Hitting minors on bounded treewidth graphs. II. Single-exponential algorithms

Julien Baste, Ignasi Sau, Dimitrios M. Thilikos

Hitting minors on bounded treewidth graphs. II. Single-exponential algorithms[*]

Julien Baste[†][‡] Ignasi Sau[§] Dimitrios M. Thilikos[§]

Abstract

For a finite collection of graphs $\mathcal{F}$, the $\mathcal{F}$-M-Deletion (resp. $\mathcal{F}$-TM-Deletion) problem consists in, given a graph G and an integer k, decide whether there exists $S \subseteq V(G)$ with $|S| \leq k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor (resp. topological minor). We are interested in the parameterized complexity of both problems when the parameter is the treewidth of G, denoted by tw, and specifically in the cases where $\mathcal{F}$ contains a single connected planar graph H. We present algorithms running in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$, called *single-exponential*, when H is either P_3, P_4, C_4, the paw, the chair, and the banner for both $\{H\}$-M-Deletion and $\{H\}$-TM-Deletion, and when $H = K_{1,i}$, with $i \geq 1$, for $\{H\}$-TM-Deletion. Some of these algorithms use the rank-based approach introduced by Bodlaender et al. [Inform Comput, 2015]. This is the second of a series of articles on this topic, and the results given here together with other ones allow us, in particular, to provide a tight dichotomy on the complexity of $\{H\}$-M-Deletion in terms of H.

Keywords: parameterized complexity; graph minors; treewidth; hitting minors; topological minors; dynamic programming; Exponential Time Hypothesis.

[*]Emails of authors: `julien.baste@uni-ulm.de`, `ignasi.sau@lirmm.fr`, `sedthilk@thilikos.info`.

The results of this article are permanently available at `https://arxiv.org/abs/1704.07284`. Extended abstracts containing some of the results of this article appeared in the *Proc. of the 12th International Symposium on Parameterized and Exact Computation (**IPEC 2017**)* [2] and in the *Proc. of the 13th International Symposium on Parameterized and Exact Computation (**IPEC 2018**)* [3]. Work supported by French projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010).

[†]LIRMM, Université de Montpellier, Montpellier, France.
[‡]Sorbonne Université, Laboratoire d'Informatique de Paris 6, LIP6, Paris, France.
[§]AlGCo project-team, LIRMM, CNRS, Université de Montpellier, Montpellier, France.

1 Introduction

Let $\mathcal{F}$ be a finite non-empty collection of non-empty graphs. In the $\mathcal{F}$-M-Deletion (resp. $\mathcal{F}$-TM-Deletion) problem, we are given a graph G and an integer k, and the objective is to decide whether there exists a set $S \subseteq V(G)$ with $|S| \leq k$ such that $G \setminus S$ does not contain any of the graphs in $\mathcal{F}$ as a minor (resp. topological minor). These problems have a big expressive power, as instantiations of them correspond to several well-studied problems. For instance, the cases $\mathcal{F} = \{K_2\}$, $\mathcal{F} = \{K_3\}$, and $\mathcal{F} = \{K_5, K_{3,3}\}$ of $\mathcal{F}$-M-Deletion (or $\mathcal{F}$-TM-Deletion) correspond to Vertex Cover, Feedback Vertex Set, and Vertex Planarization, respectively. For the sake of readability, we use the notation $\mathcal{F}$-Deletion in statements that apply to *both* $\mathcal{F}$-M-Deletion and $\mathcal{F}$-TM-Deletion.

We are interested in the parameterized complexity of $\mathcal{F}$-Deletion when the parameter is the treewidth of the input graph. Courcelle's theorem [10] implies that $\mathcal{F}$-Deletion can be solved in time $\mathcal{O}^*(f(\mathsf{tw}))$ on graphs with treewidth at most tw, where f is some computable function[1]. Our objective is to determine, for a fixed collection $\mathcal{F}$, which is the *smallest* such function f that one can (asymptotically) hope for, subject to reasonable complexity assumptions.

This line of research has recently attracted some attention in the parameterized complexity community. For instance, Vertex Cover is easily solvable in time $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw})})$, called *single-exponential*, by standard dynamic-programming techniques, and no algorithm with running time $\mathcal{O}^*(2^{o(\mathsf{tw})})$ exists, unless the Exponential Time Hypothesis (ETH)[2] fails [17]. For Feedback Vertex Set, standard dynamic programming techniques give a running time of $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw}\cdot\log\mathsf{tw})})$, while the lower bound under the ETH [17] is again $\mathcal{O}^*(2^{o(\mathsf{tw})})$. This gap remained open for a while, until Cygan et al. [12] presented an optimal algorithm running in time $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw})})$, introducing the celebrated *Cut&Count* technique, which produces randomized algorithms. This article triggered several other techniques to obtain single-exponential *deterministic* algorithms for so-called *connectivity problems* on graphs of bounded treewidth, mostly based on algebraic tools [7,16]. We refer the reader to [4] for a more detailed discussion about related work. In particular, in this article we make use of one of the techniques presented by Bodlaender et al. [7], called *rank-based approach*. It is worth mentioning that this approach has been recently applied to dense graph classes, namely those with structured neighborhoods [6].

