# A Turing kernelization dichotomy for structural parameterizations of F-Minor-Free Deletion

Journal of Computer and System Sciences 119 (2021) 164–182

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

# A Turing kernelization dichotomy for structural parameterizations of $\mathcal{F}$-Minor-Free Deletion

Huib Donkers *, Bart M.P. Jansen [1]

*Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands*

ARTICLE INFO

*Article history:*
Received 10 October 2019
Received in revised form 8 January 2021
Accepted 17 February 2021
Available online 3 March 2021

*Keywords:*
Kernelization
Turing kernelization
Minor-free deletion
Subgraph-free deletion
Structural parameterization

ABSTRACT

For a fixed finite family of graphs $\mathcal{F}$, the $\mathcal{F}$-Minor-Free Deletion problem takes as input a graph $G$ and integer $\ell$ and asks whether a size-$\ell$ vertex set $X$ exists such that $G - X$ is $\mathcal{F}$-minor-free. $\{K_2\}$-Minor-Free Deletion and $\{K_3\}$-Minor-Free Deletion encode Vertex Cover and Feedback Vertex Set respectively. When parameterized by the feedback vertex number of $G$ these two problems are known to admit a polynomial kernelization. We show $\{P_3\}$-Minor-Free Deletion parameterized by the feedback vertex number is MK[2]-hard. This rules out the existence of a polynomial kernel assuming NP $\not\subseteq$ coNP/poly. Our hardness result generalizes to any $\mathcal{F}$ containing only graphs with a connected component of at least 3 vertices, using as parameter the vertex-deletion distance to treewidth $\min \mathrm{tw}(\mathcal{F})$, where $\min \mathrm{tw}(\mathcal{F})$ denotes the minimum treewidth of the graphs in $\mathcal{F}$. For all other families $\mathcal{F}$ we present a polynomial Turing kernelization. Our results extend to $\mathcal{F}$-Subgraph-Free Deletion.



## 1. Introduction

**Background and motivation.** Kernelization is a framework for the scientific investigation of provably effective preprocessing procedures for NP-hard problems. It uses the notion of a parameterized (decision) problem to capture meaningful performance guarantees for preprocessing. In a parameterized problem, every problem input $x$ has an associated integer $k$ called the parameter, which captures the difficulty of the input in some way. A *kernelization* for a parameterized problem is a polynomial-time algorithm that transforms any parameterized instance $(x, k)$ into an instance $(x', k')$ with the same answer, such that $|x'|$ and $k'$ are both bounded by $f(k)$ for some computable function $f$. The function $f$ is the *size* of the kernel. Of particular interest are kernels of polynomial size. Determining which parameterized problems admit kernels of polynomial size has become a rich area of algorithmic research [1–3].

A common approach in kernelization [4–7] is to take the solution size as the parameter $k$, with the aim of showing that large inputs that ask for a small solution can be efficiently reduced in size. However, this method does not give any nontrivial guarantees when the solution size is known to be proportional to the total size of the input. For that reason, there is an alternative line of research [8–15] that focuses on parameterizations based on a measure of nontriviality of the instance (cf. [16]). One formal way to capture nontriviality of a graph problem is to measure how many vertex-deletions are needed to reduce the input graph to a graph class in which the problem can be solved in polynomial time. Since many graph

* Corresponding author.
*E-mail addresses:* h.t.donkers@tue.nl (H. Donkers), b.m.p.jansen@tue.nl (B.M.P. Jansen).

[1] Supported by NWO Gravitation grant 024.002.003 “Networks”.

https://doi.org/10.1016/j.jcss.2021.02.005

problems can be solved in polynomial time on trees and forests, the structural graph parameter *feedback vertex number* (the minimum number of vertex deletions needed to make the graph acyclic, i.e., a forest) is a relevant measure of the distance of the input to a trivially solvable one.

Previous research has shown that for the Vertex Cover problem, there is a polynomial kernel parameterized by the feedback vertex number [12]. This preprocessing algorithm guarantees that inputs which are large with respect to their feedback vertex number can be efficiently reduced. The Vertex Cover problem is the simplest in a family of so-called minor-free deletion problems. For a fixed finite family of graphs $\mathcal{F}$, an input to $\mathcal{F}$-Minor-Free Deletion consists of a graph $G$ and an integer $\ell$. The question is whether there is a set $S$ of at most $\ell$ vertices in $G$, such that the graph $G - S$ obtained by removing these vertices does not contain any graph from $\mathcal{F}$ as a minor. Various classic graph optimization problems such as Vertex Cover, Feedback Vertex Set, and Vertex Planarization fit this framework by a suitable choice of $\mathcal{F}$. The investigation of minor-free deletion problems has led to numerous advances in the study of kernelization and parameterized algorithmics [17,18,5,19]. Motivated by the fact that Vertex Cover and Feedback Vertex Set, arguably the simplest $\mathcal{F}$-Minor-Free Deletion problems, admit polynomial kernels when parameterized by the feedback vertex number, we set out to resolve the following question: Do *all* $\mathcal{F}$-Minor-Free Deletion problems admit a polynomial kernel when parameterized by the feedback vertex number?

**Results.** To our initial surprise, we prove that the answer to this question is *no*. While the parameterization by feedback vertex number admits polynomial kernels for $\mathcal{F} = \{K_2\}$ [12], for $\mathcal{F} = \{K_3\}$ [20,21,7], and for any set $\mathcal{F}$ containing a planar graph[2] but no forests [5], there are also cases that do not admit polynomial kernels (under the assumption that $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$, which we tacitly assume throughout the informal discussion in this introduction). For example, the case of $\mathcal{F}$ consisting of a single graph $P_3$ that forms a path on three vertices does not admit a polynomial kernel. This lower bound for $\mathcal{F} = \{P_3\}$ follows from a more general theorem that we state below.

Recall that a graph is a forest if and only if its treewidth is one [22]. Hence the feedback vertex number is exactly the minimum number of vertex deletions needed to obtain a graph of treewidth one. Let $\mathrm{tw}(G)$ denote the treewidth of graph $G$, and define $\min \mathrm{tw}(\mathcal{F}) := \min_{H \in \mathcal{F}} \mathrm{tw}(H)$. Our lower bound also holds for $\mathcal{F}$-Subgraph-Free Deletion, which is the related problem that asks whether there is a vertex set $S$ of size at most $k$ such that $G - S$ contains no graph $H \in \mathcal{F}$ as a *subgraph*. We prove the following.

**Theorem 1.** *Let $\mathcal{F}$ be a finite set of graphs, such that each graph in $\mathcal{F}$ has a connected component on at least three vertices. Then $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion do not admit polynomial kernels when parameterized by the vertex-deletion distance to a graph of treewidth* $\min \mathrm{tw}(\mathcal{F})$, *unless* $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

To see that Theorem 1 implies the claimed lower bound for $\mathcal{F} = \{P_3\}$, observe that whenever $\mathcal{F}$ contains an acyclic graph with at least one edge we have $\min \mathrm{tw}(\mathcal{F}) = 1$ and therefore the vertex-deletion distance to treewidth $\min \mathrm{tw}(\mathcal{F})$ equals the feedback vertex number. The theorem also generalizes earlier results of Cygan et al. [9, Theorem 13], who investigated the problem of *losing treewidth*. They proved that for each fixed $1 \leq \eta < \rho$, the $\eta$-Transversal problem (delete at most $\ell$ vertices to get a graph of treewidth at most $\eta$) does not have a polynomial kernel when parameterized by the vertex-deletion distance to treewidth $\rho$. Since the treewidth of a graph does not increase when taking minors, there is a finite set $\mathcal{F}_\eta$ of forbidden minors (cf. [23]) that characterize the graphs of treewidth at most $\eta$. As the members of the obstruction set for $\eta \geq 1$ are easily seen to be connected, have treewidth $\eta + 1$, and at least three vertices, the lower bound of Theorem 1 encompasses the theorem of Cygan et al. and generalizes it to arbitrary $\mathcal{F}$-Minor-Free Deletion problems.

Theorem 1 is obtained through a polynomial-parameter transformation from the cnf-sat problem parameterized by the number of variables, for which a superpolynomial kernelization lower bound is known [24,25]. The main technical contribution in the hardness proof consists of the design of a gadget that acts as a clause checker. A certain budget of vertex deletions is available to break all $\mathcal{F}$-minors present in the gadget, and this is possible if and only if one of the neighboring vertices in a variable gadget is removed by the solution. This removal encodes that the variable is set in a way that satisfies the clause. The intricate part of the construction is to design the gadget knowing only that $\mathcal{F}$ has a graph with a connected component of at least three vertices. Here we extensively rely on the fact that minimal minor models of biconnected graphs live in biconnected subgraphs, together with the fact that the treewidth of a graph does not increase when attaching structures along cut vertices in a tree-like manner.

Using the framework of Hermelin et al. [26], our polynomial-parameter transformation from cnf-sat parameterized by the number of variables to the structural parameterization of $\mathcal{F}$-Minor-Free Deletion, also rules out the existence of polynomial-size *Turing* kernelizations under a certain hardness assumption. Turing kernelization [27] is a relaxation of the traditional form of kernelization. Intuitively, it investigates whether inputs $(x, k)$ can be solved efficiently using the answers to subproblems of size $f(k)$ which are provided by an oracle, which models an external computation cluster. Note that a parameterized problem that has a kernel of size $\mathcal{O}(k^c)$ can be solved by a polynomial-time algorithm that first spends polynomial time to prepare a query of size $\mathcal{O}(k^c)$, and then queries an oracle for its answer. Turing kernelization investigates

[2] If $\mathcal{F}$ contains no forests, then any acyclic graph is $\mathcal{F}$-minor free, implying the size of an optimal solution is at most the size of a feedback vertex set. Hence the kernelization for the solution-size parameterization yields a kernel of size bounded polynomially in the feedback vertex number.

if and how polynomial-time algorithms can solve NP-hard parameterized problems by querying an oracle for the answers to instances of size $k^{\mathcal{O}(1)}$, potentially multiple times. Some problems that do *not* admit polynomial kernelizations, do admit polynomial-size Turing kernelizations [28–32].

Formally, a Turing kernelization of size $f$ for a parameterized problem $\mathcal{Q}$ is an algorithm that can query an oracle to obtain the answer to any instance of problem $\mathcal{Q}$ of size and parameter bounded by $f(k)$ in a single step, and using this power solves any instance $(x, k)$ in time polynomial in $|x| + k$. The reduction proving Theorem 1 also proves the non-existence of polynomial-size Turing kernelizations, unless all parameterized problems in the complexity class MK[2] defined by Hermelin et al. [26] have polynomial Turing kernels. (The CNF-SAT problem with clauses of unbounded length, parameterized by the number of variables, is MK[2]-complete [26, Thm. 1, cf. Thm. 10] and widely believed *not* to admit polynomial-size Turing kernels.)

Motivated by the general form of the lower bound statement in Theorem 1, we also investigate upper bounds and derive a complexity dichotomy. For any $\mathcal{F}$ that does not meet the criterion of Theorem 1, we obtain a polynomial Turing kernel.

**Theorem 2.** *Let $\mathcal{F}$ be a finite set of graphs, such that some $H \in \mathcal{F}$ has no connected component of three or more vertices. Then $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion admit polynomial Turing kernels when parameterized by the vertex-deletion distance to a graph of treewidth* $\min \mathrm{tw}(\mathcal{F})$.

The main insight in the Turing kernelization is the following. If $H \in \mathcal{F}$ has no connected component of three or more vertices, then $H$ consists of disjoint edges and isolated vertices. If $H$ only has isolated vertices, then $\mathcal{F}$-Minor-Free Deletion is polynomial-time solvable because the leftover graph has less than $|V(H)| \in \mathcal{O}(1)$ vertices, for which we can search by brute force. Otherwise, $H$ is a matching of size $t \geq 1$ plus potentially some isolated vertices. The isolated vertices turn out only to make a difference if the solution $\mathcal{F}$-free graph has constant size. In the interesting case, we can focus on $H \in \mathcal{F}$ being a matching of size $t$. Then a graph that is $\mathcal{F}$-minor-free does not admit a matching of size $t$, and therefore has a vertex cover of size at most $t$. Hence a solution to $\mathcal{F}$-Minor-Free Deletion can be extended to a vertex cover by including $\mathcal{O}(1)$ additional vertices. Using the Tutte-Berge formula, we can make the relation between $\mathcal{F}$-Minor-Free Deletion and the vertex cover precise, and use it to reduce an instance of $\mathcal{F}$-Minor-Free Deletion parameterized by deletion distance to $\min \mathrm{tw}(\mathcal{F})$, to the logical OR of a polynomial number of instances of Vertex Cover parameterized by deletion distance to $\min \mathrm{tw}(\mathcal{F})$. If $\mathcal{F}$ has a graph with no component of size at least three, then $\min \mathrm{tw}(\mathcal{F}) = 1$, implying that the parameter is the feedback vertex set size. This allows us to use the polynomial kernel for Vertex Cover parameterized by feedback vertex set on each generated instance. We query the resulting instances of size $k^{\mathcal{O}(1)}$ to the oracle to find the answer.

**Organization.** We present preliminaries on graphs and kernelization in Section 2. Section 3 develops the lower bounds on (Turing) kernelization when all graphs in $\mathcal{F}$ have a connected component with at least three vertices. In Section 4 we show that in all other cases, a polynomial Turing kernelization exists.

## 2. Preliminaries

All graphs we consider in this paper are simple, finite and undirected. We denote the vertex set and edge set of a graph $G$ by $V(G)$ and $E(G)$ respectively. We call the graph on the empty vertex set the *null graph*. For a vertex set $S \subseteq V(G)$ let $G[S]$ be the subgraph of $G$ induced by $S$, and let $G - S$ denote the subgraph of $G$ induced by $V(G) \setminus S$. For a vertex $v$ we use $G - v$ as shorthand for $G - \{v\}$. For a non-negative integer $n$ we use $n \cdot G$ to denote the graph consisting of $n$ disjoint copies of $G$. Let $N_G(S)$ and $N_G(v)$ denote the open neighborhood in $G$ of a vertex set $S$ and a vertex $v$ respectively. Let $\deg_G(v)$ denote the degree of $v$ in $G$. The subscript may be omitted when $G$ is clear from the context. We use $\mathrm{FVS}(G)$ to denote the feedback vertex number of $G$.

A graph $H$ is a minor of graph $G$, denoted by $H \preceq G$, if $H$ can be obtained from $G$ by a series of edge contractions, edge deletions, and vertex deletions. An $H$-model in $G$ is a function $\varphi\colon V(H) \to 2^{V(G)}$ such that (i) for every vertex $v \in V(H)$, the graph $G[\varphi(v)]$ is connected, (ii) for every edge $\{u, v\} \in E(H)$ there exists an edge $\{u', v'\} \in E(G)$ with $u' \in \varphi(u)$ and $v' \in \varphi(v)$, and (iii) for distinct $u, v \in V(H)$ we have $\varphi(v) \cap \varphi(u) = \emptyset$. The sets $\varphi(v)$ are called *branch sets*. Clearly, $H \preceq G$ if and only if there is an $H$-model in $G$. For any function $f\colon A \to B$ and set $A' \subseteq A$ we use $f(A')$ as a shorthand for $\bigcup_{a \in A'} f(a)$. Specifically in the case of an $H$-model $\varphi$ in a graph $G$, we use $\varphi(H)$ to denote $\bigcup_{v \in V(H)} \varphi(v)$. An $H$-model $\varphi$ is called *minimal* if there does not exist an $H$-model $\varphi'$ with $\varphi'(H) \subsetneq \varphi(H)$. We say a graph $H$ is a *component-wise minor* of a graph $G$, denoted as $H \precsim G$, when every connected component of $H$ is a minor of $G$.

