam, The Netherlands*, volume 274 of *LIPICs*, pages 49:1–49:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.ESA.2023.49.
- 10 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, Geevarghese Philip, and Saket Saurabh. Hitting forbidden minors: Approximation and kernelization. *SIAM J. Discret. Math.*, 30(1):383–410, 2016. doi:10.1137/140997889.

- 11 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar F-Deletion: Approximation, kernelization and optimal FPT algorithms. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 470–479. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.62.
- 12 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. *SIAM J. Comput.*, 49(6):1397–1422, 2020. doi:10.1137/16M1080264.
- 13 Valentin Garnero, Christophe Paul, Ignasi Sau, and Dimitrios M. Thilikos. Explicit linear kernels via dynamic programming. *SIAM J. Discret. Math.*, 29(4):1864–1894, 2015. doi:10.1137/140968975.
- 14 Archontia C. Giannopoulou, Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. Uniform kernelization complexity of hitting forbidden minors. *ACM Trans. Algorithms*, 13(3):35:1–35:35, 2017. doi:10.1145/3029051.
- 15 Anupam Gupta, Euiwoong Lee, Jason Li, Pasin Manurangsi, and Michal Włodarczyk. Losing treewidth by separating subsets. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1731–1749. SIAM, 2019. doi:10.1137/1.9781611975482.104.
- 16 Eva-Maria C. Hols, Stefan Kratsch, and Astrid Pieterse. Approximate turing kernelization for problems parameterized by treewidth. In Fabrizio Grandoni, Grzegorz Herman, and Peter Sanders, editors, *28th Annual European Symposium on Algorithms, ESA 2020, September 7-9, 2020, Pisa, Italy (Virtual Conference)*, volume 173 of *LIPICs*, pages 60:1–60:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.ESA.2020.60.
- 17 Bart M. P. Jansen, Marcin Pilipczuk, and Marcin Wrochna. Turing kernelization for finding long paths in graph classes excluding a topological minor. *Algorithmica*, 81(10):3936–3967, 2019. doi:10.1007/S00453-019-00614-4.
- 18 Bart M. P. Jansen and Michal Włodarczyk. Lossy planarization: A constant-factor approximate kernelization for planar vertex deletion. *SIAM J. Comput.*, 54(1):1–91, 2025. doi:10.1137/22M152058X.
- 19 Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. *ACM Trans. Algorithms*, 12(2), December 2015. doi:10.1145/2797140.
- 20 Tuukka Korhonen, Michal Pilipczuk, and Giannos Stamoulis. Minor containment and disjoint paths in almost-linear time. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 53–61. IEEE, 2024. doi:10.1109/FOCS61266.2024.00014.
- 21 Stefan Kratsch and Pascal Kunz. Approximate turing kernelization and lower bounds for domination problems. In Neeldhara Misra and Magnus Wahlström, editors, *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPICs*, pages 32:1–32:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICs.IPEC.2023.32.
- 22 William Lochet and Roohani Sharma. Uniform polynomial kernel for deletion to  $K_{2,p}$  minor-free graphs. In Julián Mestre and Anthony Wirth, editors, *35th International Symposium on Algorithms and Computation, ISAAC 2024, December 8-11, 2024, Sydney, Australia*, volume 322 of *LIPICs*, pages 46:1–46:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICs.ISAAC.2024.46.
- 23 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 224–237. ACM, 2017. doi:10.1145/3055399.3055456.
- 24 M. S. Ramanujan. On approximate compressions for connected minor-hitting sets. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPICs*, pages 78:1–78:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.ESA.2021.78.

- 25 Jean-Florent Raymond and Dimitrios M. Thilikos. Recent techniques and results on the Erdős-Pósa property. *Discret. Appl. Math.*, 231:25–43, 2017. doi:10.1016/J.DAM.2016.12.025.
- 26 Neil Robertson and Paul D. Seymour. Graph minors. V. Excluding a planar graph. *J. Comb. Theory B*, 41(1):92–114, 1986. doi:10.1016/0095-8956(86)90030-4.
- 27 Neil Robertson and Paul D. Seymour. Graph minors .xiii. the disjoint paths problem. *J. Comb. Theory B*, 63(1):65–110, 1995. doi:10.1006/JCTB.1995.1006.
- 28 Ignasi Sau, Giannos Stamoulis, and Dimitrios M. Thilikos.  $k$ -apices of minor-closed graph classes. II. Parameterized algorithms. *ACM Trans. Algorithms*, 18(3):21:1–21:30, 2022. doi:10.1145/3519028.