Our results and techniques. We provide several single-exponential algorithms when $\mathcal{F}$ contains a single connected planar graph H. Namely, we show that if $\mathcal{F} \in \{\{P_3\}, \{P_4\}, \{K_{1,i}\},$

[1]The notation $\mathcal{O}^*(\cdot)$ suppresses polynomial factors depending on the size of the input graph.

[2]The ETH states that 3-SAT on n variables cannot be solved in time $2^{o(n)}$; see [17] for more details.

$\{C_4\}$ $\{\mathsf{paw}\}, \{\mathsf{chair}\}, \{\mathsf{banner}\}\}$ (see Figure 1 for an illustration of these graphs), then $\mathcal{F}$-TM-Deletion can be solved in single-exponential time. Note that all these graphs have maximum degree at most three, except $K_{1,i}$ for $i \geq 4$, and therefore the corresponding algorithms also apply to the $\mathcal{F}$-M-Deletion problem. Indeed, for graphs H with maximum degree at most three, containing H as a minor is equivalent to containing H as a topological minor. The fact that we are not able to provide single-exponential algorithms for $\{K_{1,i}\}$-M-Deletion with $i \geq 4$ seems to be unavoidable: we prove in [5] that there is no algorithm in time $\mathcal{O}^*(2^{o(\mathsf{tw} \cdot \log \mathsf{tw})})$ for these cases, unless the ETH fails. This exhibits, to the best of our knowledge, the first difference between the computational complexity of both problems.

The single-exponential algorithms presented in this article are ad hoc, some being easier than others. All of them exploit a structural characterization of the graphs that exclude that particular graph H as a (topological) minor; cf. for instance Lemmas 2 and 9. Intuitively, the "complexity" of this characterization is what determines the difficulty of the corresponding dynamic programming algorithm, and is also what makes the difference between being solvable in single-exponential time or not.

More precisely, the algorithms for $\{P_3\}$-Deletion, $\{P_4\}$-Deletion, and $\{K_{1,i}\}$-TM-Deletion use standard (but non-trivial) dynamic programming techniques on graphs of bounded treewidth, exploiting the simple structure of graphs that do not contain these particular graphs as a topological minor (or as a subgraph, which in these cases is equivalent). The algorithms for $\{P_3\}$-Deletion and $\{K_{1,i}\}$-TM-Deletion are quite simple, while the one for $\{P_4\}$-Deletion is slightly more technical.

The algorithms for $\{C_4\}$-Deletion and $\{\mathsf{paw}\}$-Deletion are more involved, and use the rank-based approach introduced by Bodlaender et al. [7], exploiting again the structure of graphs that do not contain C_4 or the paw as a minor (cf. Lemma 5 and 7, respectively). It might seem counterintuitive that this technique works for C_4, and stops working for C_i with $i \geq 5$. A possible reason for that is that the only cycles of a C_4-minor-free graph are triangles and each triangle must be contained in a bag of a tree decomposition. This property, which is not true anymore for C_i-minor-free graphs with $i \geq 5$, permits to keep track of the structure of partial solutions with tables of small size. The algorithm for $\{\mathsf{paw}\}$-Deletion combines classical dynamic programming techniques and the rank-based approach.

Finally, the algorithms for $\{\mathsf{chair}\}$-Deletion and $\{\mathsf{banner}\}$-Deletion are a combination of the above ones, the latter one using again the rank-based approach. Given the large amount of labels that we need in the tables and the similarity with other algorithms for which we provide all the details, we only present a sketch of these two algorithms.