**Observation 3.** *For graphs $H$ and $G$, if $H \preceq G$ and $G \preceq H$ then $H$ and $G$ are isomorphic.*

**Observation 4.** *For graphs $H$ and $G$, if $H \precsim G$ then* $\mathrm{tw}(H) \leq \mathrm{tw}(G)$. *(The treewidth of $H$ is the maximum treewidth of its connected components and each connected component $H'$ of $H$ is a minor of $G$, hence* $\mathrm{tw}(H') \leq \mathrm{tw}(G)$.*)*

**Definition 1.** Let $\mathcal{F}$ be a family of graphs and let $G \in \mathcal{F}$. For every relation $\trianglelefteq \in \{\preceq, \precsim\}$ we define minimal and maximal elements as follows:

- $G$ is said to be $\trianglelefteq$-*minimal* in $\mathcal{F}$ when for all graphs $H \in \mathcal{F}$ we have $H \trianglelefteq G \Rightarrow G \trianglelefteq H$.
- $G$ is said to be $\trianglelefteq$-*maximal* in $\mathcal{F}$ when for all graphs $H \in \mathcal{F}$ we have $G \trianglelefteq H \Rightarrow H \trianglelefteq G$.

**Definition 2.** We call a connected component $C$ of a graph $G$ a $\preceq$-*maximal component* of $G$ when $C$ is $\preceq$-maximal in the set of graphs that form the connected components of $G$.

For $type \in \{\text{minor}, \text{subgraph}\}$ and a finite family of graphs $\mathcal{F}$, we define:

$\mathcal{F}$-*type*-Free Deletion
**Input:** A graph $G$ and an integer $\ell$.
**Parameter:** vertex-deletion distance of $G$ to a graph of treewidth $\min \text{tw}(\mathcal{F})$.
**Question:** Is there a set $X \subseteq V(G)$ of at most $\ell$ vertices such that $G - X$ does not contain any $H \in \mathcal{F}$ as a *type*?

A vertex $v \in V(G)$ is a *cut vertex* when its removal from $G$ increases the number of connected components. A graph is called *biconnected* when it is connected and contains no cut vertex. A *biconnected component* of a graph $G$ is a maximal biconnected subgraph of $G$. For any integer $\alpha$, a graph $G$ is called $\alpha$-*robust* when $|V(G)| \geq \alpha$ and no vertex $v \in V(G)$ exists such that $G - v$ contains a connected component with strictly less than $\alpha - 1$ vertices.

**Proposition 5.** *Any graph $G$ has a unique maximal $\alpha$-robust subgraph. Any $\alpha$-robust subgraph of $G$ is a subgraph of the maximal $\alpha$-robust subgraph of $G$.*

**Proof.** The proposition follows straightforwardly from the fact that if $G[A]$ and $G[B]$ are $\alpha$-robust, then so is $G[A \cup B]$. We now prove this fact.

Consider two vertex sets $A, B \subseteq V(G)$, such that $G[A]$ and $G[B]$ are $\alpha$-robust. We show that $G[A \cup B]$ is $\alpha$-robust. Since $G[A]$ is $\alpha$-robust we have $|A| \geq \alpha$ so then $|A \cup B| \geq \alpha$. Suppose for contradiction that there exists a vertex $v \in A \cup B$ such that $G[A \cup B] - v$ contains a connected component of size smaller than $\alpha - 1$. Let $C$ be the vertices of this connected component. We know $C$ contains vertices of at least one of $A$ and $B$. Assume, without loss of generality, $A \cap C \neq \emptyset$. Then $G[A \cap C]$ is a connected component of size less than $\alpha - 1$ in $G[A] - v$. If $v \in A$ this directly contradicts $\alpha$-robustness of $G[A]$, so assume $v \notin A$. Now $G[A]$ contains a connected component with less than $\alpha - 1$ vertices. Since $|C| < \alpha \leq |A|$ there exists a vertex $u \in A \setminus C$, so then $G[A] - u$ contains a connected component with less than $\alpha - 1$ vertices, which contradicts $\alpha$-robustness of $G[A]$. □

We define a *leaf-block* of a graph $G$ as a biconnected component of $G$ that contains at most one vertex $v$ that is a cut vertex in $G$. The size of a leaf-block $H$ is $|V(H)|$. The size of a smallest leaf-block of a graph $G$ is denoted as $\lambda(G)$. Observe that $G$ is $\alpha$-robust if and only if $\lambda(G) \geq \alpha$. For any graph $G$ and integer $\alpha$, let $\alpha$-prune$(G)$ denote the unique maximal $\alpha$-robust subgraph of $G$, which may be empty. Note that $\alpha$-prune$(G)$ can be obtained from $G$ by repeatedly removing interior vertices of leaf blocks of size less than $\alpha$.

A *polynomial-parameter transformation* from parameterized problem $\mathcal{P}$ to parameterized problem $\mathcal{Q}$ is a polynomial-time algorithm that, given an instance $(x, k)$ of $\mathcal{P}$, outputs an instance $(x', k')$ of $\mathcal{Q}$ such that all of the following are true:

1. $(x, k) \in \mathcal{P} \Leftrightarrow (x', k') \in \mathcal{Q}$,
2. $k'$ is upper-bounded by a polynomial in $k$.

## 3. Lower bound

In this section we consider the case where all graphs in $\mathcal{F}$ contain a connected component of at least three vertices and give a polynomial-parameter transformation from cnf-sat parameterized by the number of variables. In order to construct a clause gadget $G$ for a clauses with more than two literals, our construction relies on the presence of a connected component $H_\uparrow$ with at least three vertices in a $\precsim$-minimal graph in $\mathcal{F}$. Such a graph only exists when *all* graphs in $\mathcal{F}$ have a connected component on at least 3 vertices. Intuitively, the reason we need at least three vertices is as follows. The gadget for a clause $C_i$ is constructed based on the sequence of its literals $\ell_1, \ldots, \ell_{|C_i|}$. For each literal $\ell_i$, there is a corresponding part of the gadget which checks three things: whether a literal before $\ell_i$ is satisfied, whether $\ell_i$ itself is satisfied, and whether a literal after $\ell_i$ is satisfied. To implement this check, we need that $H$ has at least three vertices.

### 3.1. Properties of biconnected and robust subgraphs

Our construction exploits the way in which biconnected components of $H$ and the clause gadget $G$ restrict the options for an $H$-model to exist in $G$. We therefore first derive some relevant properties.

**Proposition 6.** *If $H$ is an $\alpha$-robust graph and $\varphi$ is a minimal $H$-model in a graph $G$, then $G[\varphi(H)]$ is $\alpha$-robust.*

**Proof.** Since $H$ is $\alpha$-robust we have $|V(H)| \geq \alpha$. So $|\varphi(H)| \geq |V(H)| \geq \alpha$ and it remains to verify that $G[\varphi(H)] - v$ does not have any connected components smaller than $\alpha - 1$ for any $v$. Take an arbitrary vertex $v \in \varphi(H)$ and let $u \in V(H)$ be such that $v \in \varphi(u)$. Since $H - u$ does not have connected components smaller than $\alpha - 1$, the graph $G[\varphi(H)] - \varphi(u)$ cannot have connected components smaller than $\alpha - 1$. Consider a spanning tree of $G[\varphi(u)]$. Each leaf of this spanning tree must be connected to a vertex in a different branch set, otherwise $\varphi$ is not minimal. We know every connected component in $G[\varphi(u)] - v$ contains at least one leaf of this spanning tree, hence every connected component of $G[\varphi(u)] - v$ is connected to $G[\varphi(H)] - \varphi(u)$. So $G[\varphi(H)] - v$ does not contain a connected component smaller than $\alpha - 1$. Since $v$ was arbitrary, the graph $G[\varphi(H)]$ is $\alpha$-robust. □

**Proposition 7.** *If $\varphi$ is an $H$-model in $G$ and $B$ is a biconnected component of $H$, then $G[\varphi(B)]$ contains a biconnected subgraph on at least $|B|$ vertices.*

**Proof.** Since $G[\varphi(B)]$ clearly contains $B$ as a minor, there is a minimal $B$-model $\varphi'$ in $G$ with $\varphi'(B) \subseteq \varphi(B)$, so that $G[\varphi'(B)]$ is a subgraph of $G[\varphi(B)]$. It suffices to show that $G[\varphi'(B)]$ contains a biconnected component on at least $|V(B)|$ vertices. Since $B$ is biconnected, it is $|V(B)|$-robust so by Proposition 6 we know $G[\varphi'(B)]$ is $|V(B)|$-robust. Hence $G[\varphi'(B)]$ contains a biconnected component on at least $|V(B)|$ vertices. □

**Observation 8.** *For any graph $H$, which may be the null graph, and integers $\alpha \geq \beta$ we have $\alpha$-prune($\beta$-prune($H$)) = $\alpha$-prune($H$).*

**Proposition 9.** *For graphs $H$ and $G$ we have $H \preceq G \Rightarrow \alpha$-prune($H$) $\preceq \alpha$-prune($G$) for any integer $\alpha$.*

**Proof.** Clearly $\alpha$-prune($H$) $\preceq H$ so since $H \preceq G$ we have $\alpha$-prune($H$) $\preceq G$. For simplicity let $H' := \alpha$-prune($H$). Let $\varphi$ be a minimal $H'$-model in $G$ and observe $H' \preceq G[\varphi(H')]$. From Proposition 6 we know $G[\varphi(H')]$ is $\alpha$-robust. Since $G[\varphi(H')]$ is an $\alpha$-robust subgraph of $G$ it is also a subgraph of $\alpha$-prune($G$) by Proposition 5. Hence $\alpha$-prune($H$) $\preceq G[\varphi(H')] \preceq \alpha$-prune($G$). □

**Proposition 10.** *For any $\trianglelefteq \in \{\preceq, \precsim\}$, two integers $\alpha \geq \beta$, and graphs $H$ and $G$ we have that $H \trianglelefteq G \Rightarrow \alpha$-prune($H$) $\trianglelefteq \beta$-prune($G$).*

**Proof.** Suppose $H \preceq G$, then

$$\begin{aligned} \alpha\text{-prune}(H) &\preceq \alpha\text{-prune}(G) && \text{by Proposition 9} \\ &= \alpha\text{-prune}(\beta\text{-prune}(G)) && \text{by Observation 8} \\ &\preceq \beta\text{-prune}(G). \end{aligned}$$

Alternatively, suppose $H \precsim G$. Let $H'$ be a connected component of $\alpha$-prune($H$), then there exists a connected component $H''$ of $H$ such that $H' \preceq H''$. Since $H \precsim G$ we have $H'' \preceq G$ so then $H' \preceq G$. As shown above, this implies $\alpha$-prune($H'$) $\preceq \beta$-prune($G$). Note that $\alpha$-prune($H'$) = $H'$ since $H'$ is a connected component of $\alpha$-prune($H$). It follows that $\alpha$-prune($H$) $\precsim \beta$-prune($G$). □

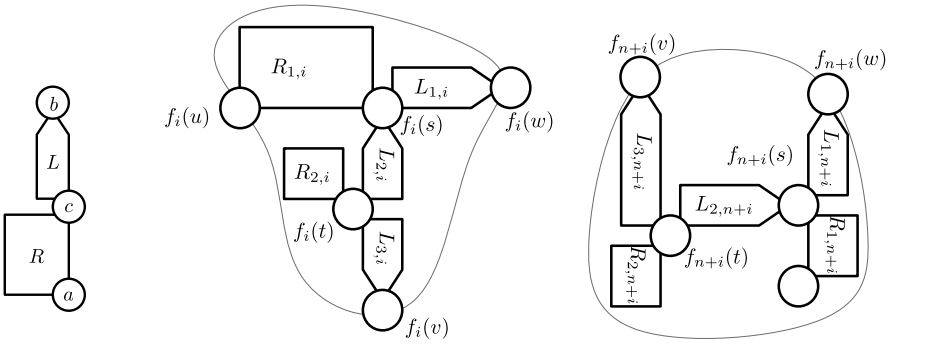
### 3.2. Clause gadget construction

We proceed to construct a clause gadget to be used in the polynomial-parameter transformation from CNF-SAT.

**Lemma 11.** *For any connected graph $H$ with at least three vertices there exists a polynomial-time algorithm that, given an integer $n \geq 1$, outputs a graph $G$ and a vertex set $S \subseteq V(G)$ of size $n$ such that all of the following are true:*

1. $\text{tw}(G) \leq \text{tw}(H)$,
2. *$G$ contains a packing of $3n - 1$ vertex-disjoint $H$-subgraphs,*
3. *$G - S$ contains a packing of $3n - 2$ vertex-disjoint $H$-subgraphs, and*
4. *$\forall v \in S$ there exists $X \subseteq V(G)$ of size $3n - 1$ s.t. all of the following are true:*
   (a) *$v \in X$,*
   (b) *$G - X$ is $H$-minor-free,*
   (c) *$\lambda(H)$-prune($G - X$) $\precsim H$, and*
   (d) *for all connected components $G_c$ of $G - X$ that contain a vertex of $S$ we have $|V(G_c)| < \lambda(H)$ and $G_c$ contains exactly one vertex of $S$.*

**Proof.** Before describing the construction of $G$ and $S$, we define a few subgraphs and vertices. Let $L$ be a smallest leaf-block of $H$. Let $R$ be the graph obtained from $H$ by removing all vertices of $L$ that are not cut vertices in $H$. Note that when $H$ is biconnected, $L = H$ and $R$ is the null graph. We distinguish three distinct vertices $a, b, c$ in $H$. Vertices $c$ and $b$ are both




(a) Graph $H$ (b) Graph $M_i$ for $1 \leq i \leq n$ (c) Graph $M_{n+i}$ for $1 \leq i < n$

**Fig. 1.** We show the situation where $a$ is contained in $R$. Note that $a$ can always be chosen such that it is contained in $R$ when $H$ is not biconnected. Note that the graphs in Fig. 1b and 1c are isomorphic but drawn differently.

part of $L$, where $c$ is the cut vertex (if there is one) and $b$ is any other vertex in $L$. Finally vertex $a$ is any vertex in $H$ that is not $c$ or $b$. See Fig. 1a. In the construction of $G$ we will combine copies of $H$ such that $a$, $b$, and $c$ form cut vertices in $G$ and are part of two different $H$-subgraphs. Intuitively this choice of $b$ and $c$ ensures that removing either one from a copy of $H$ in $G$ means no vertex from the $L$-subgraph of this copy of $H$ can be used in a minimal $H$-model in $G$. In the remainder of this proof we use $f_{K \to K'} \colon V(K) \to V(K')$ for isomorphic graphs $K$ and $K'$ to denote a fixed isomorphism.

**Construction.** Take two copies of $H$, call them $H_1$ and $H_2$. Let $R_1$ and $L_1$ denote the subgraphs of $H_1$ related to $R$ and $L$, respectively, by the isomorphism between $H$ and $H_1$. Similarly let $R_2$ and $L_2$ denote the subgraphs of $H_2$. Take a copy of $L$ which we call $L_3$. Let $M$ be the graph obtained from the disjoint union of $H_1$, $H_2$, and $L_3$ by identifying the pair $f_{H \to H_1}(c)$ and $f_{H \to H_2}(b)$ into a single vertex $s$, and identifying the pair $f_{H \to H_2}(c)$ and $f_{L \to L_3}(c)$ into a single vertex $t$. We label $f_{H \to H_1}(a)$, $f_{H \to H_1}(b)$, and $f_{L \to L_3}(b)$ as $u$, $w$, and $v$ respectively.

This construction is motivated by the fact that the graphs $M - \{v, s\}$, $M - \{u, t\}$, and $M - \{w, t\}$ are all $H$-minor-free, which we will exploit in the formal correctness argument later. We will connect copies of $M$ to each other via the vertices $u$, $v$, and $w$ such that, although two vertices need to be removed in every copy of $M$, one such vertex can always be in two copies of $M$ at the same time.

