PLANAR F-DELETION: Approximation, Kernelization and Optimal FPT Algorithms (Extended Abstract)

Fedor V. Fomin*, Daniel Lokshtanov†, Neeldhara Misra‡and Saket Saurabh‡
*University of Bergen, Norway.
Email: fomin@ii.uib.no
†University of California, USA.
dlokshtanov@cs.ucsd.edu
‡Institute of Mathematical Sciences, India.
(neeldhara|saket)@imsc.res.in

***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.

***Keywords*-approximation; f-deletion; kernelization; fpt; algorithms; graphs;**

I. 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 $S \subseteq V(G)$, $\lvert S\rvert \leq k$, 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 [49], 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 $O(f(k) \cdot n^3)$ [56]. 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 [35], [36]. 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.

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

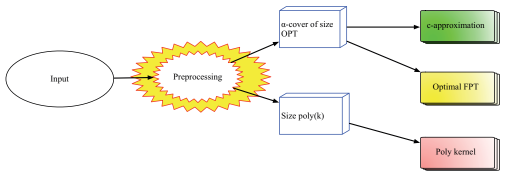

Figure 1. General view of our approach

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 [30]. Precisely, what we use is the modern notion of "protrusion reduction" that has been recently employed in [13], [37] for obtaining meta-kernelization theorems for problems on sparse graphs like planar graphs, graphs of bounded genus [13], graphs excluding a fixed graph as a minor or induced subgraph [37], [34], or graphs excluding a fixed graph as a topological minor [48]. 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 arguments one upper bounds the size of the reduced induced (a graph without large protrusion). The FPT algorithms use the replacement technique developed in [13], [34], 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 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 [13], [37] 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 [34]. 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 [59]. 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 [51], [4]. 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 [50]. 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 [5] and thus the dividing line of approximability lies somewhere else. On a related matter, Yannakakis [58] 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 [58] 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. [32] 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], [13], [37], [46].

The recent development of a framework for ruling out polynomial kernels under certain complexity-theoretic assumptions [12], [23], [38] has added a new dimension to the field and strengthened its connections to classical complexity. For overviews of kernelization we refer to surveys [11], [40] and to the corresponding chapters in books on Parameterized Complexity [33], [52].

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], [18], [42]. 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. [14], improved to $O(k^3)$ by Bodlaender [10], and then finally to $O(k^2)$ by Thomassé [57]. 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 [34]. Philip et al. [53] and Cygan et al. [21] 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 [30], [31] 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 [7], later improved by Djidjev and Reif [26]. Gupta and Impagliazzo studied bounds on the size of a planar intertwine of two given planar graphs [41]. Lagergren [47] 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. [28] 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 $2k + 1$ [25]. Obstructions for graphs with feedback vertex set of size at most k is discussed in the work of Dinneen et al. [24].

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 [29], who gave a non-constructive algorithm running in time $O(f(k) \cdot n^2)$ for some function $f(k)$ [29, Theorem 6]. This result was improved by Bodlaender [8] 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. [19]. Randomized parameterized single exponential algorithm for FEEDBACK VERTEX SET was given by Becker et al. [6] 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 [55] to reduce the running time to $2^{O(k)}n^{O(1)}$ [15], [17], [20], [22], [39], [54]. The current champion for FEEDBACK VERTEX SET are the deterministic algorithm of Cao et al. [15] with running time $O(3.83^k kn^2)$ and the randomized of Cygan et al. with running time time $3^k n^{O(1)}$ [20]. Recently, Joret et al. [44] 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. [53] 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 [21]. Kim et al. [45] gave a single exponential algorithm for $\mathcal{F} = \{K_4\}$. Unless Exponential Time Hypothesis (ETH) fails [16], [43], 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 $c > 1$.*

II. PRELIMINARIES

We use $V(G)$ to denote the vertex set of a graph G, and $E(G)$ to denote the edge set. The degree of a vertex v in G is the number of edges incident on v, and is denoted by $d(v)$. We use $\mathbf{tw}(G)$ to denote the treewidth of the input graph. The following fact about excluding planar graphs as minors will be useful.

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

A t-boundaried graph is a graph G and a set $B \subseteq V(G)$ of size at most t with each vertex $v \in B$ having a label $\ell_G(v) \in \{1, \dots, t\}$. Each vertex in B has a unique label. We refer to B as the *boundary* of G. For a t-boundaried G the function $\delta(G)$ returns the boundary of G. 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 $u_1v_1 \in E(G_1)$ or $u_2v_2 \in E(G_2)$.

Finite Integer Index.: 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 ([13]). *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 [13] and the starting point for our algorithms.