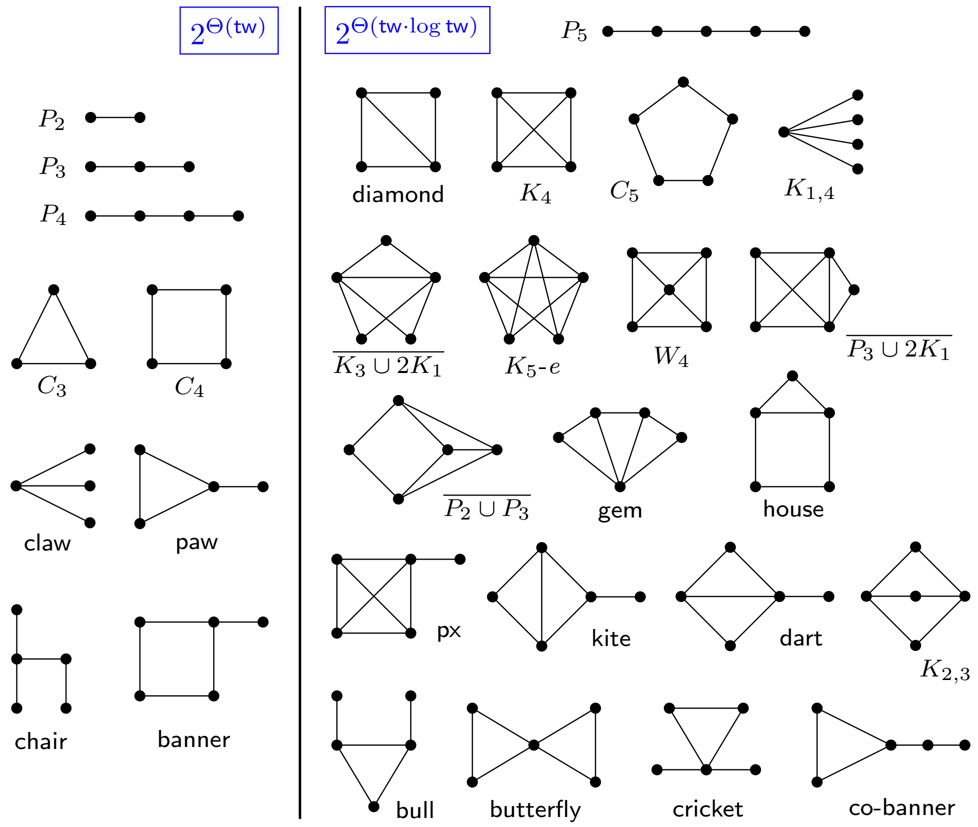
Results in other articles of the series and discussion. In the first article of this series [4], we show, among other results, that for every collection $\mathcal{F}$ containing at least one planar graph (resp. subcubic planar graph), $\mathcal{F}$-M-Deletion (resp. $\mathcal{F}$-TM-Deletion) can be solved in time $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw}\cdot\log\mathsf{tw})})$. In the third article of this series [5], we focus on lower bounds under the ETH. Namely, we prove that for any connected[3] $\mathcal{F}$, $\mathcal{F}$-Deletion cannot be solved in time $\mathcal{O}^*(2^{o(\mathsf{tw})})$, even if the input graph G is planar, and we provide superexponential lower bounds for a number of collections $\mathcal{F}$. In particular, we prove a lower bound of $\mathcal{O}^*(2^{o(\mathsf{tw}\cdot\log\mathsf{tw})})$ when $\mathcal{F}$ contains a single connected graph that is either P_5 or is not a minor of the banner, with the exception of $K_{1,i}$ for the topological minor version. These lower bounds, together with the ad hoc single-exponential algorithms given in this article and the general algorithms described in [4], cover all the cases of $\mathcal{F}$-M-Deletion where $\mathcal{F}$ contains a single connected planar graph H, yielding a dichotomy in terms of H. Namely, $\{H\}$-M-Deletion can be solved in time

- $\mathcal{O}^*(2^{\Theta(\mathsf{tw})})$, if H is a minor of the banner that is different from P_5, and
- $\mathcal{O}^*(2^{\Theta(\mathsf{tw}\cdot\log\mathsf{tw})})$, otherwise.

In the above statements, we use the Θ-notation to indicate that these algorithms are *optimal* under the ETH. This dichotomy is depicted in Figure 1, containing all connected planar graphs H with $2 \leq |V(H)| \leq 5$; note that if $|V(H)| \geq 6$, then H is not a minor of the banner, and therefore the second item above applies. Note also that K_4 and the $\mathsf{diamond}$ are the only graphs on at most four vertices for which the problem is solvable in time $\mathcal{O}^*(2^{\Theta(\mathsf{tw}\cdot\log\mathsf{tw})})$ and that the chair and the banner are the only graphs on at least five vertices for which the problem is solvable in time $\mathcal{O}^*(2^{\Theta(\mathsf{tw})})$. Note also that the cases $\mathcal{F} = \{P_2\}$ [11, 17], $\mathcal{F} = \{P_3\}$ [1, 21], and $\mathcal{F} = \{C_3\}$ [7, 12] were already known.

The crucial role payed by the banner in the complexity dichotomy may seem surprising at first sight. In fact, we realized a posteriori that the "easy" cases can be succinctly described in terms of the banner (and P_5) by taking a look at Figure 1. Nevertheless, there is some intuitive reason for which excluding the banner constitutes the horizon on the existence of single-exponential algorithms (forgetting about the "exception" $\mathcal{F} = \{P_5\}$). Namely, every connected component of a graph that excludes the banner as a (topological) minor is either a cycle (of any length) or a tree in which some vertices have been replaced by triangles; both such types of components can be maintained by a dynamic programming algorithm in single-exponential time. It appears that if the characterization of the allowed connected components is enriched in some way, such as restricting the length of the allowed cycles or forbidding certain degrees, the problem becomes inherently more difficult.

[3]A *connected* collection $\mathcal{F}$ is a collection containing only connected graphs.