Now take $2n-1$ copies of $M$, call them $M_1, \ldots, M_{2n-1}$. For readability we denote $f_{M \to M_i}$ as $f_i$ for all $1 \leq i \leq 2n-1$. For all $1 \leq i < n$ we identify $f_i(w)$ and $f_{n+i}(v)$, and we identify $f_{n+i}(w)$ and $f_{i+1}(u)$. Let this graph be $G$, and let $S$ be the set of vertices $f_i(v)$ for all $1 \leq i \leq n$. Let $H_{1,i}$, $H_{2,i}$, $R_{1,i}$, $R_{2,i}$, $L_{1,i}$, $L_{2,i}$, and $L_{3,i}$ denote the subgraphs in $M_i$ that correspond to the subgraphs $H_1$, $H_2$, $R_1$, $R_2$, $L_1$, $L_2$, and $L_3$ in $M$. See Fig. 1b and 1c.

This concludes the description of graph $G$ and set $S$, see Fig. 2 for an illustration. In Fig. 5 the subgraphs $W_1$, $W_2$, and $W_3$ form a concrete example of $G$ with the choice $H = P_3$ and $n = 4$, $n = 3$, and $n = 3$ respectively. It is easily seen that $G$ and $S$ can be constructed in polynomial time.

**Correctness.** It remains to verify that all conditions of the lemma statement are met.

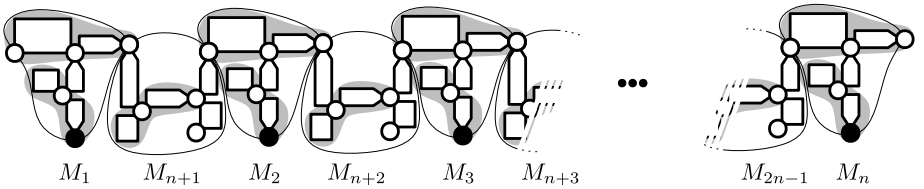
1. Since we connected copies of $L$ and $R$ in a treelike fashion along cut vertices, we did not introduce any new biconnected components. The treewidth of a graph is equal to the maximum treewidth over all its biconnected components so we know that $\mathrm{tw}(G) \leq \max\{\mathrm{tw}(R), \mathrm{tw}(L)\} = \mathrm{tw}(H)$.

2. For each $1 \leq i \leq n$ we can distinguish two $H$-subgraphs in $M_i$, namely $H_{1,i}$ and $L_{3,i} \cup R_{2,i}$. This gives us $2n$ $H$-subgraphs in $G$. Note that since all $M_1, \ldots, M_n$ are vertex-disjoint, these $2n$ $H$-subgraphs are also vertex-disjoint in $G$. For each $n < i \leq 2n-1$ we distinguish one $H$-subgraph, namely $H_{2,i}$. Note that since $H_{2,i}$ is vertex-disjoint from all $M_1, \ldots, M_{i-1}$, $M_{i+1}, \ldots, M_{2n-1}$ we have a total of $2n + n - 1 = 3n - 1$ vertex-disjoint $H$-subgraphs in $G$. This packing is shown in Fig. 2a.

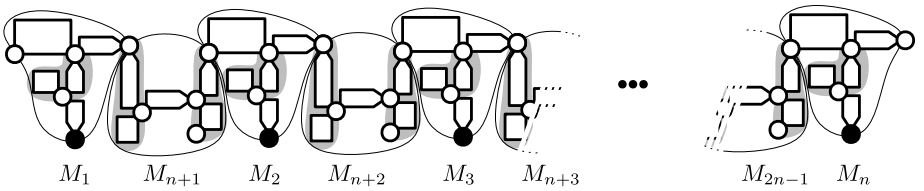
3. Alternatively, for each $1 \leq i \leq n$ we can distinguish one $H$-subgraph in $M_i$, namely $H_{2,i}$. For each $n < i \leq 2n-1$ we distinguish two $H$-subgraphs in $M_i$, namely $H_{1,i}$ and $L_{3,i} \cup R_{2,i}$. Again these $H$-subgraphs are vertex-disjoint, and since they also do not contain any vertices of $S$, they form a packing of $n + 2(n-1) = 3n - 2$ vertex-disjoint $H$-subgraphs in $G - S$. See Fig. 2b.

4. Finally we prove that for all $v \in S$ there exists a set $X \subseteq V(G)$ of size $3n-1$ such that the four parts listed in condition 4. are true. For this purpose we first identify a family $\mathcal{Q}$ of vertex sets such that any $H$-model in $G$ spans at least one vertex set in $\mathcal{Q}$. Let $\mathcal{Q}$ be defined as follows: (see Fig. 3)

$$\mathcal{Q} = \{\{f_i(v), f_i(t)\} \mid 1 \leq i \leq 2n-1\} \cup \{\{f_i(t), f_i(s)\} \mid 1 \leq i \leq 2n-1\}$$
$$\cup \{\{f_i(s), f_i(w)\} \mid n+1 \leq i \leq 2n-1\} \cup \{\{f_i(u), f_i(s), f_i(w)\} \mid 1 \leq i \leq n\}.$$




(a) A packing of $3n-1$ vertex-disjoint $H$-subgraphs in $G$




(b) A packing of $3n-2$ vertex-disjoint $H$-subgraphs in $G-S$

**Fig. 2.** Two packings of vertex-disjoint $H$-subgraphs in $G$ and $G-S$. Vertices in $S$ are marked black.

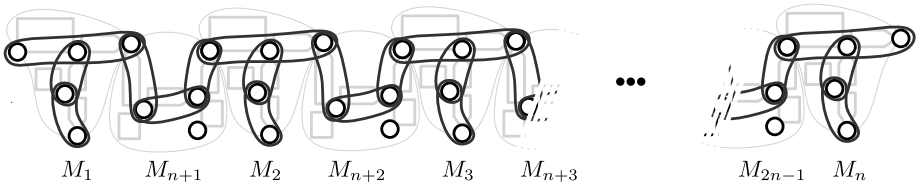



**Fig. 3.** Vertex sets in $\mathcal{Q}$ are encircled.

**Claim 12.** *If $\varphi$ is an $H$-model in $G$, then $\varphi(H) \supseteq Q$ for some $Q \in \mathcal{Q}$.*

**Proof.** Let $\varphi$ be an arbitrary $H$-model in $G$. We know from Proposition 7 that $G[\varphi(L)]$ contains a biconnected subgraph on at least $|L|$ vertices. Let $B$ be such a biconnected subgraph in $G$. Subgraph $B$ must be fully contained in a biconnected component of $G$. Such a biconnected component must contain at least $|B| \geq |L|$ vertices. We make a case distinction over all biconnected components in $G$ with size at least $|L|$, and prove that if $B$ is contained in them, then $\varphi(H) \supseteq Q$ for some $Q \in \mathcal{Q}$.

- $L_{2,i}$ for any $1 \leq i \leq 2n-1$: We know $|L_{2,i}| = |L|$ so $B = L$, hence $f_i(t), f_i(s) \in \varphi(L)$.
- $L_{3,i}$ for any $1 \leq i \leq 2n-1$: We know $|L_{3,i}| = |L|$ so $B = L$, hence $f_i(v), f_i(t) \in \varphi(L)$.
- $L_{1,i}$ for any $n+1 \leq i \leq 2n-1$: We know $|L_{1,i}| = |L|$ so $B = L$, hence $f_i(s), f_i(w) \in \varphi(L)$.
- $L_{1,i}$ for any $1 \leq i \leq n$: We know $|L_{1,i}| = |L|$ so $B = L$, hence $f_i(s), f_i(w) \in \varphi(L)$. If $f_i(t) \in \varphi(H)$ or $f_i(u) \in \varphi(H)$ then clearly $\mathcal{Q} \ni \{f_i(t), f_i(s)\} \subseteq \varphi(H)$ or $\mathcal{Q} \ni \{f_i(u), f_i(s), f_i(w)\} \subseteq \varphi(H)$. If $f_{n+i}(t) \in \varphi(H)$ then $\mathcal{Q} \ni \{f_{n+i}(v), f_{n+i}(t)\} \subseteq \varphi(H)$, since $f_i(w) = f_{n+i}(v)$. Suppose $\varphi(H)$ does not contain $f_i(t)$, $f_i(u)$, or $f_{n+i}(t)$, then $\varphi$ must be an $H$-model in the graph $G' := (H_{1,i} - f_i(u)) \cup (L_{2,i} - f_i(t)) \cup (L_{3,n+i} - f_i(t))$, so $H \preceq G'$. By Proposition 9 we know that $|L|$-prune$(H) \preceq |L|$-prune$(G')$. Clearly $|L|$-prune$(H) = H$. The graph $G'$ contains at least two leaf blocks that are smaller than $|L|$, namely $L_{2,i} - f_i(t)$ and $L_{3,n+i} - f_i(t)$, so $|L|$-prune$(G')$ is a subgraph of $H_{1,i} - f_i(u)$. But then $|V(|L|\text{-prune}(G'))| < |V(H)|$ so $|L|$-prune$(G')$ cannot contain an $H$-model. Contradiction.
- There can be biconnected components of size at least $|L|$ in $R_{2,i}$ for any $1 \leq i \leq 2n-1$. Suppose $f_i(t) \notin \varphi(H)$, then $\varphi$ must be an $H$-model in the graph $R_{2,i} - f_i(t)$. Clearly this is not possible since $|V(R_{2,i} - f_i(t))| < |V(H)|$, so $f_i(t) \in \varphi(H)$. If $f_i(v) \in \varphi(H)$ or $f_i(s) \in \varphi(H)$ then $\mathcal{Q} \ni \{f_i(v), f_i(t)\} \subseteq \varphi(H)$ or $\mathcal{Q} \ni \{f_i(t), f_i(s)\} \subseteq \varphi(H)$. Suppose $\varphi(H)$ does not contain $f_i(v)$ or $f_i(s)$, then $\varphi$ must be an $H$-model in the graph $G' := R_{2,i} \cup (L_{3,i} - f_i(v)) \cup (L_{2,i} - f_i(s))$, so $H \preceq G'$. By Proposition 9 we know that $|L|$-prune$(H) \preceq |L|$-prune$(G')$, so $H \preceq |L|$-prune$(G') = R_{2,i}$. This is a contradiction since $R_{2,i}$ cannot contain $H$ as a minor.
- There can be biconnected components of size at least $|L|$ in $R_{1,i}$ for any $n+1 \leq i \leq 2n-1$. Suppose $f_i(s) \notin \varphi(H)$, then $\varphi$ must be an $H$-model in the graph $R_{1,i} - f_i(s)$. As before this is not possible since $|V(R_{1,i} - f_i(s))| < |V(H)|$, so $f_i(s) \in \varphi(H)$. If $f_i(t) \in \varphi(H)$ or $f_i(w) \in \varphi(H)$ then $\mathcal{Q} \ni \{f_i(t), f_i(s)\} \subseteq \varphi(H)$ or $\mathcal{Q} \ni \{f_i(s), f_i(w)\} \subseteq \varphi(H)$. Suppose $\varphi(H)$ does not contain $f_i(t)$ or $f_i(w)$, then $\varphi$ must be an $H$-model in the graph $G' := R_{1,i} \cup (L_{2,i} - f_i(t)) \cup (L_{1,i} - f_i(w))$, so $H \preceq G'$. As before, by Proposition 9 it follows that $H \preceq R_{1,i}$ which is a contradiction.
- There can be biconnected components of size at least $|L|$ in $R_{1,i}$ for any $1 < i < n$. Suppose $f_{n+i-1}(s) \in \varphi(H)$ then $f_i(u) = f_{n+i-1}(w) \in \varphi(H)$ since any path in $G$ connecting $f_{n+i-1}(s)$ to any vertex in $R_{1,i}$ includes $f_i(u)$. So $\mathcal{Q} \ni \{f_{n+i-1}(s), f_{n+i-1}(w)\} \subseteq \varphi(H)$. Similarly if $f_i(t) \in \varphi(H)$ then $\mathcal{Q} \ni \{f_i(t), f_i(s)\} \subseteq \varphi(H)$ and if $f_{n+i}(t) \in \varphi(G)$ then $\mathcal{Q} \ni \{f_{n+i}(v), f_{n+i}(t)\} \subseteq \varphi(H)$. Suppose $\varphi(H)$ does not contain $f_{n+i-1}(s)$, $f_i(t)$, or $f_{n+i}(t)$, then $\varphi$ must be an $H$-

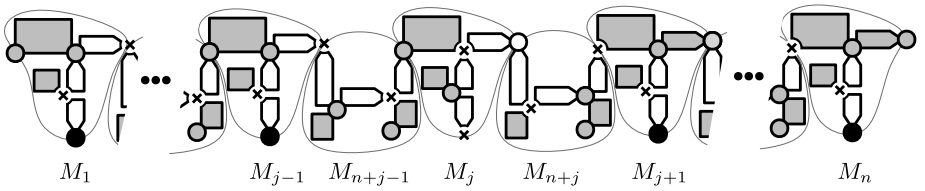



**Fig. 4.** The graph $G - X$. Vertices in $X$ that are removed from the graph are marked by a cross. Vertices in $S$ are marked black. A supergraph of $\lambda(H)$-prune$(G-X)$ is shown in gray. Note that when $|V(R)| = |V(L)|$, not all subgraphs and vertices marked gray are necessarily part of $\lambda(H)$-prune$(G-X)$. Note that the subgraphs $W_1$, $W_2$, and $W_3$ (shaded in gray) include some vertices from $G_{var}$.

model in the graph $G' := H_{1,i} \cup (L_{1,n+i-1} - f_{n+i-1}(s)) \cup (L_{2,i} - f_i(t)) \cup (L_{3,n+i} - f_{n+i}(t))$. If $\mathcal{Q} \ni \{f_i(u), f_i(s), f_i(w)\} \subseteq \varphi(H)$, then the claim holds, so suppose $\{f_i(u), f_i(s), f_i(w)\} \not\subseteq \varphi(H)$, then for some $p \in \{f_i(u), f_i(s), f_i(w)\}$ we have that $\varphi$ is an $H$-model in $G' - p$. Therefore $H \preceq G' - p$ and by Proposition 9 we know $|L|$-prune$(H) \preceq |L|$-prune$(G' - p)$, so $H \preceq |L|$-prune$(G' - p) = |L|$-prune$(H_{1,i} - p)$. However $|L|$-prune$(H_{1,i} - p)$ has at most $|V(H_{1,i} - p)| = |V(H)| - 1$ vertices, so it cannot contain an $H$-model. Contradiction.

- There can be biconnected components of size at least $|L|$ in $R_{1,1}$. As in the previous case we can assume that $\varphi(H)$ does not contain $f_1(t)$ or $f_{n+1}(t)$, so then $\varphi$ must be an $H$-model in $G' := H_{1,1} \cup (L_{2,1} - f_i(t)) \cup (L_{3,n+1} - f_{n+1}(t))$. Like in the previous case, by Proposition 9 this results in a contradiction.
- There can be biconnected components of size at least $|L|$ in $R_{1,n}$. As above we can assume that $\varphi(H)$ does not contain $f_{2n-1}(s)$ or $f_1(t)$, so then $\varphi$ must be an $H$-model in $G' := H_{1,1} \cup (L_{1,2n-1} - f_{2n-1}(s)) \cup (L_{2,1} - f_i(t))$. Again, by Proposition 9 this results in a contradiction.

This concludes the proof of Claim 12. ⌟

We now proceed to prove condition 4. of the lemma statement. Let $f_j(v) \in S$ be an arbitrary vertex in $S$, implying $1 \leq j \leq n$, and let $X$ be defined as:

$$\left( \bigcup_{1 \leq i < j} \{f_i(t), f_i(w), f_{i+n}(s)\} \right) \cup \{f_j(v), f_j(s)\} \cup \left( \bigcup_{j < i \leq n} \{f_i(t), f_i(u), f_{i+n-1}(t)\} \right).$$

In Fig. 4 the vertices in $X$ are shown in graph $G$ as a cross. Observe that $|X| = 3n - 1$ and $f_j(v) \in X$. Furthermore $X$ contains at least one element from each set in $\mathcal{Q}$, hence $G - X$ is $H$-minor-free by Claim 12. This shows parts 4.(a) and 4.(b) of the lemma statement hold. We proceed to show parts 4.(c) and 4.(d).