Proposition 3 ([13]). *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.

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 $T \setminus M'$, $|N(C)| \leq 2$.*

III. ALGORITHMS FOR "CONNECTED" p-$\mathcal{F}$-DELETION

In this section we give a randomized FPT algorithm for p-$\mathcal{F}$-DELETION when every graph in $\mathcal{F} \in \mathscr{F}$ is connected, that is, when $\mathcal{F}$ 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 full version of the paper we give an implementation of the algorithm with expected running time $O(n \cdot OPT)$. Then we show how to modify (speed up) the 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 a 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 [9]. ■

```
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'.

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.

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.

Protrusion Decompositions and Proof of Lemma 3. We recall the notion of a protrusion decomposition defined in [13] 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][13] *A graph G has an (α, β)*-protrusion decomposition *if $V(G)$ has a partition $\mathcal{P} = \{R_0, R_1, \dots, R_t\}$ where*

- $\max\{t, |R_0|\} \leq \alpha$,
- *each $N_G[R_i]$, $i \in \{1, \dots, t\}$ is a β-protrusion of G, and*
- *for all $i > 1$, $N[R_i] \subseteq R_0$.*

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

We can 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.*

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 $R_0 \dots R_t$. 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 is no vertex of degree 0 in S. ∎

Approximation Algorithms. We can show that the algorithm given in Figure 2 gives a randomized constant factor approximation algorithm for $\mathcal{F}$-DELETION, an optimization version of p-$\mathcal{F}$-DELETION. However for this we need a notion of protrusion replacer that is also approximation preserving. Towards this we introduce the notion of *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. More precisely it can be defined as follows.

Definition 1 (Lossless Protrusion Replacer)**.** *A lossless protrusion replacer for a 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-boundaried 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 show that not only $\mathcal{F}$-DELETION but a large number of vertex subset problems, that are "strongly monotone" [13], admit lossless protrusion replacer. Then using this rather than the normal protrusion replacer in the algorithm given in Figure 2 and letting the algorithm run until the current graph does not have any graph from $\mathcal{F}$ as a minor, gives us a randomized constant factor approximation algorithm.

Fast Protrusion Replacement Algorithms. What makes the polynomial factor of the algorithm given in Figure 2 large is the algorithm of Lemma 2 to remove all large enough protrusions with small border size. However, one can show that reducing almost all protrusions instead of all protrusions is sufficient to speed up the algorithm given in Figure 2. We show that if the graph G on n vertices satisfies that $n > ck$, then then there exists a protrusion decomposition having $\alpha > \frac{n}{d}$ for some fixed constant d. Then we show that in expected linear time we can discover some constant fraction of these protrusions and replace them. Since each protrusion replacement reduces the size of the graph by at least one, we have that the size of the reduced instance is a constant factor smaller than the original graph. A repeated application of this idea gives us our fast protrusion replacers.

IV. AN OVERVIEW OF THE KERNELIZATION ALGORITHM

Towards kernelization, we begin by showing 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 $|\partial(C)| \leq r$ 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 algorithm begins by running a c-approximation algorithm for $\mathcal{F}$-DELETION. If the solution returned by the approximation algorithm is more than ck, the the kernelization algorithm returns a trivial no instance. Otherwise, let X denote the approximate solution. Based on X, and exploiting the fact that $G \setminus X$ has constant treewidth (say η), we are able to construct $Z \subseteq G \setminus X$ on $O(k^3)$ vertices such that: (a) for every connected component C of $G \setminus (X \cup Z)$, $|N(C) \cap Z| \leq 2(\eta + 1)$, and (b) 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.

We now consider the sets X and Z and a single component C of $G \setminus (X \cup Z)$. We show 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 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 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 in G'. This leads to the first reduction rule, which is designed to ensure that the proof of correctness works as described.

We introduce the notion of an interesting set, and their signatures. The formal definitions are beyond the scope of this abridged discussion. We note that if $P \subseteq C$ is interesting, then it has a constant-sized neighborhood in $G \setminus X$. Call $B \subseteq V(G)$ a *candidate boundary set* if it satisfies that outside X, B is contained in $N(P)$, and $|B \cap X| \leq \eta + 1$. Let $\mathcal{Q}$ denote the family of all graphs on $|B|+h$ vertices with boundary set B. For a family $\mathcal{Q}$ of t-boundaried graphs, a $\mathcal{Q}$-deletion set for a t-boundaried graph G is a subset of vertices S such that $G \setminus S$ does not contain any graph from $\mathcal{Q}$ as a boundaried minor. The signature of P is a function that records the size of the smallest $\mathcal{Q}$-

deletion set for G_P^B (which denotes the graph with boundary B and internal set P), for all candidate boundary sets B and the corresponding families $\mathcal{Q}$. The k-truncated signature simply agrees with the signature whenever the output is at most k, and returns infinity otherwise. This brings us to the first reduction rule.

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 endpoint 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).*