$2^{\Theta(\mathsf{tw})}$
$2^{\Theta(\mathsf{tw}\cdot\log\mathsf{tw})}$
P_5
P_2
P_3
P_4
diamond
K_4
C_5
$K_{1,4}$
$\overline{K_3 \cup 2K_1}$
K_5-e
W_4
$\overline{P_3 \cup 2K_1}$
C_3
C_4
$\overline{P_2 \cup P_3}$
gem
house
claw
paw
px
kite
dart
$K_{2,3}$
chair
banner
bull
butterfly
cricket
co-banner

Figure 1: Classification of the complexity of $\{H\}$-M-Deletion for all connected simple planar graphs H with $2 \leq |V(H)| \leq 5$: for the nine graphs on the left (resp. 20 graphs on the right, and all the larger ones), the problem is solvable in time $2^{\Theta(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$ (resp. $2^{\Theta(\mathsf{tw}\cdot\log\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$). For $\{H\}$-TM-Deletion, $K_{1,4}$ should be on the left. This figure also appears in [5].

Organization of the paper. In Section 2 we give some preliminaries. We deal with P_3, P_4, $K_{1,s}$, C_4, the paw, the chair, and the banner in Sections 3, 4, 5, 6, 7, 8, and 9, respectively. We conclude in Section 10 with some questions for further research.

2 Preliminaries

In this section we provide some preliminaries to be used in the following sections.

Sets, integers, and functions. We denote by $\mathbb{N}$ the set of every non-negative integer and we set $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$. Given two integers p and q, the set $[p, q]$ refers to the set of every integer r such that $p \leq r \leq q$. Moreover, for each integer $p \geq 1$, we set $\mathbb{N}_{\geq p} = \mathbb{N} \setminus [0, p-1]$.

We use $\emptyset$ to denote the empty set and $\varnothing$ to denote the empty function, i.e., the unique subset of $\emptyset \times \emptyset$. Given a function $f : A \to B$ and a set S, we define $f|_S = \{(x, f(x)) \mid x \in S \cap A\}$. Moreover if $S \subseteq A$, we set $f(S) = \bigcup_{s \in S}\{f(s)\}$. Given a set S, we denote by $\binom{S}{2}$ the set containing every subset of S that has cardinality two.

Graphs. All the graphs that we consider in this paper are undirected, finite, and without loops or multiple edges. We use standard graph-theoretic notation, and we refer the reader to [13] for any undefined terminology. Given a graph G, we denote by $V(G)$ the set of vertices of G and by $E(G)$ the set of the edges of G. We call $|V(G)|$ *the size* of G. A graph is *the empty* graph if its size is zero. We also denote by $L(G)$ the set of the vertices of G that have degree exactly ones. If G is a tree (i.e., a connected acyclic graph) then $L(G)$ is the set of the *leaves* of G. A *vertex labeling* of G is some injection $\rho : V(G) \to \mathbb{N}^+$. Given a vertex $v \in V(G)$, we define the *neighborhood* of v as $N_G(v) = \{u \mid u \in V(G), \{u, v\} \in E(G)\}$ and the *closed neighborhood* of v as $N_G[v] = N_G(v) \cup \{v\}$. If $X \subseteq V(G)$, then we write $N_G(X) = (\bigcup_{v \in X} N_G(v)) \setminus X$. The *degree* of a vertex v in G is defined as $\mathsf{deg}_G(v) = |N_G(v)|$. A graph is called *subcubic* if all its vertices have degree at most three.

A *subgraph* $H = (V_H, E_H)$ of a graph $G = (V, E)$ is a graph such that $V_H \subseteq V(G)$ and $E_H \subseteq E(G) \cap \binom{V(H)}{2}$. If $S \subseteq V(G)$, the subgraph of G *induced by* S, denoted $G[S]$, is the graph $(S, E(G) \cap \binom{S}{2})$. We also define $G \setminus S$ to be the subgraph of G induced by $V(G) \setminus S$. If $S \subseteq E(G)$, we denote by $G \setminus S$ the graph $(V(G), E(G) \setminus S)$.

If $s, t \in V(G)$, an *(s, t)-path* of G is any connected subgraph P of G with maximum degree two and where $s, t \in L(P)$. We say that two vertices s and t are *connected* in G if G contains an (s, t)-path as a subgraph. We finally denote by $\mathcal{P}(G)$ the set of all paths of G. Given $P \in \mathcal{P}(G)$, we say that $v \in V(P)$ is an *internal vertex* of P if $\mathsf{deg}_P(v) = 2$. Given an integer i and a graph G, we say that G is i-connected if for each $\{u, v\} \in \binom{V(G)}{2}$, there exists a set $\mathcal{Q} \subseteq \mathcal{P}(G)$ of (u, v)-paths of G such that $|\mathcal{Q}| = i$ and for each $P_1, P_2 \in \mathcal{Q}$ such that $P_1 \neq P_2$, $V(P_1) \cap V(P_2) = \{u, v\}$. We denote by K_r, P_r, and C_r, the complete graph, the path, and the cycle on r vertices, respectively.