4.(c) Consider the graph $G' := \lambda(H)$-prune$(G - X)$. Fig. 4 shows a supergraph of $G'$ in gray for the case that $a \in V(R)$. Every connected component in $G'$ must contain a biconnected component with at least $\lambda(H) = |L|$ vertices. Consider all biconnected components in $G - X$ containing at least $|L|$ vertices. These can only be contained in the following subgraphs of $G$: $R_{2,i}$ for any $1 \leq i \leq 2n - 1$, $H_{1,i}$ for any $1 \leq i \leq n$, and $R_{1,i}$ for any $n + 1 \leq i \leq 2n - 1$. Note that any path from a vertex of one of these subgraphs to a vertex of another contains at least one vertex in $X$, hence any connected component in $G'$ contains vertices of at most one of these subgraphs. Since all other biconnected components in $G - X$ have size less than $|L|$ we know that each connected component in $|L|$-prune$(G - X)$ is a subgraph of $R_{1,i}$, $R_{2,i}$ or $H_{1,i}$ for some $i$, hence $|L|$-prune$(G - X) \precsim H$.

4.(d) Finally we show that all connected components in $G - X$ that contain a vertex of $S$ have size less than $|L|$. Since we have $f_j(v) \in X$, there is no connected component in $G - X$ containing $f_j(v)$. For all $i \neq j$ we have $f_i(t) \in X$ so the connected components in $G - X$ containing a vertex from $S$ are $L_{3,i} - f_i(t)$ for all $1 \leq i < j$ or $j < i \leq n$. These all have size $|L| - 1$ and contain exactly one vertex of $S$. □

### 3.3. Reduction for connected graphs $H$

Using the clause gadget described in Lemma 11 we give a polynomial-parameter transformation for the case where $\mathcal{F}$ contains a single, connected graph $H$.

**Lemma 13.** *For any connected graph $H$ with at least three vertices there exists a polynomial-time algorithm that, given a* CNF*-formula $\Phi$ with $k$ variables, outputs a graph $G$ and an integer $\ell$ such that all of the following are true:*

1. *there is a set $S \subseteq V(G)$ of at most $2k$ vertices such that* $\mathrm{tw}(G - S) \leq \mathrm{tw}(H)$,
2. *$G$ contains $\ell$ vertex-disjoint $H$-subgraphs,*
3. *if $\Phi$ is not satisfiable then there does not exist a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $H$-subgraph-free,*
4. *if $\Phi$ is satisfiable then there exists a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $H$-minor-free, $\lambda(H)$-prune$(G - X) \precsim H$, and* $\mathrm{tw}(G - X) \leq \mathrm{tw}(H)$.

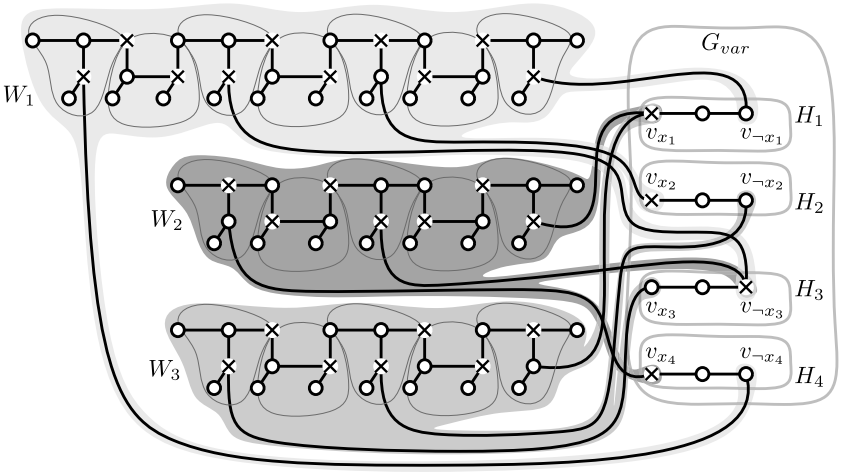



**Fig. 5.** The graph $G$ as obtained with $H = P_3$ and $\Phi = (\neg x_4 \vee \neg x_3 \vee x_2 \vee \neg x_1) \wedge (x_4 \vee \neg x_3 \vee x_1) \wedge (x_3 \vee \neg x_2 \vee x_1)$. Vertices in a solution corresponding to the satisfying assignment $x_1 = \text{True}, x_2 = \text{True}, x_3 = \text{False}, x_4 = \text{True}$ are marked with a cross.

**Proof.** Let $x_1, \ldots, x_k$ denote the variables of $\Phi$, let $C_1, \ldots, C_m$ denote the sets of literals in each clause of $\Phi$, and let $n$ denote the total number of occurrences of literals in $\Phi$, i.e., $n = \sum_{1 \leq j \leq m} |C_j|$.

**Construction.** Let $H_1, \ldots, H_k$ be copies of $H$. In each copy $H_i$ we arbitrarily label one vertex $v_{x_i}$ and another $v_{\neg x_i}$. Let $G_{var}$ be the graph obtained from the disjoint union of $H_1, \ldots, H_k$. For each clause $C_j$ of $\Phi$ we create a graph called $W_j$ and vertex set $S_j \subseteq V(W_j)$ by invoking Lemma 11 with $H$ and $|C_j|$. Let $G$ be the graph obtained from the disjoint union of $W_1, \ldots, W_m$ and $G_{var}$ where we identify the vertices in $S_j$ with the appropriate $v_{x_i}$ or $v_{\neg x_i}$ as follows: For each clause $C_j$ let $s_1, \ldots, s_{|C_j|}$ be the vertices in $S_j$ in some arbitrary order, and let $c_1, \ldots, c_{|C_j|}$ be the literals in $C_j$, then we identify $s_i$ and $v_{c_i}$ for each $1 \leq i \leq |C_j|$. Finally let $\ell = k + 3n - 2m$ and $S = \bigcup_{1 \leq i \leq k} \{v_{x_i}, v_{\neg x_i}\}$. Note that $S_j \subseteq S$ for all $1 \leq j \leq m$. This concludes the description of $G$, $\ell$, and $S$. See Fig. 5 for an example.

**Correctness.** It is easy to see they can be constructed in polynomial time. We proceed to show that all conditions in the lemma statement are met.

1. Clearly $|S| = 2k$ and since every connected component in $G - S$ is a subgraph of $H_1, \ldots, H_k$ or $W_1, \ldots, W_m$, it follows from Lemma 11(1) that $\text{tw}(G - S) \leq \text{tw}(H)$.

2. For all $1 \leq j \leq m$ we know from Lemma 11(3) that $W_j - S$ contains a packing of $3|C_j| - 2$ $H$-subgraphs. Since $W_j - S$ and $W_i - S$ are vertex-disjoint for $j \neq i$ we can combine these packings to obtain a packing in $G - S$ of $\sum_{1 \leq j \leq m}(3|C_j| - 2) = 3n - 2m$ vertex-disjoint $H$-subgraphs. Note that this packing does not contain vertices from $H_1, \ldots, H_k$, so we can add these to the packing and obtain a packing of $k + 3n - 2m = \ell$ vertex disjoint $H$-subgraphs in $G$.

3. We now show that if $\Phi$ is not satisfiable, then there does not exist a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $H$-subgraph-free. Suppose there exists such a set $X$. Since there is a packing of $\ell$ vertex-disjoint $H$-subgraphs in $G$, we know that $X$ contains exactly one vertex from each $H$-subgraph in the packing. Since $v_{x_i}$ and $v_{\neg x_i}$ belong to the same subgraph, they cannot both be contained in $X$. Consider the variable assignment where $x_i$ is assigned *true* if $v_{x_i} \in X$ or *false* otherwise. Since we assumed $\Phi$ is not satisfiable, there is at least one clause in $\Phi$ that evaluates to *false* with this variable assignment. Let $C_j$ denote such a clause. Since $C_j$ evaluates to *false*, all of its literals must be *false*, so for all variables $x_i$ that are not negated in $C_j$ we have $x_i = \textit{false}$ and therefore $v_{x_i} \notin X$. For all negated variables $x_i$ in $C_j$ we know $x_i = \textit{true}$ meaning $v_{x_i} \in X$, so $v_{\neg x_i} \notin X$. This means that $\emptyset = X \cap S_j = X \cap V(W_j) \cap V(G_{var})$, but since $G_{var}$ contains $k$ vertex-disjoint $H$-subgraphs we have $|X \cap V(G_{var})| \geq k$, so then $|X \cap (V(G_{var}) \setminus V(W_j))| \geq k$. For all $i$ there is a packing of $3|C_i| - 2$ vertex-disjoint $H$-subgraphs in $W_i - S = W_i - V(G_{var})$, so in the graph $G - V(W_j)$ there are $k + \sum_{i \neq j}(3|C_i| - 2)$ vertex-disjoint $H$-subgraphs. This means that $|X \cap (V(G) \setminus V(W_j))| \geq k + \sum_{i \neq j}(3|C_i| - 2)$, and since $|X| = k + \sum_{1 \leq i \leq m}(3|C_i| - 2)$ we know that $|X \cap V(W_j)| \leq 3|C_j| - 2$. However $W_j$ contains $3|C_j| - 1$ vertex-disjoint $H$-subgraphs, so $G - X$ cannot be $H$-subgraph-free. Contradiction.

4. Finally we show that if $\Phi$ is satisfiable then there exists a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $H$-minor-free, $\lambda(H)$-prune$(G - X) \precsim H$, and $\text{tw}(G - X) \leq \text{tw}(H)$. Since $\Phi$ is satisfiable there exists a variable assignment such that each clause contains at least one literal that is true. Consider the set $X'$ consisting of all vertices $v_{x_i}$ when $x_i$ is *true* and $v_{\neg x_i}$ when $x_i$ is *false*. Since every clause contains one literal that is true, we know for each $1 \leq j \leq m$ that $W_j$ contains at least one vertex from $X'$. So for each $1 \leq j \leq m$ we have $X' \cap S_j \neq \emptyset$. Take an arbitrary vertex $v_j \in X' \cap S_j$ and let $X_j \subseteq V(W_j)$ be the vertex set containing $v_j$ obtained from condition 4. of Lemma 11. Let $X = X' \cup \bigcup_{1 \leq j \leq m} X_j$. For all $1 \leq j \leq m$ we know $|X' \cap X_j| \geq 1$ since $v_j \in X' \cap X_j$. So $|X| \leq |X'| + \sum_{1 \leq j \leq m}(3|C_j| - 2) = k + 3n - 2m = \ell$.

By Lemma 11(4(b)) we have that $W_j - X_j$ is $H$-minor-free for all $1 \leq j \leq m$, so clearly $W_j - X$ is also $H$-minor-free. Consider an arbitrary connected component $G'$ of $G - X$. If $G'$ is also a connected component of $W_j - X$ for some $1 \leq j \leq m$, then we have that $G'$ is $H$-minor-free, $\lambda(H)$-prune$(G') \preceq H$ (by Lemma 11(4(c)), and $\mathrm{tw}(G') \leq \mathrm{tw}(W_j) \leq \mathrm{tw}(H)$. If $G'$ is not a connected component of $W_j - X$ for any $1 \leq j \leq m$, then it contains a connected component of $H_i - X$ as a subgraph, for some $1 \leq i \leq k$. When $G'$ does not contain any vertices of $S$ we know that $G'$ must be a subgraph of $H_i$, so $G'$ is $H$-minor-free, $\lambda(H)$-prune$(G') \preceq G' \preceq H$, and $\mathrm{tw}(G') \leq \mathrm{tw}(H_i) = \mathrm{tw}(H)$.

Suppose on the other hand $G'$ does contain a vertex $v \in S$. No connected component of $W_j - X_j$ contains more than one vertex from $S$ and each connected component of $G_{var}$ contains exactly two vertices of $S$, one of which is in $X$. So $v$ is the only vertex in $G'$ that is contained in $S$. Moreover, since $S$ is the only overlap between the graphs $G_{var}$ and $W_j$ for all $1 \leq j \leq m$, we have that $v$ is a cut vertex in $G'$, such that for some $1 \leq i \leq k$, each biconnected component of $G'$ is a subgraph of $H_i - X$ or $W_j - X$ for any $1 \leq j \leq m$. So each of these biconnected components of $G'$ has treewidth at most $\mathrm{tw}(H)$, hence $\mathrm{tw}(G') \leq \mathrm{tw}(H)$. Also, each biconnected component in $G'$ that is a subgraph of $W_j - X = W_j - X_j$ for some $1 \leq j \leq m$ contains a vertex from $S$ and therefore has size at most $\lambda(H) - 1$ by condition 4.(d) on the choice of $X_j$. So we have that $\lambda(H)$-prune$(G')$ is a subgraph of $H_i$, hence $\lambda(H)$-prune$(G') \preceq G' \preceq H$. Additionally since $H_i$ contains at least one vertex that is not contained in $G'$ we have $H \not\preceq \lambda(H)$-prune$(G')$. Because $H = \lambda(H)$-prune$(H)$ we can conclude by Proposition 10 that $G'$ is $H$-minor-free. Since $H$ is connected, and all connected components of $G - X$ are $H$-minor-free, $G - X$ must also be $H$-minor-free. We also know for all connected components $G'$ of $G - X$ that $\lambda(H)$-prune$(G') \preceq H$, so $\lambda(H)$-prune$(G - X) \precsim H$. Finally since $\mathrm{tw}(G') \leq \mathrm{tw}(H)$ for each connected component $G'$ of $G - X$ we have that $\mathrm{tw}(G - X) \leq \mathrm{tw}(H)$. □

The construction from Lemma 13 can directly be used to give a polynomial-parameter transformation from CNF-SAT parameterized by the number of variables. Observe that if $G - X$ is $\mathcal{F}$-minor-free, then $G - X$ is also $\mathcal{F}$-subgraph-free. Similarly, if $G - X$ contains an $H$-subgraph for all $X \subseteq V(G)$ with $|X| \leq \ell$, then $G - X$ also contains an $H$-minor. Therefore, for any *type* $\in$ {minor, subgraph} and $\mathcal{F}$ consisting of one connected graph on at least three vertices, Lemma 13 gives a polynomial-parameter transformation from CNF-SAT parameterized by the number of variables to $\mathcal{F}$-*type*-Free Deletion parameterized by deletion distance to $\min \mathrm{tw}(\mathcal{F})$.

### 3.4. Reduction for families of disconnected graphs

When $\mathcal{F}$ contains multiple graphs, each containing a connected component of at least three vertices, it is possible to select a connected component $H$ of one of the graphs in $\mathcal{F}$ such that the construction described in Lemma 13 forms the main ingredient for a polynomial-parameter transformation. This will formally be argued in the next lemma. To aid the intuition for this technical construction, we describe a simple special case. If $\mathcal{F}$ is a family of connected graphs, each on at least three vertices, and we choose $H \in \mathcal{F}$ as a $\precsim$-minimal graph in $\mathcal{F}$ with $\mathrm{tw}(H) = \min \mathrm{tw}(\mathcal{F})$, we may safely apply the construction of Lemma 13, to reduce the satisfiability of a CNF-formula $\Phi$ to $\mathcal{F}$-Minor-Free Deletion on a graph $G$. For a deletion set $X \subseteq V(G)$ corresponding to a satisfiable assignment, the graph $G - X$ is guaranteed to be $H$-minor-free by Lemma 13, and $\mathrm{tw}(G - X) \leq \mathrm{tw}(H)$. The latter implies that $G - X$ also does not contain any graphs $F \in \mathcal{F}$ with $\mathrm{tw}(F) > \min \mathrm{tw}(\mathcal{F})$ as a minor; and since $H$ is connected and $\precsim$-minimal among the treewidth-minimal graphs in $\mathcal{F}$, the fact that $G - X$ is $H$-minor-free implies that $G - X$ does not contain any other treewidth-minimal graph in $\mathcal{F}$ as a minor either. Hence our choice of $H$ ensures that $G - X$ is not only $H$-minor-free, but also $\mathcal{F}$-minor-free. The next lemma introduces a more sophisticated choice of $H$ that also works when $\mathcal{F}$ contains disconnected graphs.