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. For a component C, a candidate boundary set B 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 then have the following reduction rule, which amounts to showing that whenever the number of components in $G \setminus X$ is too large then we can remove a vertex from one of them.

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.*

Since X is a solution returned by a constant-factor approximation algorithm, we have that $|X| = O(k)$. We also have that $|Z| = O(k^3)$ by construction. Further, it can be shown with a careful analysis that in a graph reduced with respect to the first reduction rule, the size of each component of $G \setminus (X \cup Z)$ is $k^{O(1)}$. Finally, reduction with respect to the second reduction rule ensures that the number of components of $G \setminus (X \cup Z)$ is $k^{O(1)}$.

V. 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 [37] with fast protrusion reducers, we have that kernelization algorithms on apex-free and H-minor free graphs for *all* bidimensional problems from [37] 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 [35], 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 [35] 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 [35] 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 [36] 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 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] R. Bar-Yehuda and S. Even. A linear-time approximation algorithm for the weighted vertex cover problem. *J. Algorithms*, 2(2):198–203, 1981.

[5] 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.

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

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

[8] H. Bodlaender. Treewidth: Algorithmic techniques and results. In MFCS'97, volume 1295 of *LNCS*, pages 19–36. Springer, 1997.

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

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

[11] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In IWPEC 2009, volume 5917 of *LNCS*, pages 17–37, 2009.

[12] 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.

[13] 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.

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

[15] Y. Cao, J. Chen, and Y. Liu. On feedback vertex set new measure and new structures. In SWAT 2010, volume 6139 of *LNCS*, pages 93–104, 2010.

[16] 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.

[17] 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.

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

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

[20] 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 FOCS 2011, pages 150–159. IEEE, 2011.

[21] 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 *LNCS*, pages 95–106, 2010.

[22] 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.

[23] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *STOC 2010*, pages 251–260, ACM, 2010.

[24] 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.

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

[26] H. Djidjev and J. H. Reif. An efficient algorithm for the genus problem with explicit construction of forbidden subgraphs. In STOC 91, pages 337–347, ACM, 1991.

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

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

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

[30] 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.

[31] 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.

[32] 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 *LNCS*, pages 191–204. Springer, 2010.

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

[34] F. V. Fomin, D. Lokshtanov, N. Misra, G. Philip, and S. Saurabh. Hitting forbidden minors: Approximation and kernelization. In STACS 2011, volume 9 of *LIPIcs*, pages 189–200. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[35] F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh. Bidimensionality and EPTAS. In SODA 2011, pages 748–759. SIAM, 2011.

[36] F. V. Fomin, D. Lokshtanov, and S. Saurabh. Bidimensionality and geometric graphs. In SODA 2012, pages 1563–1575. SIAM, 2012.

[37] F. V. Fomin, D. Lokshtanov, S. Saurabh, and D. M. Thilikos. Bidimensionality and kernels. In SODA 2010, pages 503–510, 2010.

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

[39] 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.

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

[41] A. Gupta and R. Impagliazzo. Computing planar intertwines. In FOCS 1991, pages 802–811. IEEE Computer Society, 1991.

[42] 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.

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

[44] G. Joret, C. Paul, I. Sau, S. Saurabh, and S. Thomassé. Hitting and harvesting pumpkins. In ESA 2011, volume 6942 of *LNCS*, pages 394–407. Springer, 2011.

[45] E. J. Kim, C. Paul, and G. Philip. A single-exponential FPT algorithm for K_4-minor cover problem. In SWAT, volume 7357 of *LNCS*, pages 119–130. Springer, 2012.

[46] S. Kratsch and M. Wahlström. Compression via matroids: a randomized polynomial kernel for odd cycle transversal. In SODA 2012, pages 94–103, 2012.

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

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

[49] 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.

[50] C. Lund and M. Yannakakis. The approximation of maximum subgraph problems. In ICALP 1993, volume 700 of *LNCS*, pages 40–51, 1993.

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

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

[53] G. Philip, V. Raman, and Y. Villanger. A quartic kernel for pathwidth-one vertex deletion. In *WG*, volume 6410 of *LNCS*, pages 196–207, 2010.

[54] 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.

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

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

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

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

[59] M. Yannakakis. Some open problems in approximation. In CIAC 1994, volume 778 of *Lecture Notes in Comput. Sci.*, pages 33–39, 1994.