Minors and topological minors. Given two graphs H and G and two functions $\phi : V(H) \to V(G)$ and $\sigma : E(H) \to \mathcal{P}(G)$, we say that (ϕ, σ) is *a topological minor model of H in G* if

- for every $\{x, y\} \in E(H)$, $\sigma(\{x, y\})$ is an $(\phi(x), \phi(y))$-path in G and
- if P_1, P_2 are two distinct paths in $\sigma(E(H))$, then none of the internal vertices of P_1 is a vertex of P_2.

The *branch* vertices of (ϕ, σ) are the vertices in $\phi(V(E))$, while the *subdivision* vertices of (ϕ, σ) are the internal vertices of the paths in $\sigma(E(H))$.

We say that G contains H as a *topological minor*, denoted by $H \preceq_{\mathsf{tm}} G$, if there is a topological minor model (ϕ, σ) of H in G.

Given two graphs H and G and a function $\phi : V(H) \rightarrow 2^{V(G)}$, we say that ϕ is *a minor model of H in G* if

- for every $x \in V(H)$, $G[\phi(x)]$ is a connected non-empty graph and
- for every $\{x, y\} \in E(H)$, there exist $x' \in \phi(x)$ and $y' \in \phi(y)$ such that $\{x', y'\} \in E(G)$.

We say that G contains H as a *minor*, denoted by $H \preceq_{\mathsf{m}} G$, if there is a minor model ϕ of H in G.

Graph collections. Let $\mathcal{F}$ be a collection of graphs. From now on instead of "collection of graphs" we use the shortcut "collection". If $\mathcal{F}$ is a collection that is finite, non-empty, and all its graphs are non-empty, then we say that $\mathcal{F}$ is a *proper collection.* For any proper collection $\mathcal{F}$, we define $\mathsf{size}(\mathcal{F}) = \max\{\{|V(H)| \mid H \in \mathcal{F}\} \cup \{|\mathcal{F}|\}\}$. Note that if the size of $\mathcal{F}$ is bounded, then the size of the graphs in $\mathcal{F}$ is also bounded. We say that $\mathcal{F}$ is a *planar collection* (resp. *planar subcubic collection*) if it is proper and *at least one* of the graphs in $\mathcal{F}$ is planar (resp. planar and subcubic). We say that $\mathcal{F}$ is a *connected collection* if it is proper and *all* the graphs in $\mathcal{F}$ are connected. We say that $\mathcal{F}$ is an *(topological) minor antichain* if no two of its elements are comparable via the (topological) minor relation.

Let $\mathcal{F}$ be a proper collection. We extend the (topological) minor relation to $\mathcal{F}$ such that, given a graph G, $\mathcal{F} \preceq_{\mathsf{tm}} G$ (resp. $\mathcal{F} \preceq_{\mathsf{m}} G$) if and only if there exists a graph $H \in \mathcal{F}$ such that $H \preceq_{\mathsf{tm}} G$ (resp. $H \preceq_{\mathsf{m}} G$). We also denote $\mathsf{ex}_{\mathsf{tm}}(\mathcal{F}) = \{G \mid \mathcal{F} \not\preceq_{\mathsf{tm}} G\}$, i.e., $\mathsf{ex}_{\mathsf{tm}}(\mathcal{F})$ is the class of graphs that do not contain any graph in $\mathcal{F}$ as a topological minor. The set $\mathsf{ex}_{\mathsf{m}}(\mathcal{F})$ is defined analogously.

Tree decompositions. A *tree decomposition* of a graph G is a pair $\mathcal{D} = (T, \mathcal{X})$, where T is a tree and $\mathcal{X} = \{X_t \mid t \in V(T)\}$ is a collection of subsets of $V(G)$ such that:

- $\bigcup_{t \in V(T)} X_t = V(G)$,
- for every edge $\{u, v\} \in E$, there is a $t \in V(T)$ such that $\{u, v\} \subseteq X_t$, and
- for each $\{x, y, z\} \subseteq V(T)$ such that z lies on the unique path between x and y in T, $X_x \cap X_y \subseteq X_z$.

We call the vertices of T *nodes* of $\mathcal{D}$ and the sets in $\mathcal{X}$ *bags* of $\mathcal{D}$. The *width* of a tree decomposition $\mathcal{D} = (T, \mathcal{X})$ is $\max_{t \in V(T)} |X_t| - 1$. The *treewidth* of a graph G, denoted by $\mathsf{tw}(G)$, is the smallest integer w such that there exists a tree decomposition of G of width at most w. For each $t \in V(T)$, we denote by E_t the set $E(G[X_t])$.

We need to introduce nice tree decompositions, which will make the presentation of the algorithms much simpler.