**Lemma 14.** *For any fixed finite set of graphs $\mathcal{F}$, all with a connected component of at least 3 vertices, there exists a polynomial time algorithm that, given a* CNF*-formula $\Phi$ with $k$ variables, outputs a graph $G$ and integer $\ell$ such that all of the following are true:*

1. *there exists a set $S \subseteq V(G)$ of at most $k^{\mathcal{O}(1)}$ vertices such that $\mathrm{tw}(G - S) \leq \min \mathrm{tw}(\mathcal{F})$,*
2. *if $\Phi$ is not satisfiable then there does not exist a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $\mathcal{F}$-subgraph-free, and*
3. *if $\Phi$ is satisfiable then there exists a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ is $\mathcal{F}$-minor-free.*

**Proof.** Before describing the construction of $G$ and $\ell$ we define some graphs and sets based on $\mathcal{F}$.

Note that as a consequence of Observation 4, there is a graph $F \in \mathcal{F}$ that is $\precsim$-minimal with $\mathrm{tw}(F) = \min \mathrm{tw}(\mathcal{F})$. Let $\mathcal{F}_\downarrow \subseteq \mathcal{F}$ denote the set of all $\precsim$-minimal graphs in $\mathcal{F}$ that have treewidth $\min \mathrm{tw}(\mathcal{F})$. We select a $\preceq$-maximal component $H_\uparrow$ of a graph $H \in \mathcal{F}_\downarrow$ such that $\lambda(H_\uparrow) \leq \lambda(H'_\uparrow)$ for all $\preceq$-maximal components $H'_\uparrow$ of any $H' \in \mathcal{F}_\downarrow$. Note that $H_\uparrow$ contains at least 3 vertices since otherwise $H_\uparrow$ would be a minor of at least one connected component of $H$ containing at least 3 vertices, which contradicts $H_\uparrow$ being a $\preceq$-maximal component of $H$. Let $c \geq 1$ denote the number of connected components in $H$ isomorphic to $H_\uparrow$ and let $Y$ denote the set of vertices contained in these connected components, i.e., $H[Y]$ is isomorphic to $c \cdot H_\uparrow$. See Fig. 6 for an example of the choices of $\mathcal{F}_\downarrow$, $H$, $H_\uparrow$, and $c$ for a concrete $\mathcal{F}$.

**Construction.** We take the algorithm from Lemma 13 for the graph $H_\uparrow$ and apply it to $\Phi$ to construct a graph $G'$ and integer $\ell'$. Let $S' \subseteq V(G')$ be the vertex set obtained from Lemma 13(1). Let $G_1 := (2c - 1) \cdot G'$, and let the set $S$ be the union of all $2c - 1$ corresponding copies of $S'$. Take $\ell := (2c - 1) \cdot \ell'$ and let $G_2 := (\ell + 1) \cdot (H - Y)$ and $G := G_2 \cup G_1$. See Fig. 7 for a concrete example of $G$.

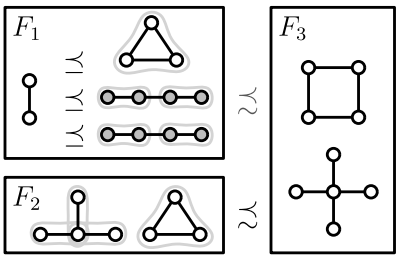



**Fig. 6.** In $\mathcal{F} = \{F_1, F_2, F_3\}$ there are two graphs ($F_1$ and $F_2$) that are $\precsim$-minimal, in this case both with treewidth $2 = \min \mathrm{tw}(\mathcal{F})$, hence $\mathcal{F}_\downarrow = \{F_1, F_2\}$. Together, the graphs in $\mathcal{F}_\downarrow$ contain five $\preceq$-maximal components. The leaf-blocks of these components are circled in gray. Observe that this leaves three candidates for $H_\uparrow$, namely those with a leaf-block of size 2. Suppose we select $H_\uparrow = P_4$, so $H = F_1$, then $c = 2$ since $H_\uparrow$ occurs twice in $H$. Vertices in $Y$ are colored gray.

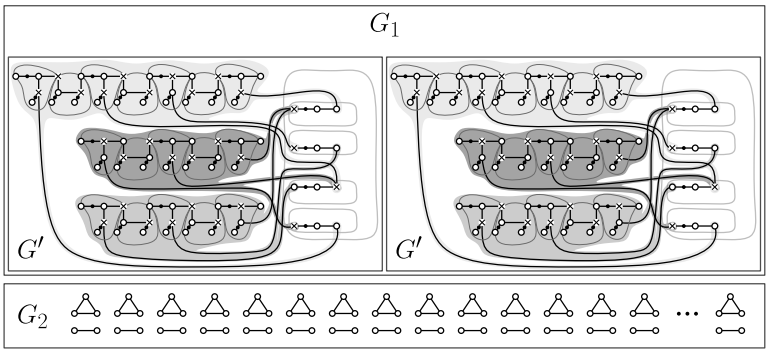



**Fig. 7.** Based on the choices of $H$, $H_\uparrow$, $c$, and $Y$ in Fig. 6 and the CNF-formula $\Phi$ as in Fig. 5 we obtain the graph $G = G_1 \cup G_2$ depicted above.

Before proving the conditions of the lemma statement hold for $G$ and $\ell$ we prove some properties of $G_2$.

**Claim 15.** *$G_2$ has the following properties: (1) $G_2 \precsim H$, (2) $\mathrm{tw}(G_2) \leq \mathrm{tw}(H)$, (3) $G_2$ is $H$-minor-free, and (4) $G_2$ is $\mathcal{F}$-minor-free.*

**Proof.** Property (1) follows directly from the construction and Property (2) follows directly from Property (1). To show Property (3), we show that $G_2$ is $H_\uparrow$-minor-free. Suppose for contradiction that $G_2$ contains $H_\uparrow$ as minor then, since $H_\uparrow$ is connected, there is a connected component $H'$ of $G_2$ that contains $H_\uparrow$ as minor. $H'$ is also a connected component of $H$. Since $H_\uparrow$ is a $\preceq$-maximal component of $H$ and $H_\uparrow \preceq H'$ we know $H' \preceq H_\uparrow$, and it follows from Observation 3 that $H'$ is isomorphic to $H_\uparrow$. This is a contradiction since $G_2$ contains only connected components of $H$ that are not isomorphic to $H_\uparrow$.

Having shown that $G_2$ is $H_\uparrow$-minor-free, Property (4) is easily shown by contradiction. Suppose $G_2$ is not $\mathcal{F}$-minor-free, then there exists a graph $B \in \mathcal{F}$ such that $B \preceq G_2$. It follows from $G_2 \precsim H$ that $B \precsim H$ and since $H$ is $\precsim$-minimal in $\mathcal{F}$ we have that $H \precsim B \preceq G_2$, but then $H_\uparrow \preceq G_2$. This is a contradiction since $G_2$ is $H_\uparrow$-minor-free. ⌟

**Correctness.** We show all conditions of the lemma statement hold for $G$ and $\ell$.

1. Observe that $|S| = (2c-1) \cdot 2k \in k^{\mathcal{O}(1)}$. By Lemma 13(1) that $\mathrm{tw}(G_1 - S) \leq \mathrm{tw}(H_\uparrow) \leq \mathrm{tw}(H)$ and since $\mathrm{tw}(G_2) \leq \mathrm{tw}(H)$ by Claim 15, we obtain $\mathrm{tw}(G - S) \leq \mathrm{tw}(H) = \min \mathrm{tw}(\mathcal{F})$.

2. Suppose $\Phi$ is not satisfiable, and take an arbitrary $X \subseteq V(G)$ of size at most $\ell$. We prove $G - X$ is not $\mathcal{F}$-subgraph-free by showing that $G - X$ contains an $H$-subgraph. First note that $G_2 - X$ contains at least one copy of $H - Y = H - c \cdot H_\uparrow$, so it remains to show that $G_1 - X$ contains $c$ vertex-disjoint $H_\uparrow$-subgraphs. Recall that $G_1$ is the disjoint union of $2c-1$ copies of $G'$. Consider the subgraph $\hat{G}_1$ of $G_1$ consisting of the $G'$-subgraphs in $G_1$ that contain at most $\ell'$ vertices of $X$. Since $\Phi$ is not satisfiable, $G'$ leaves at least one $H_\uparrow$-subgraph when $\ell'$ or fewer vertices are removed, so each $G'$-subgraph in $\hat{G}_1$ leaves at least one $H_\uparrow$-subgraph in $G_1 - X$. When $\hat{G}_1$ contains at least $c$ vertex-disjoint $G'$-subgraphs, we know that there are at least $c$ vertex-disjoint $H_\uparrow$-subgraphs in $G_1 - X$, concluding the proof. Suppose instead that $\hat{G}_1$ contains less than $c$ vertex-disjoint $G'$-subgraphs. Let $x$ be the number of $G'$-subgraphs in $G_1 - V(\hat{G}_1)$. Since $G_1$ contains $2c-1$ vertex-disjoint $G'$-subgraphs we have $x \geq c$. Each of the $G'$-subgraphs in $G_1 - V(\hat{G}_1)$ contains at least $\ell' + 1$ vertices of $X$, so $\hat{G}_1$ contains at most $\ell - x(\ell'+1)$ vertices of $X$. We also know $\hat{G}_1$ contains $\ell'((2c-1)-x)$ vertex-disjoint $H_\uparrow$-subgraphs since $G'$ contains $\ell'$ vertex-disjoint $H_\uparrow$-subgraphs (by Lemma 13(2)) and there are $(2c-1)-x$ vertex-disjoint $G'$-subgraphs in $\hat{G}_1$. We conclude that the number of vertex-disjoint $H_\uparrow$-subgraphs in $\hat{G}_1 - X$, and therefore also in $G_1 - X$, is at least

$$\ell'((2c-1)-x) - (\ell - x(\ell'+1)) = \ell'((2c-1)-x) - (\ell'(2c-1) - \ell' x - x)$$

$$= \ell'((2c-1)-x) - \ell'((2c-1)-x) + x$$
$$= x \geq c.$$

This concludes the proof of condition 2.

3. When $\Phi$ is satisfiable we know that there exists a set $X' \subseteq V(G')$ of size at most $\ell'$ such that $G' - X'$ is $H_\uparrow$-minor-free and $\lambda(H_\uparrow)$-prune$(G' - X') \precsim H_\uparrow$. So then there exists a set $X \subseteq V(G_1)$ of size at most $(2c-1)\cdot\ell' = \ell$ such that $G_1 - X$ is $H_\uparrow$-minor-free and $\lambda(H_\uparrow)$-prune$(G_1 - X) \precsim H_\uparrow$. Since $G_2$ is also $H_\uparrow$-minor-free we know that $G - X$ is $H_\uparrow$-minor-free and therefore also $H$-minor-free. We now show that $G - X$ is also $\mathcal{F}$-minor-free.

First observe the following:

$$\lambda(H_\uparrow)\text{-prune}(G_2 - X) \preceq G_2 - X \preceq G_2 \precsim H, \text{ and} \tag{1}$$
$$\lambda(H_\uparrow)\text{-prune}(G_1 - X) \precsim H_\uparrow \preceq H. \tag{2}$$

We now deduce

$$\begin{aligned}\lambda(H_\uparrow)\text{-prune}(G - X) &= \lambda(H_\uparrow)\text{-prune}((G_2 - X) \cup (G_1 - X)) \\ &= \lambda(H_\uparrow)\text{-prune}(G_2 - X) \cup \lambda(H_\uparrow)\text{-prune}(G_1 - X)) \\ &\precsim H \qquad \text{(by Equation (1) and (2))}\end{aligned}$$

Suppose $G - X$ is not $\mathcal{F}$-minor-free, then for some $H' \in \mathcal{F}$ we have $H' \preceq G - X$. There must exist a graph $B \in \mathcal{F}$ such that $B$ is $\precsim$-minimal in $\mathcal{F}$ and $B \precsim G - X$ since if $H'$ is $\precsim$-minimal in $\mathcal{F}$ then $H'$ forms such a graph $B$, and if on the other hand $H'$ is not $\precsim$-minimal in $\mathcal{F}$ then there exists a graph $H'' \in \mathcal{F}$ such that $H'' \precsim H'$ and $H''$ is $\precsim$-minimal in $\mathcal{F}$, meaning $H''$ forms such a graph $B$.

Since $B \precsim G - X$ we know by Observation 4 that $\text{tw}(B) \leq \text{tw}(G - X)$. Recall that $\text{tw}(G - X) \leq \min \text{tw}(\mathcal{F})$ so then $B \in \mathcal{F}_\downarrow$. Because of how we chose $H_\uparrow$, we know for all $\preceq$-maximal components $B_\uparrow$ of $B$ that $\lambda(B_\uparrow) \geq \lambda(H_\uparrow)$. Therefore

$$\begin{aligned}B &\precsim \lambda(H_\uparrow)\text{-prune}(B) && \text{since } B = \lambda(H_\uparrow)\text{-prune}(B) \\ &\precsim \lambda(H_\uparrow)\text{-prune}(G - X) && \text{by Proposition 10 since } B \precsim G - X \\ &\precsim H.\end{aligned}$$

Since $H$ is $\precsim$-minimal in $\mathcal{F}$, it follows that $H \precsim B$. By definition of $\precsim$ we have $H_\uparrow \preceq B \precsim G - X$. Since $H_\uparrow$ is connected we conclude $H_\uparrow \preceq G - X$. This is a contradiction since $G - X$ is $H_\uparrow$-minor-free. □

We conclude that a polynomial-parameter transformation exists for all *type* $\in \{\text{minor}, \text{subgraph}\}$ and $\mathcal{F}$ containing only graphs with a connected component on at least three vertices. Together with the fact that CNF-SAT is MK[2]-hard and does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly (cf. [26, Lemma 9]), this proves the following generalization of Theorem 1.

**Theorem 16.** *For type* $\in \{\text{minor}, \text{subgraph}\}$ *and a set* $\mathcal{F}$ *of graphs, all with a connected component of at least three vertices,* $\mathcal{F}$*-type-*Free Deletion *parameterized by vertex-deletion distance to a graph of treewidth* $\min \text{tw}(\mathcal{F})$ *is* MK[2]*-hard and does not admit a polynomial kernel unless* NP $\subseteq$ coNP/poly.

## 4. A polynomial Turing kernelization

In this section we consider the case where $\mathcal{F}$ contains a graph with no connected component of more than two vertices; or in short $\mathcal{F}$ contains a $P_3$-subgraph-free graph. This graph consists of isolated vertices and disjoint edges. Let isol$(G)$ denote the set of isolated vertices in a graph $G$, i.e., isol$(G) = \{v \in V(G) \mid \deg(v) = 0\}$. We first show that the removal of all isolated vertices from all graphs in $\mathcal{F}$ only changes the answer to $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion when the input is of constant size.

**Lemma 17.** *For type* $\in \{\text{minor}, \text{subgraph}\}$ *and any family of graphs* $\mathcal{F}$ *containing a* $P_3$*-subgraph-free graph, let* $\mathcal{F}' = \{F - \text{isol}(F) \mid F \in \mathcal{F}\}$*. For any graph* $G$*, if* $G$ *is* $\mathcal{F}$*-type-free but not* $\mathcal{F}'$*-type-free, then* $|V(G)| < \max_{F\in\mathcal{F}}(|V(F)| + 2|V(F)|^3)$.

**Proof.** We first prove the lemma for *type* = subgraph. Suppose $G$ is $\mathcal{F}$-subgraph-free but not $\mathcal{F}'$-subgraph-free. Now $G$ contains an $H'$-subgraph for some graph $H' \in \mathcal{F}'$. This subgraph consists of $|V(H')|$ vertices. Let $H \in \mathcal{F}$ be the graph for which $H' = H - \text{isol}(H)$. The graph $G$ cannot contain $|\text{isol}(H)|$ vertices in addition to the vertices in the $H'$-subgraph because otherwise $G$ trivially contains an $\mathcal{F}$-subgraph. Hence $|V(G)| < |V(H')| + |\text{isol}(H)| = |V(H)| \leq \max_{F\in\mathcal{F}} |V(F)|$.

Next, we show the lemma holds for *type* = minor. If some graph $G$ is $\mathcal{F}$-minor-free but not $\mathcal{F}'$-minor-free then for some graph $H \in \mathcal{F}$ we have $H' \preceq G$ but not $H \preceq G$ where $H' = H - \text{isol}(H)$. Let $\varphi$ be a minimal $H'$-model in $G$. The graph $G$ has less than $|V(\text{isol}(H))|$ vertices that are not in any branch set of $\varphi$, since otherwise an $H$-model could be constructed in $G$ by taking the branch sets of $\varphi$ and adding $|V(\text{isol}(H))|$ branch sets consisting of a single vertex.

The number of vertices in $G$ that are contained in a branch set of $\varphi$ can also be limited. For an arbitrary vertex $v \in V(H')$ consider a spanning tree $T$ of $G[\varphi(v)]$. If $\varphi(v)$ contains multiple vertices then for each leaf $p$ of $T$, there must be a vertex $u \in N_{H'}(v)$ and $q \in N_G(p) \cap \varphi(u)$, such that $p$ is the only vertex from $\varphi(v)$ that is adjacent to $\varphi(u)$; otherwise, removing leaf $p$ from the branch set $\phi(v)$ would yield a smaller $H'$-model in $G$. Hence there can only be $\max\{1, \deg_{H'}(v)\}$ leaves in $T$.

To give a bound on the size of each branch set consider a smallest graph $D \in \mathcal{F}'$ that is $P_3$-subgraph-free. Take $\ell = |V(D)|$ and note that $D \preceq P_\ell$. Since we know that $G$ is $\mathcal{F}'$-minor-free, $G$ must also be $P_\ell$-subgraph-free, therefore $T$ is also $P_\ell$-subgraph-free. Consider an arbitrary vertex $r$ in $T$. Since $T$ is a tree, there is exactly one path from $r$ to each leaf of $T$ and every vertex of $T$ lies on at least one path from $r$ to a leaf of $T$. Since there are no more than $\max\{1, \deg_{H'}(v)\}$ leaves in $T$ there are at most $\max\{1, \deg_{H'}(v)\}$ such paths, and all these paths contain less than $\ell$ vertices since $T$ is $P_\ell$-subgraph-free, hence in total $T$ contains less than $\deg_{H'}(v) \cdot \ell$ vertices. We can now give a bound on the total number of vertices in $G$ as follows:

$$\begin{aligned}
|V(G)| &< |\text{isol}(H)| + \sum_{v \in H - \text{isol}(H)} |\varphi(v)| \\
&\leq |\text{isol}(H)| + \sum_{v \in H - \text{isol}(H)} (\deg_{H'}(v) \cdot \ell) \\
&\leq |\text{isol}(H)| + 2 \cdot |E(H)| \cdot \ell \\
&\leq |V(H)| + 2 \cdot |V(H)|^2 \cdot |V(D)| \\
&\leq \max_{F \in \mathcal{F}} (|V(F)| + 2|V(F)|^3)
\end{aligned}$$

This concludes the proof. □

After the removal of isolated vertices in $\mathcal{F}$ to obtain $\mathcal{F}'$, we know that $\mathcal{F}'$ contains a graph consisting entirely of disjoint edges, i.e., this graph is isomorphic to $c \cdot P_2$ for some integer $c \geq 0$. If $c = 0$ then $\mathcal{F}$-*type*-free graphs have constant size and the problem is polynomial-time solvable. We proceed assuming $c \geq 1$. Let the matching number of a graph $G$, denoted as $\nu(G)$, be the size of a maximum matching in $G$. We make the following observation.

**Observation 18.** *For all $c \geq 1$, a graph $G$ is $c \cdot P_2$-subgraph-free if and only if $\nu(G) \leq c - 1$.*

We give a characterization of graphs with bounded matching number, based on an adaptation of the Tutte-Berge formula [33]. We use $\text{odd}(G)$ to denote the number of connected components in $G$ that consist of an odd number of vertices.

**Lemma 19.** *For any graph $G$ and integer $m$ we have $\nu(G) \leq m$ if and only if $V(G)$ can be partitioned into three disjoint sets $U, R, S$ such that all of the following are true:*

- *all connected components in $G[R]$ have an odd number of at least 3 vertices,*
- *$G[S]$ is independent,*
- *$N_G(S) \subseteq U$, and*
- *$|U| + \frac{1}{2}(|R| - \text{odd}(G[R])) \leq m$.*

**Proof.** Consider the Tutte-Berge formula [33] (cf. [34, Chapter 24]):

$$\nu(G) = \tfrac{1}{2} \min_{U \subseteq V(G)} (|V(G)| - \text{odd}(G - U) + |U|).$$

Suppose $\nu(G) \leq m$. It follows from the Tutte-Berge formula that there exists a $U_1 \subseteq V(G)$ such that $\frac{1}{2}(|V(G)| - \text{odd}(G - U_1) + |U_1|) = \nu(G) \leq m$. From each connected component $H$ in $G - U_1$ on an even number of vertices, select a vertex that is not a cut vertex of $H$ (any leaf of a spanning tree of $H$ suffices) and add the selected vertices to a set $U_2$. Now take $U = U_1 \cup U_2$. Note that $G - U$ contains only connected components with an odd number of vertices and $\text{odd}(G - U) = \text{odd}(G - U_1) + |U_2|$. Let $S$ be the set of isolated vertices in $G - U$ and let $R = V(G) \setminus (U \cup S)$. Observe that $U$, $R$, and $S$ satisfy the first three conditions in the lemma statement: $G[R]$ contains only connected components with an odd number of at least 3 vertices, $G[S]$ is independent, and $N_G(S) \subseteq U$. Note that this implies that $\text{odd}(G[R]) + |S| = \text{odd}(G[R \cup S]) = \text{odd}(G - U)$. The last requirement follows from the Tutte-Berge formula as follows:

$$\begin{aligned}
|U| + \tfrac{1}{2}(|R| - \text{odd}(G[R])) &= \tfrac{1}{2}(2|U| + |S| - |S| + |R| - \text{odd}(G[R])) \\
&= \tfrac{1}{2}((|U| + |S| + |R|) - (\text{odd}(G[R]) + |S|) + |U|) \\
&= \tfrac{1}{2}(|V(G)| - \text{odd}(G - U) + |U|) \\
&= \tfrac{1}{2}(|V(G)| - (\text{odd}(G - U_1) + |U_2|) + |U_1| + |U_2|) \\
&= \tfrac{1}{2}(|V(G)| - \text{odd}(G - U_1) + |U_1|) \\
&= \nu(G) \leq m
\end{aligned}$$

For the reverse direction of the proof, suppose $V(G)$ can be partitioned into disjoint sets $U, R, S$ as described in the lemma statement. A maximum matching in $G[R]$ has size at most $\frac{1}{2}(|R| - \text{odd}(G[R]))$ since at least one vertex in each odd component remains unmatched and every matching edge covers two vertices. Since $N_G(S) \subseteq U$ we know that $S$ is isolated in $G - U$, so $\nu(G - U) = \nu(G[R]) \leq \frac{1}{2}(|R| - \text{odd}(G[R]))$. Since a matching in $G$ is at most $|U|$ edges larger than a matching in $G - U$ we conclude $\nu(G) \leq |U| + \frac{1}{2}(|R| - \text{odd}(G[R])) \leq m$. □

Let us showcase how Lemma 19 can be used to attack $\mathcal{F}$-Minor-Free Deletion when $\mathcal{F}$ consists of a single graph $c \cdot P_2$, so that the problem is to find a set $X \subseteq V(G)$ of size at most $\ell$ such that $G - X$ has matching number less than $c$.

**Theorem 20.** *For any constant $c$, the $\{c \cdot P_2\}$-Minor-Free Deletion problem parameterized by the size $k$ of a feedback vertex set, can be solved in polynomial time using an oracle that answers* Vertex Cover *instances with $\mathcal{O}(k^3)$ vertices.*

**Proof.** If an instance $(G, \ell)$ admits a solution $X$, then Lemma 19 guarantees that $V(G - X)$ can be partitioned into $U, R, S$ satisfying the four conditions for $m = c - 1$. We try all relevant options for the sets $U$ and $R$ in the partition, of which there are only polynomially many since $|U| + \frac{1}{3}|R| \leq m \in \mathcal{O}(1)$.

For given sets $U, R \subseteq V(G)$, we can decide whether there is a solution $X$ of size at most $\ell$ for which $U, R$, and $S := V(G) \setminus (U \cup R \cup X)$ form the partition witnessing that $G - X$ has matching number at most $m$, as follows. If some component of $G[R]$ has an even number of vertices or less than three vertices, we reject outright. Similarly, if $|U| + \frac{1}{2}(|R| - \text{odd}(G[R])) > m$, we reject. Now, if $U$ and $R$ were guessed correctly, then Lemma 19 guarantees that the only neighbors of $R$ in the graph $G - X$ belong to $U$. Hence we infer that all vertices of $X' := N_G(R) \setminus U$ must belong to the solution $X$. Note that since $S$ is an independent set in $G - X$, the solution $X$ forms a vertex cover of $G - (U \cup R)$, so that $X'' := X \setminus X'$ is a vertex cover of $G' := G - (U \cup R \cup X')$. On the other hand, for every vertex cover $X''$ of $G'$, the graph $G - (X' \cup X'')$ will have matching number at most $m$, as witnessed by the partition. Hence the problem of finding a minimum solution $X$ whose corresponding graph $G - X$ has $U$ and $R$ as two of the classes in its witness partition, reduces to finding a minimum vertex cover of the graph $G'$. In terms of the decision problem, this means $G$ has a solution of size at most $\ell$ with $U$ and $R$ as witness partite sets, if and only if $G'$ has a vertex cover of size at most $\ell - |X'|$. Since $\text{fvs}(G') \leq \text{fvs}(G)$, we can apply the known [12] kernel for Vertex Cover parameterized by the feedback vertex number to reduce $(G', \ell - |X'|)$ to an equivalent instance with $\mathcal{O}(\text{fvs}(G)^3)$ vertices, which is queried to the oracle. If the oracle answers positively to any query, then $(G, \ell)$ has answer yes; otherwise the answer is no. □

We remark that by using the polynomial-time reduction guaranteed by NP-completeness, the queries to the oracle can be posed as instances of the original $\mathcal{F}$-Minor-Free Deletion problem, rather than Vertex Cover. The following lemma formalizes this and will be used as a black box for our general Turing kernelization.

**Lemma 21.** *Let $\mathcal{F}$ be a finite set of graphs such that each graph in $\mathcal{F}$ contains at least one edge, and let type $\in \{minor, subgraph\}$. There is a polynomial-time algorithm that, given a graph $G$ and integer $\ell$, decides whether $G$ has a vertex cover of size at most $\ell$, using an oracle that answers $\mathcal{F}$-type-Free Deletion instances with $\text{fvs}(G)^{\mathcal{O}(1)}$ vertices.*

**Proof.** For a fixed $\mathcal{F}$ and *type*, the following procedure solves the Vertex Cover instance $(G, \ell)$ in polynomial time using an oracle for $\mathcal{F}$-*type*-Free Deletion instances with $\text{fvs}(G)^{\mathcal{O}(1)}$ vertices.

1. Compute a 2-approximate feedback vertex set $S$ on $G$ in polynomial time, for example using the algorithm by Bafna et al. [35].
2. Apply the kernelization by Jansen and Bodlaender [12] for Vertex Cover parameterized by feedback vertex set to the instance $(G, \ell)$ and the approximate feedback vertex set $S$. This takes polynomial time, and results in an instance $(G_1, \ell_1)$ of Vertex Cover on $\mathcal{O}(|S|^3) \leq \mathcal{O}(\text{fvs}(G)^3)$ vertices that is equivalent to $(G, \ell)$.
3. Since every graph in $\mathcal{F}$ contains at least one edge, the $\mathcal{F}$-*type*-Free Deletion problem is NP-complete [36]. The Vertex Cover problem is also known to be NP-complete, hence there exists a polynomial-time algorithm that transforms the Vertex Cover instance $(G_1, \ell_1)$ into an equivalent $\mathcal{F}$-*type*-Free Deletion instance $(G_2, \ell_2)$. Since this algorithm runs in polynomial time and the size of its input is $\text{fvs}(G)^{\mathcal{O}(1)}$, the number of vertices in $G_2$ is upper-bounded by $\text{fvs}(G)^{\mathcal{O}(1)}$.

4. Query the instance $(G_2, \ell_2)$ of size $\text{FVS}(G)^{\mathcal{O}(1)}$ to the $\mathcal{F}$-*type*-Free Deletion oracle, and output the oracle's answer as the decision on the Vertex Cover instance $(G, \ell)$. □

We point out that in Lemma 21, the oracle that answers $\mathcal{F}$-*type*-Free Deletion instances with $\text{FVS}(G)^{\mathcal{O}(1)}$ vertices may be replaced with an oracle that answers Vertex Cover instances on $\mathcal{O}(\text{FVS}(G)^3)$ vertices, due to the application of the Vertex Cover kernelization in Step 2. Hence when using an oracle for Vertex Cover, the query size can be bounded uniformly and does not depend on $\mathcal{F}$.

We now present our general (non-adaptive) Turing kernelization for the minor-free and subgraph-free deletion problems for all families $\mathcal{F}$ containing a $P_3$-subgraph-free graph, combining three ingredients. Lemma 17 allows us to focus on families whose graphs have no isolated vertices. The guessing strategy of Theorem 20 is the second ingredient. The final ingredient is required to deal with the fact that a solution subgraph $G - X$ that is $c \cdot P_2$-minor-free for some $c \cdot P_2 \in \mathcal{F}$, may still have one of the other graphs in $\mathcal{F}$ as a minor. To cope with this issue, we show in Lemma 23 that if $G - X$ has no matching of size $c$ (i.e., $G - X$ has a vertex cover of size at most $2c$), but does contain a minor model of some graph in $\mathcal{F}$, then there is such a minor model of constant size. By employing a more expensive (but still polynomially bounded) guessing step, this allows us to complete the Turing kernelization. In the following lemmas $\text{vc}(G)$ will denote the vertex cover number and $\Delta(G)$ will denote the maximum degree of $G$.