Nice tree decompositions. Let $\mathcal{D} = (T, \mathcal{X})$ be a tree decomposition of G, r be a vertex of T, and $\mathcal{G} = \{G_t \mid t \in V(T)\}$ be a collection of subgraphs of G, indexed by the vertices of T. We say that the triple $(\mathcal{D}, r, \mathcal{G})$ is a *nice tree decomposition* of G if the following conditions hold:

- $X_r = \emptyset$ and $G_r = G$,
- each node of $\mathcal{D}$ has at most two children in T,
- for each leaf $t \in V(T)$, $X_t = \emptyset$ and $G_t = (\emptyset, \emptyset)$. Such t is called a *leaf node*,
- if $t \in V(T)$ has exactly one child t', then either
  - $X_t = X_{t'} \cup \{v_{\text{insert}}\}$ for some $v_{\text{insert}} \notin X_{t'}$ and $G_t = G[V(G_{t'}) \cup \{v_{\text{insert}}\}]$. The node t is called *introduce vertex* node and the vertex v_{insert} is the *insertion vertex* of X_t,
  - $X_t = X_{t'} \setminus \{v_{\text{forget}}\}$ for some $v_{\text{forget}} \in X_{t'}$ and $G_t = G_{t'}$. The node t is called *forget vertex* node and v_{forget} is the *forget vertex* of X_t.
- if $t \in V(T)$ has exactly two children t' and t'', then $X_t = X_{t'} = X_{t''}$, $E(G_{t'}) \cap E(G_{t''}) = E(G[X_t])$, and $G_t = (V(G_{t'}) \cup V(G_{t''}), E(G_{t'}) \cup E(G_{t''}))$. The node t is called a *join* node.

For each $t \in V(T)$, we denote by V_t the set $V(G_t)$. As discussed in [18], given a tree decomposition, it is possible to transform it in polynomial time to a *nice* new one of the same width. Moreover, by Bodlaender et al. [8] we can find in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n$ a tree decomposition of width $\mathcal{O}(\mathsf{tw})$ of any graph G. Hence, since in this section we focus on single-exponential algorithms, we may assume that a nice tree decomposition of width $w = \mathcal{O}(\mathsf{tw})$ is given with the input.

We also need the following simple observation that will be implicitly used in the algorithms of Sections 3, 4, and 6.

Observation 1. *Let G be a graph and h be a positive integer. Then the following assertions are equivalent.*

- *G contains P_h as a topological minor.*
- *G contains P_h as a minor.*
- *G contains P_h as a subgraph.*

Moreover, the following assertions are also equivalent.

- *G contains C_h as a topological minor.*
- *G contains C_h as a minor.*

Parameterized complexity. We refer the reader to [11, 14] for basic background on parameterized complexity, and we recall here only some very basic definitions. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. For an instance $I = (x,k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter*. A parameterized problem is *fixed-parameter tractable* (FPT) if there exists an algorithm $\mathcal{A}$, a computable function f, and a constant c such that given an instance $I = (x,k)$, $\mathcal{A}$ (called an FPT *algorithm*) correctly decides whether $I \in L$ in time bounded by $f(k) \cdot |I|^c$.

Main ingredients of the rank-based approach. We are now going to restate the tools introduced by Bodlaender et al. [7] that we need for our purposes.

Let U be a set. We define $\Pi(U)$ to be the set of all partitions of U. Given two partitions p and q of U, we define the coarsening relation $\sqsubseteq$ such that $p \sqsubseteq q$ if for each $S \in q$, there exists $S' \in p$ such that $S \subseteq S'$. $(\Pi(U), \sqsubseteq)$ defines a lattice with minimum element $\{\{U\}\}$ and maximum element $\{\{x\} \mid x \in U\}$. On this lattice, we denote by $\sqcap$ the meet operation and by $\sqcup$ the join operation.

Let $p \in \Pi(U)$. For $X \subseteq U$ we denote by $p_{\downarrow X} = \{S \cap X \mid S \in p, S \cap X \neq \emptyset\} \in \Pi(X)$ the partition obtained by removing all elements not in X from p, and analogously for $U \subseteq X$ we denote $p_{\uparrow X} = p \cup \{\{x\} \mid x \in X \setminus U\} \in \Pi(X)$ the partition obtained by adding to p a singleton for each element in $X \setminus U$. Given a subset S of U, we define the partition $U[S] = \{\{x\} \mid x \in U \setminus S\} \cup \{S\}$.

A set of *weighted partitions* is a set $\mathcal{A} \subseteq \Pi(U) \times \mathbb{N}$. We also define $\mathsf{rmc}(\mathcal{A}) = \{(p,w) \in \mathcal{A} \mid \forall (p',w') \in \mathcal{A} : p' = p \Rightarrow w \leq w'\}$.

We now define some operations on weighted partitions. Let U be a set and $\mathcal{A} \subseteq \Pi(U) \times \mathbb{N}$.

Union. Given $\mathcal{B} \subseteq \Pi(U) \times \mathbb{N}$, we define $\mathcal{A} \uplus \mathcal{B} = \mathsf{rmc}(\mathcal{A} \cup \mathcal{B})$.

Insert. Given a set X such that $X \cap U = \emptyset$, we define $\mathsf{ins}(X, \mathcal{A}) = \{(p_{\uparrow U \cup X}, w) \mid (p,w) \in \mathcal{A}\}$.