**Proposition 22** *([10, Proposition 1]). If $G$ contains $H$ as a minor, then there is a subgraph $G^*$ of $G$ containing an $H$-minor such that $\Delta(G^*) \leq \Delta(H)$ and $|V(G^*)| \leq |V(H)| + \text{vc}(G^*) \cdot (\Delta(H) + 1)$.*

**Lemma 23.** *For any type $\in$ {minor, subgraph}, let $\mathcal{F}$ be a family of graphs, let $G$ be a graph with vertex cover $C$, and let $S = V(G - C)$. If $G$ contains an $\mathcal{F}$-type, then there exists $S' \subseteq S$ such that $G[C \cup S']$ contains an $\mathcal{F}$-type and $|S'| \leq \max_{H \in \mathcal{F}} |V(H)| + |C| \cdot (\Delta(H) + 1)$.*

**Proof.** Suppose *type* = minor, then by Proposition 22 we know that if $G$ contains $H \in \mathcal{F}$ as a minor, then there is a subgraph $G^*$ of $G$ containing an $H$-minor such that $|V(G^*)| \leq |V(H)| + \text{vc}(G^*) \cdot (\Delta(H) + 1)$. Take $S' = V(G^*) \cap S$, then $G[C \cup S'] = G[C \cup V(G^*)]$ contains an $\mathcal{F}$-minor and

$$\begin{aligned} |S'| &\leq |V(G^*)| \\ &\leq |V(H)| + \text{vc}(G^*) \cdot (\Delta(H) + 1) \\ &\leq |V(H)| + |C| \cdot (\Delta(H) + 1) \\ &\leq \max_{H \in \mathcal{F}} |V(H)| + |C| \cdot (\Delta(H) + 1). \end{aligned}$$

On the other hand, when *type* = subgraph then $G$ contains an $H$-subgraph for some $H \in \mathcal{F}$, and trivially there exists a set $X \subseteq V(G)$ of $|V(H)|$ vertices such that $G[X]$ contains an $H$-subgraph. Take $S' = X - C$ and clearly $G[C \cup S']$ contains an $H$-subgraph. □

Armed with Lemma 23 we now present the proof of the general Turing kernelization.

**Theorem 2.** *Let $\mathcal{F}$ be a finite set of graphs, such that some $H \in \mathcal{F}$ has no connected component of three or more vertices. Then $\mathcal{F}$-*Minor-Free Deletion *and $\mathcal{F}$-*Subgraph-Free Deletion *admit polynomial Turing kernels when parameterized by the vertex-deletion distance to a graph of treewidth* $\min \text{tw}(\mathcal{F})$.

**Proof.** Fix some *type* $\in$ {minor, subgraph}. First, consider input instances $(G, \ell)$ for which $|V(G)| - \ell \leq \max_{F \in \mathcal{F}}(|V(F)| + 2|V(F)|^3)$. If $|V(G)| - \ell < 0$, there is a trivial solution. Otherwise, there exists a vertex set $X$ of size at most $\ell$ such that $G - X$ is $\mathcal{F}$-*type*-free if and only if there exists a set $X'$ of size exactly $\ell$ such that $G - X'$ is $\mathcal{F}$-*type*-free, since $\mathcal{F}$-*type*-free graphs are hereditary and $X'$ be obtained by adding sufficiently many vertices to $X$. Such a set $X'$ exists if and only if there exists a vertex set $Y$ of size exactly $|V(G)| - \ell \leq \max_{F \in \mathcal{F}}(|V(F)| + 2|V(F)|^3)$ such that $G[Y]$ is $\mathcal{F}$-*type*-free. Since there are only polynomially many such vertex sets $Y$, and for each $Y$ we can check in polynomial time whether $G[Y]$ contains an $\mathcal{F}$-*type* [37], we can apply brute force to solve the instance in polynomial time.

So from now on we only consider instances $(G, \ell)$ for which $|V(G)| - \ell > \max_{F \in \mathcal{F}}(|V(F)| + 2|V(F)|^3)$. This means that for any vertex set $X$ of size at most $\ell$, the graph $G - X$ contains more than $\max_{F \in \mathcal{F}}(|V(F)| + 2|V(F)|^3)$ vertices. Take $\mathcal{F}' = \{F - \text{isol}(F) \mid F \in \mathcal{F}\}$ and we obtain from Lemma 17 that if $G - X$ is $\mathcal{F}$-*type*-free, it is also $\mathcal{F}'$-*type*-free, and clearly if $G - X$ contains an $\mathcal{F}$-*type* it also contains an $\mathcal{F}'$-*type*. Hence the $\mathcal{F}$-*type*-Free Deletion instance $(G, \ell)$ is equivalent to the $\mathcal{F}'$-*type*-Free Deletion instance $(G, \ell)$. Note that if $\mathcal{F}$ contains an edgeless graph then $\mathcal{F}'$ contains the null graph. In this case the instance is trivially false since every graph contains the null graph as a subgraph. In the rest of the algorithm we assume each graph in $\mathcal{F}'$ contains at least one edge.

Since every graph in $\mathcal{F}$ contains an edge and at least one graph in $\mathcal{F}$ has no component of three vertices or more, we have $\min \text{tw}(\mathcal{F}) = 1$. Therefore the parameter, the deletion distance to treewidth $\min \text{tw}(\mathcal{F})$, is equal to $\text{FVS}(G)$.

To complete the Turing kernelization for $\mathcal{F}$-*type*-Free Deletion, it suffices to give a polynomial-time algorithm solving $\mathcal{F}'$-*type*-Free Deletion using an oracle that can solve $\mathcal{F}$-Minor-Free Deletion instances $(G', \ell')$ for which $|V(G')| \leq \text{fvs}(G)^{\mathcal{O}(1)}$ and $\text{fvs}(G') \leq \text{fvs}(G)^{\mathcal{O}(1)}$. Note that the latter condition on $(G', \ell')$ is redundant since $\text{fvs}(G') < |V(G')|$ for any graph $G'$.

Our Turing kernelization will use the algorithm described in Lemma 21 to solve Vertex Cover instances $(G'', \ell'')$ for induced subgraphs $G''$ of $G$. This algorithm requires an oracle for $\mathcal{F}$-*type*-Free Deletion instances with $\text{fvs}(G'')^{\mathcal{O}(1)}$ vertices. Note that since $G''$ is an induced subgraph of $G$, we have $\text{fvs}(G'') \leq \text{fvs}(G)$, hence our $\mathcal{F}$-*type*-Free Deletion oracle for instances with $\text{fvs}(G)^{\mathcal{O}(1)}$ suffices. We will refer to this algorithm as `VCoracle`.

Using this subroutine, the Turing kernelization algorithm is given in Algorithm 1. The high-level idea is as follows. Let $M$ be the smallest graph in $\mathcal{F}'$ that has no component of three or more vertices, or equivalently, which is $P_3$-subgraph-free; then $M$ consists of isolated edges. The Turing kernelization first guesses the sets $U$ and $R$ as per Lemma 19 witnessing that the graph $G - X$ obtained after removing the unknown solution $X$ does not have a matching of $|E(M)|$ edges (i.e., that $G - X$ does not contain $M \in \mathcal{F}'$ as both a minor and a subgraph). Since Lemma 19 guarantees that in the graph $G - X$ we have $N_{G-X}(R) \subseteq U$, it follows that $N_G(R) \setminus U$ must belong to the unknown solution $X$ if this guess was correct. The algorithm then considers the remaining vertices $Q := V(G) \setminus (U \cup R \cup N_G(R))$ and classifies them into $2^{|U|}$ types based on their adjacency to $U$. An additional guessing step attempts to guess up to $\alpha$ (line 2) vertices of each type in $G - X$, which will be part of the set $S$ in the partition of Lemma 19, by taking them into the range of the function $f$ (line 9). The algorithm tests whether the graph $G[f(2^U) \cup U \cup R]$ is $\mathcal{F}'$-*type*-free. If not, then the guess was incorrect. If so, then for each type of which fewer than $\alpha$ vertices were guessed to remain behind in $G - X$, the algorithm collects the remaining vertices of that type in a set $Q'$ to be added to the solution $X$, and a Vertex Cover instance is formulated on the remaining vertices of $Q$. For types of which $\alpha$ vertices remained behind, no vertices have to be added to $Q'$ or the solution $X$ in this step, because using Lemma 23 it can be guaranteed that having more vertices of that type will not lead to an $\mathcal{F}'$-*type*. The algorithm returns *true* if the formulated instance of Vertex Cover has a solution that yields a set of size at most $\ell$ when combined with the vertices of $N_G(R) \setminus U$ and $Q'$.

**Algorithm 1:** Solving $\mathcal{F}'$-*type*-Free Deletion instances using `VCoracle` with $\mathcal{F}'$ containing a $P_3$-subgraph-free graph and no edgeless graphs.

**input :** A graph $G$ and an integer $\ell$
**output:** `true` if there exists a set $X$ of size at most $\ell$ such that $G - X$ is $\mathcal{F}'$-*type*-free, or `false` otherwise.

1 $m := |E(M)| - 1$ where $M$ is a smallest $P_3$-subgraph-free graph in $\mathcal{F}'$
2 $\alpha := \max_{H \in \mathcal{F}'} |V(H)| + 3m(\Delta(H) + 1)$
3 **forall** $U \subseteq V(G)$ with $|U| \leq m$ **do**
4  **forall** $R \subseteq V(G - U)$ such that
5  all connected components in $G[R]$ have an odd number of at least 3 vertices
6  and $|U| + \frac{1}{2}(|R| - \text{odd}(G[R])) \leq m$
7  **do**
8  $Q := V(G) \setminus (U \cup R \cup N_G(R))$
9  **forall** functions $f : 2^U \to 2^Q$ such that
10  $G[f(2^U)]$ is independent and ▷ Recall that $f(2^U) = \bigcup_{Y \subseteq U} f(Y)$
11  $G[f(2^U) \cup U \cup R]$ *is* $\mathcal{F}'$-*type*-free and
12  $\forall_{Y \subseteq U} |f(Y)| \leq \alpha$ and
13  $\forall_{Y \subseteq U} \forall_{v \in f(Y)} N_G(v) \cap U = Y$
14  **do**
15  $Q' := \{v \in Q \setminus f(2^U) \mid |f(N_G(v) \cap U)| < \alpha\}$
16  **if** `VCoracle`$(G[Q] - Q', \ell - |(N_G(R) \setminus U) \cup Q'|)$ **then**
17  **return** `true`
18 **return** `false`

**Soundness.** When the algorithm returns *true*, then consider the values of $U$, $R$, $Q$, $f$, and $Q'$ at the time that *true* is returned. There exists a vertex cover $X'$ of size at most $\ell - |(N_G(R) \setminus U) \cup Q'|$ in $G[Q] - Q'$. Let $X = X' \cup (N_G(R) \setminus U) \cup Q'$, which has size at most $\ell$. The set $X$ is a vertex cover in $G - (U \cup R)$, since $G - (U \cup R) - X = (G[Q] - Q') - X'$. Hence $S := V(G - (U \cup R)) \setminus X$ is an independent set, even in $G$. The sets $U, R, S, X$ form a partition of $V(G)$. See Fig. 8a for a visual representation of these sets. We will show that $X$ is a solution to $\mathcal{F}'$-*type*-Free Deletion on $G$.

Consider an arbitrary vertex $v \in S$. Note that since $N_G(R) \subseteq U \cup X$ we have $S = V(G) \setminus (U \cup R \cup X) \subseteq V(G) \setminus (U \cup R \cup N_G(R)) = Q$, so $v \in Q$. By definition of $X$ we know $Q' \subseteq X$ so $v \notin Q'$. Then by definition of $Q'$ on line 15 we observe the following:

**Observation 24.** *For all $v \in S$ we have $v \in f(2^U)$ or $|f(N_G(v) \cap U)| \geq \alpha$.*

Assume for a contradiction that $G - X = G[U \cup R \cup S]$ contains an $\mathcal{F}'$-*type*. Since $G[S]$ is independent, $U \cup R$ is a vertex cover in $G - X$, and by Lemma 23 there exists a set $S' \subseteq S$ with $|S'| \leq \max_{H \in \mathcal{F}'} |V(H)| + |U \cup R| \cdot (\Delta(H) + 1)$ such

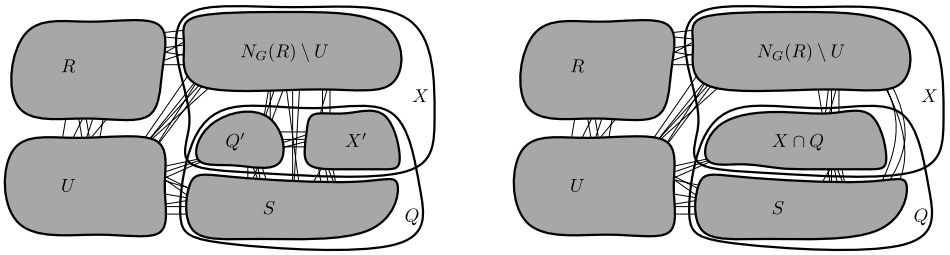



**Fig. 8.** We show two partitions of $G$. Fig. 8a shows a partition of $G$ given that Algorithm 1 returns *true*, while Fig. 8b shows a partition of $G$ given that $G - X$ is $\mathcal{F}$*-type*-free. Note that in both cases there can be no edges between $R$ and $Q$.

that $G[U \cup R \cup S']$ contains an $\mathcal{F}'$*-type*. Note that $|R| - 3\,\text{odd}(G[R]) \geq 0$ since every connected component in $G[R]$ contains at least 3 vertices, so then

$$\begin{aligned}
|S'| &\leq \max_{H \in \mathcal{F}'} |V(H)| + |U \cup R| \cdot (\Delta(H) + 1) \\
&\leq \max_{H \in \mathcal{F}'} |V(H)| + (|U| + |R| + \frac{1}{2}(|R| - 3\,\text{odd}(G[R]))) \cdot (\Delta(H) + 1) \\
&\leq \max_{H \in \mathcal{F}'} |V(H)| + 3(|U| + \frac{1}{2}(|R| - \text{odd}(G[R]))) \cdot (\Delta(H) + 1) \\
&\leq \max_{H \in \mathcal{F}'} |V(H)| + 3m(\Delta(H) + 1) \\
&= \alpha.
\end{aligned}$$

**Claim 25.** *The graph $G[U \cup R \cup S']$ is isomorphic to a subgraph of $G[U \cup R \cup f(2^U)]$.*

**Proof.** Observe that $S$ contains no neighbors of $R$, and since $G[S]$ is independent, we know for all $v \in S$ that $N_G(v) \subseteq U \cup X$ and therefore $N_{G-X}(v) = N_G(v) \cap U$. From Observation 24 it follows for all $v \in S'$ that $v \in f(2^U)$ or $|f(N_G(v) \cap U)| \geq \alpha$. In the latter case $v$ is a false twin of any vertex $u \in f(N_G(v) \cap U)$ in $G - X$ since by definition of $f$ we have $N_G(u) \cap U = N_G(v) \cap U$ for all vertices $u \in f(N_G(v) \cap U)$. We have $|S'| \leq |f(N_G(v) \cap U)|$ for all $v \in S'$, so there exists a bijection that maps all vertices $v \in S'$ to a vertex in $u \in f(2^U)$ that is a false twin of $v$ in $G - X$. Any two false twins in $G - X$ are interchangeable in $G - X$, hence $G[U \cup R \cup S']$ is isomorphic to a subgraph of $G[U \cup R \cup f(2^U)]$. ⌟

Since $f$ is chosen such that $G[U \cup R \cup f(2^U)]$ is $\mathcal{F}'$*-type*-free on line 11, Claim 25 leads to a contradiction with the fact that $G[U \cup R \cup S']$ contains an $\mathcal{F}'$*-type*. We conclude that if the algorithm returns *true* a set $X$ of size $\ell$ exists such that $G - X$ is $\mathcal{F}'$*-type*-free.

**Completeness.** Next, we consider the reverse direction. We show that the algorithm returns *true* when there exists a set $X$ of size at most $\ell$ such that $G - X$ is $\mathcal{F}'$*-type*-free. Let $m = |E(M)| - 1$ where $M$ is the smallest $P_3$-subgraph-free graph in $\mathcal{F}'$, i.e., $M$ is isomorphic to $(m+1) \cdot P_2$ since no graph in $\mathcal{F}'$ contains isolated vertices. The graph $G - X$ is $\mathcal{F}'$*-type*-free so it is also $(m+1) \cdot P_2$-subgraph-free, and by Observation 18 we know $\nu(G - X) \leq m$. Therefore by Lemma 19 there exists a partition $U', R', S$ of $V(G - X)$ such that all of the following are true:

- all connected components in $(G - X)[R'] = G[R']$ have an odd number of at least 3 vertices,
- $(G - X)[S] = G[S]$ is independent,
- $N_{G-X}(S) \subseteq U$ or equivalently $N_G(S) \subseteq U \cup X$, and
- $|U'| + \frac{1}{2}(|R'| - \text{odd}(G[R'])) \leq m$.