Shift. Given $w' \in \mathbb{N}$, we define $\mathsf{shft}(w', \mathcal{A}) = \{(p, w + w') \mid (p, w) \in \mathcal{A}\}$.

Glue. Given a set S, we define $\hat{U} = U \cup S$ and $\mathsf{glue}(S, \mathcal{A}) \subseteq \Pi(\hat{U}) \times \mathbb{N}$ as
$\mathsf{glue}(S, \mathcal{A}) = \mathsf{rmc}(\{(\hat{U}[S] \sqcap p_{\uparrow \hat{U}}, w \mid (p, w) \in \mathcal{A}\})$.
Given $w : \hat{U} \times \hat{U} \to \mathcal{N}$, we define $\mathsf{glue}_w(\{u, v\}, \mathcal{A}) = \mathsf{shft}(w(u, v), \mathsf{glue}(\{u, v\}, \mathcal{A}))$.

Project. Given $X \subseteq U$, we define $\overline{X} = U \setminus X$ and $\mathsf{proj}(X, \mathcal{A}) \subseteq \Pi(\overline{X}) \times \mathbb{N}$ as
$\mathsf{proj}(X, \mathcal{A}) = \mathsf{rmc}(\{(p_{\downarrow \overline{X}}, w) \mid (p, w) \in \mathcal{A}, \forall e \in X : \forall e' \in \overline{X} : p \sqsubseteq U[ee']\})$.

Join. Given a set U', $\mathcal{B} \subseteq \Pi(U') \times \mathbb{N}$, and $\hat{U} = U \cup U'$, we define $\mathsf{join}(\mathcal{A}, \mathcal{B}) \subseteq \Pi(\hat{U}) \times \mathbb{N}$ as
$\mathsf{join}(\mathcal{A}, \mathcal{B}) = \mathsf{rmc}(\{(p_{\uparrow \hat{U}} \sqcap q_{\uparrow \hat{U}}, w_1 + w_2) \mid (p, w_1) \in \mathcal{A}, (q, w_2) \in \mathcal{B}\})$.

Proposition 1 (Bodlaender et al. [7]). *Each of the operations union, insert, shift, glue, and project can be carried out in time $s \cdot |U|^{\mathcal{O}(1)}$, where s is the size of the input of the operation. Given two weighted partitions $\mathcal{A}$ and $\mathcal{B}$, $\mathsf{join}(\mathcal{A}, \mathcal{B})$ can be computed in time $|\mathcal{A}| \cdot |\mathcal{B}| \cdot |U|^{\mathcal{O}(1)}$.*

Given a weighted partition $\mathcal{A} \subseteq \Pi(U) \times \mathbb{N}$ and a partition $q \in \Pi(U)$, we define $\mathsf{opt}(q, \mathcal{A}) = \min\{w \mid (p, w) \in \mathcal{A}, p \sqcap q = \{U\}\}$. Given two weighted partitions $\mathcal{A}, \mathcal{A}' \subseteq \Pi(U) \times \mathbb{N}$, we say that $\mathcal{A}$ *represents* $\mathcal{A}'$ if for each $q \in \Pi(U)$, $\mathsf{opt}(q, \mathcal{A}) = \mathsf{opt}(q, \mathcal{A}')$.

Given a set Z and a function $f : 2^{\Pi(U) \times \mathbb{N}} \times Z \to 2^{\Pi(U) \times \mathbb{N}}$, we say that f *preserves representation* if for each two weighted partitions $\mathcal{A}, \mathcal{A}' \subseteq \Pi(U) \times \mathbb{N}$ and each $z \in Z$, it holds that if $\mathcal{A}'$ represents $\mathcal{A}$ then $f(\mathcal{A}', z)$ represents $f(\mathcal{A}, z)$.

Proposition 2 (Bodlaender et al. [7]). *The union, insert, shift, glue, project, and join operations preserve representation.*

Theorem 3 (Bodlaender et al. [7]). *There exists an algorithm reduce that, given a set of weighted partitions $\mathcal{A} \subseteq \Pi(U) \times \mathbb{N}$, outputs in time $|\mathcal{A}| \cdot 2^{(\omega-1)|U|} \cdot |U|^{\mathcal{O}(1)}$ a set of weighted partitions $\mathcal{A}' \subseteq \mathcal{A}$ such that $\mathcal{A}'$ represents $\mathcal{A}$ and $|\mathcal{A}'| \leq 2^{|U|}$, where ω denotes the matrix multiplication exponent.*

Definition of the problems. Let $\mathcal{F}$ be a proper collection. We define the parameter $\mathbf{tm}_{\mathcal{F}}$ as the function that maps graphs to non-negative integers as follows:

$$\mathbf{tm}_{\mathcal{F}}(G) \quad = \quad \min\{|S| \mid S \subseteq V(G) \wedge G \setminus S \in \mathsf{ex}_{\mathsf{tm}}(\mathcal{F})\}. \tag{1}$$

The parameter $\mathbf{m}_{\mathcal{F}}$ is defined analogously. The main objective of this paper is to study the problem of computing the parameters $\mathbf{tm}_{\mathcal{F}}$ and $\mathbf{m}_{\mathcal{F}}$ for graphs of bounded treewidth under several instantiations of the collection $\mathcal{F}$. The corresponding decision problems are formally defined as follows.