Clearly $U'$ and $R'$ are such that there is an iteration in the algorithm where $U = U'$ and $R = R'$. Let $Q$ be the set as defined on line 8 in this iteration, see Fig. 8b. Let $g \colon 2^U \to 2^S$ be defined as $g(Y) = \{v \in S \mid Y = N_G(v) \cap U\}$ for all $Y \subseteq U$. We define a function $f' \colon 2^U \to 2^S$ that maps any $Y \subseteq U$ to an arbitrary subset of $g(Y)$ of size $\min\{|g(Y)|, \alpha\}$. We make the following observations:

- Since $N_G(S) \subseteq U \cup X$ we have $N_G(R) \cap S = \emptyset$ so $S = S \setminus N_G(R) = V(G) \setminus (U \cup R \cup X \cup N_G(R)) \subseteq V(G) \setminus (U \cup R \cup N_G(R)) = Q$, so $f' \colon 2^U \to 2^Q$.
- $G[S]$ is independent, so $G[f'(2^U)]$ is also independent because $f'(2^U) \subseteq S$.

- $G[U \cup R \cup f'(2^U)]$ is a subgraph of $G[U \cup R \cup S] = G - X$, and since $G - X$ is $\mathcal{F}$-*type*-free, $G[U \cup R \cup f'(2^U)]$ is also $\mathcal{F}$-*type*-free.
- Clearly $\forall_{Y \subseteq U} |f'(Y)| \leq \alpha$, and
- $\forall_{Y \subseteq U} \forall_{v \in f'(Y)} N_G(v) \cap U = Y$.

Hence $f'$ satisfies all conditions stated in line 9 of the algorithm, so there is an iteration of the algorithm where $f = f'$. Let $Q'$ be the set as defined on line 15 in this iteration. We now show that there exists a vertex cover of size at most $\ell - |(N_G(R) \setminus U) \cup Q'|$ in $G[Q] - Q'$.

Since $G[S]$ is independent, $X$ is a vertex cover in $G[X \cup S] = G - (U \cup R)$. Then clearly $X \setminus (N_G(R) \setminus U)$ is a vertex cover in $G - (U \cup R \cup (N_G(R) \setminus U))$ and since $N_G(R) \setminus U \subseteq X$ we have $|X \setminus (N_G(R) \setminus U)| \leq \ell - |N_G(R) \setminus U|$. Similarly consider the set $A = (N_G(R) \setminus U) \cup Q'$. Clearly $X \setminus A$ is a vertex cover in $G - (U \cup R \cup A)$ and $|X \setminus A| \leq \ell - |A|$ if $A \subseteq X$. We will show that $A \subseteq X$. We know $N_G(R) \setminus U \subseteq X$ so it remains to be shown that $Q' \subseteq X$. Consider an arbitrary $v \in Q'$ and suppose $v \notin X$. Since $Q' \subseteq Q$ we obtain from the definition of $Q$ that $v \notin U$ and $v \notin R$, so then $v \in S$. We also note from the definition of $Q'$ that $|f(N_G(v) \cap U)| < \alpha$. Since $f = f'$ we have $|f'(N_G(v) \cap U)| < \alpha$, and from the definition of $f'$ we know that if $|f'(Y)| < \alpha$ for some $Y \subseteq U$, then $f'(Y) = g(Y)$. By definition of $g$ we have $v \in g(N_G(v) \cap U)$, so then $v \in f(N_G(v) \cap U) \subseteq f(2^U)$. This is a contradiction since $v \notin f(2^U)$ by definition of $Q'$.

Now we have shown that $X \setminus A$ is a vertex cover of size at most $\ell - |A| = \ell - |(N_G(R) \setminus U) \cup Q'|$ in $G - (U \cup R \cup A) = G[Q] - Q'$, hence the `VCoracle` should report that a vertex cover exists on line 8.

**Running time and query size.** The sets $U$ and $R$ have a maximum size of $m$ and $2m$ respectively, so there are at most $\binom{|V(G)|}{m} \leq |V(G)|^m$ and $\binom{|V(G)|}{2m} \leq |V(G)|^{2m}$ possibilities for $U$ and $R$ respectively. The function $f$ maps all subsets of $U$ to subsets of $Q$ with a maximum size of $\alpha$, so there are at most $2^{|U|} \cdot \binom{|Q|}{\alpha} \leq 2^{2m} \cdot |V(G)|^\alpha$ possible functions $f$. From the definition of $m$ and $\alpha$ on lines 1 and 2 it can be determined that $m \in \mathcal{O}(\max_{H \in \mathcal{F}} |V(H)|)$ and $\alpha \in \mathcal{O}(\max_{H \in \mathcal{F}} |V(H)|^2)$. It can now be seen that the total number of calls to `VCoracle` is at most $|V(G)|^{\mathcal{O}(\max_{H \in \mathcal{F}} |V(H)|^2)}$. Since $\mathcal{F}$ is fixed and `VCoracle` runs in polynomial time, this yields a polynomial bound on the running time of Algorithm 1.

The `VCoracle` subroutine (Lemma 21) is invoked on induced subgraphs $G''$ of $G$ which therefore have a feedback vertex number of at most $\text{FVS}(G)$. Hence Lemma 21 only queries the oracle for instances with $\text{FVS}(G'')^{\mathcal{O}(1)} \leq \text{FVS}(G)^{\mathcal{O}(1)}$ vertices. □

## 5. Conclusion

Earlier work [20,7,12,21] has shown that several $\mathcal{F}$-Minor-Free Deletion problems admit polynomial kernelizations when parameterized by the feedback vertex number. In this paper we showed that when $\mathcal{F}$ contains a forest and each graph in $\mathcal{F}$ has a connected component of at least three vertices, the $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion problems do *not* admit such a polynomial kernel unless NP $\subseteq$ coNP/poly. This lower bound generalizes to any $\mathcal{F}$ where each graph has a connected component of at least three vertices, when we consider the vertex-deletion distance to treewidth $\min \text{tw}(\mathcal{F})$ as parameter.

For all other choices of $\mathcal{F}$ we showed that a polynomial Turing kernelization exists for $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion parameterized by the feedback vertex number. The size of the Vertex Cover queries generated by the Turing kernelization does not depend on $\mathcal{F}$: the Turing kernelization can be shown to be *uniformly polynomial* (cf. [19]). However, it remains unknown whether the *running time* can be made uniformly polynomial, and whether the Turing kernelization can be improved to a traditional kernelization. Due to the large degree of the polynomial running time, the algorithm is mainly of theoretical interest.

In our paper we discussed $\mathcal{F}$-Minor-Free Deletion and $\mathcal{F}$-Subgraph-Free Deletion. We leave open the case of $\mathcal{F}$-Induced-Subgraph-Free Deletion where a vertex set $S$ is a solution for a graph $G$ if $G - S$ does not contain any graph in $\mathcal{F}$ as *induced subgraph*. Although a number of our lower bound results also apply to this problem, the Turing kernelization we present cannot easily be generalized to $\mathcal{F}$-Induced-Subgraph-Free Deletion. This is mainly because we make use of a characterization of graphs that do not have a size-$c$ matching. A similar characterization for graphs that do not have a size-$c$ induced matching is unlikely to exist since finding a maximum induced matching is NP-complete while finding a maximum matching is not.

Our results leave open the possibility that all $\mathcal{F}$-Minor-Free Deletion problems admit a polynomial kernel when parameterized by the vertex-deletion distance to a *linear forest*, i.e., a collection of paths. Resolving this question may be an interesting direction for future work.

### CRediT authorship contribution statement

**Huib Donkers:** Conceptualization, Formal analysis, Writing – original draft, Writing – review & editing. **Bart M.P. Jansen:** Conceptualization, Formal analysis, Supervision, Writing – original draft, Writing – review & editing.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


[1] H.L. Bodlaender, Kernelization: new upper and lower bound techniques, in: Proc. 4th IWPEC, 2009, pp. 17–37.
[2] F.V. Fomin, D. Lokshtanov, S. Saurabh, M. Zehavi, Kernelization: Theory of Parameterized Preprocessing, Cambridge University Press, 2019.
[3] D. Lokshtanov, N. Misra, S. Saurabh, Kernelization - preprocessing with a guarantee, in: The Multivariate Algorithmic Revolution and Beyond, 2012, pp. 129–161.
[4] A. Agrawal, D. Lokshtanov, P. Misra, S. Saurabh, M. Zehavi, Feedback vertex set inspired kernel for chordal vertex deletion, ACM Trans. Algorithms 15 (2019) 11:1–11:28, https://doi.org/10.1145/3284356.
[5] F.V. Fomin, D. Lokshtanov, N. Misra, S. Saurabh, Planar $\mathcal{F}$-deletion: approximation, kernelization and optimal FPT algorithms, in: Proc. 53rd FOCS, 2012, pp. 470–479.
[6] S. Kratsch, M. Wahlström, Compression via matroids: a randomized polynomial kernel for odd cycle transversal, ACM Trans. Algorithms 10 (2014) 20:1–20:15, https://doi.org/10.1145/2635810.
[7] Y. Iwata, Linear-time kernelization for feedback vertex set, in: Proc. 44th ICALP, in: LIPIcs, vol. 80, 2017, pp. 68:1–68:14.
[8] M. Bougeret, I. Sau, How much does a treedepth modulator help to obtain polynomial kernels beyond sparse graphs?, Algorithmica 81 (2019) 4043–4068, https://doi.org/10.1007/s00453-018-0468-8.
[9] M. Cygan, D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, S. Saurabh, On the hardness of losing width, Theory Comput. Syst. 54 (2014) 73–82, https://doi.org/10.1007/s00224-013-9480-1.
[10] F.V. Fomin, B.M.P. Jansen, M. Pilipczuk, Preprocessing subgraph and minor problems: when does a small vertex cover help?, J. Comput. Syst. Sci. 80 (2014) 468–495, https://doi.org/10.1016/j.jcss.2013.09.004.
[11] J. Guo, F. Hüffner, R. Niedermeier, A structural view on parameterizing problems: distance from triviality, in: Proc. 1st IWPEC, 2004, pp. 162–173.
[12] B.M.P. Jansen, H.L. Bodlaender, Vertex cover kernelization revisited - upper and lower bounds for a refined parameter, Theory Comput. Syst. 53 (2013) 263–299, https://doi.org/10.1007/s00224-012-9393-4.
[13] B.M.P. Jansen, S. Kratsch, Data reduction for graph coloring problems, Inf. Comput. 231 (2013) 70–88, https://doi.org/10.1016/j.ic.2013.08.005.
[14] B.M.P. Jansen, A. Pieterse, Polynomial kernels for hitting forbidden minors under structural parameterizations, Theor. Comput. Sci. 841 (2020) 124–166, https://doi.org/10.1016/j.tcs.2020.07.009.
[15] J. Uhlmann, M. Weller, Two-layer planarization parameterized by feedback edge set, Theor. Comput. Sci. 494 (2013) 99–111, https://doi.org/10.1016/j.tcs.2013.01.029.
[16] R. Niedermeier, Reflections on multivariate algorithmics and problem parameterization, in: Proc. 27th STACS, 2010, pp. 17–32.
[17] J. Baste, I. Sau, D.M. Thilikos, Optimal algorithms for hitting (topological) minors on graphs of bounded treewidth, in: Proc. 12th IPEC, in: LIPIcs, vol. 89, 2017, pp. 4:1–4:12.
[18] F.V. Fomin, D. Lokshtanov, N. Misra, G. Philip, S. Saurabh, Hitting forbidden minors: approximation and kernelization, SIAM J. Discrete Math. 30 (2016) 383–410, https://doi.org/10.1137/140997889.
[19] A.C. Giannopoulou, B.M.P. Jansen, D. Lokshtanov, S. Saurabh, Uniform kernelization complexity of hitting forbidden minors, ACM Trans. Algorithms 13 (2017) 35:1–35:35, https://doi.org/10.1145/3029051.
[20] H.L. Bodlaender, T.C. van Dijk, A cubic kernel for feedback vertex set and loop cutset, Theory Comput. Syst. 46 (2010) 566–597.
[21] S. Thomassé, A $4k^2$ kernel for feedback vertex set, ACM Trans. Algorithms 6 (2010), https://doi.org/10.1145/1721837.1721848.
[22] H.L. Bodlaender, A partial $k$-arboretum of graphs with bounded treewidth, Theor. Comput. Sci. 209 (1998) 1–45, https://doi.org/10.1016/S0304-3975(97)00228-4.
[23] N. Robertson, P.D. Seymour, Graph minors. XX. Wagner's conjecture, J. Comb. Theory, Ser. B 92 (2004) 325–357, https://doi.org/10.1016/j.jctb.2004.08.001.
[24] H. Dell, D. van Melkebeek, Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses, J. ACM 61 (2014) 23:1–23:27, https://doi.org/10.1145/2629620.
[25] L. Fortnow, R. Santhanam, Infeasibility of instance compression and succinct PCPs for NP, J. Comput. Syst. Sci. 77 (2011) 91–106, https://doi.org/10.1016/j.jcss.2010.06.007.
[26] D. Hermelin, S. Kratsch, K. Soltys, M. Wahlström, X. Wu, A completeness theory for polynomial (Turing) kernelization, Algorithmica 71 (2015) 702–730, https://doi.org/10.1007/s00453-014-9910-8.
[27] H. Fernau, Kernelization, Turing kernels, in: Encyclopedia of Algorithms, Springer, 2016, pp. 1043–1045.
[28] D. Binkele-Raible, H. Fernau, F.V. Fomin, D. Lokshtanov, S. Saurabh, Y. Villanger, Kernel(s) for problems with no kernel: on out-trees with many leaves, ACM Trans. Algorithms 8 (2012) 38, https://doi.org/10.1145/2344422.2344428.
[29] D. Lokshtanov, New methods in parameterized algorithms and complexity, Ph.D. thesis, University of Bergen, Norway, 2009.
[30] B.M.P. Jansen, Turing kernelization for finding long paths and cycles in restricted graph classes, J. Comput. Syst. Sci. 85 (2017) 18–37, https://doi.org/10.1016/j.jcss.2016.10.008.
[31] B.M.P. Jansen, M. Pilipczuk, M. Wrochna, Turing kernelization for finding long paths in graph classes excluding a topological minor, Algorithmica 81 (2019) 3936–3967, https://doi.org/10.1007/s00453-019-00614-4.
[32] M. Weller, Aspects of preprocessing applied to combinatorial graph problems, Ph.D. thesis, Technische Universität Berlin, 2013.
[33] C. Berge, Sur le couplage maximum d'un graphe, C. R. Hebd. Séances Acad. Sci. 247 (1958) 258–259.
[34] A. Schrijver, Combinatorial Optimization. Polyhedra and Efficiency, Springer, Berlin, 2003.
[35] V. Bafna, P. Berman, T. Fujito, A 2-approximation algorithm for the undirected feedback vertex set problem, SIAM J. Discrete Math. 12 (1999) 289–297, https://doi.org/10.1137/S0895480196305124.
[36] J.M. Lewis, M. Yannakakis, The node-deletion problem for hereditary properties is NP-complete, J. Comput. Syst. Sci. 20 (1980) 219–230.
[37] N. Robertson, P.D. Seymour, Graph minors. XIII. The disjoint paths problem, J. Comb. Theory, Ser. B 63 (1995) 65–110, https://doi.org/10.1006/jctb.1995.1006.