$\mathcal{F}$-TM-Deletion	$\mathcal{F}$-M-Deletion
Input: A graph G and an integer $k \in \mathbb{N}$.	**Input:** A graph G and an integer $k \in \mathbb{N}$.
Parameter: The treewidth of G.	**Parameter:** The treewidth of G.
Output: Is $\mathbf{tm}_{\mathcal{F}}(G) \leq k$?	**Output:** Is $\mathbf{m}_{\mathcal{F}}(G) \leq k$?

Note that in both above problems, we can always assume that $\mathcal{F}$ is an antichain with respect to the considered relation. Indeed, this is the case because if $\mathcal{F}$ contains two graphs H_1 and H_2 where $H_1 \preceq_{\mathsf{tm}} H_2$, then $\mathbf{tm}_{\mathcal{F}}(G) = \mathbf{tm}_{\mathcal{F}'}(G)$ where $\mathcal{F}' = \mathcal{F} \setminus \{H_2\}$ (similarly for the minor relation).

Throughout the article, we let n and tw be the number of vertices and the treewidth of the input graph of the considered problem, respectively. We will also use w to denote the width of a (nice) tree decomposition that is given together with the input graph (which, based on [8], will differ from tw by at most a factor five).

3 A single-exponential algorithm for $\{P_3\}$-TM-Deletion

It should be noted that a single-exponential algorithm for $\{P_3\}$-TM-Deletion is already known. Indeed, Tu et al. [21] presented an algorithm running in time $\mathcal{O}^*(4^{\mathsf{tw}})$, and very recently Bai et al. [1] improved it to $\mathcal{O}^*(3^{\mathsf{tw}})$. Nevertheless, for completeness we present in this section a simpler algorithm, but involving a greater constant than [1, 21].

We first give a simple structural characterization of the graphs that exclude P_3 as a topological minor.

Lemma 1. *Let G be a graph. $P_3 \not\preceq_{\mathsf{tm}} G$ if and only if each vertex of G has degree at most one.*

Proof. Let G be a graph. If G has a connected component of size at least three, then clearly it contains a P_3. This implies that, if $P_3 \not\preceq_{\mathsf{tm}} G$, then each connected component of G has size at most two and so, each vertex of G has degree at most one. Conversely, if each vertex of G has degree at most one, then, as P_3 contains a vertex of degree two, $P_3 \not\preceq_{\mathsf{tm}} G$. □

We present an algorithm using classical dynamic programming techniques over a tree decomposition of the input graph. Let G be an instance of $\{P_3\}$-TM-Deletion and let $((T, \mathcal{X}), r, \mathcal{G})$ be a nice tree decomposition of G.

We define, for each $t \in V(T)$, the set $\mathcal{I}_t = \{(S, S_0) \mid S, S_0 \subseteq X_t,\ S \cap S_0 = \emptyset\}$ and a function $\mathbf{r}_t : \mathcal{I}_t \to \mathbb{N}$ such that for each $(S, S_0) \in \mathcal{I}_t$, $\mathbf{r}(S, S_0)$ is the minimum ℓ such that there exists a set $\widehat{S} \subseteq V(G_t)$, called the *witness* of (S, S_0), that satisfies:

- $|\widehat{S}| \leq \ell$,

- $\widehat{S} \cap X_t = S$,
- $P_3 \not\preceq_{\mathsf{tm}} G_t \setminus \widehat{S}$, and
- S_0 is the set of vertices of X_t of degree 0 in $G_t \setminus S$.

Note that with this definition, $\mathbf{tm}_{\mathcal{F}}(G) = \mathbf{r}_r(\emptyset, \emptyset)$. For each $t \in V(T)$, we assume that we have already computed $\mathbf{r}_{t'}$ for each children t' of t, and we proceed to the computation of $\mathbf{r}_t$. We distinguish several cases depending on the type of node t.

Leaf. $\mathcal{I}_t = \{(\emptyset, \emptyset)\}$ and $\mathbf{r}_t(\emptyset, \emptyset) = 0$.

Introduce vertex. If v is the insertion vertex of X_t and t' is the child of t, then for each $(S, S_0) \in \mathcal{I}_t$,

$$\begin{aligned}\mathbf{r}_t(S, S_0) \quad = \quad \min\big(& \{\mathbf{r}_{t'}(S', S_0) + 1 \mid (S', S_0) \in \mathcal{I}_{t'},\ S = S' \cup \{v\}\} \\ & \cup \{\mathbf{r}_{t'}(S, S_0') \mid (S, S_0') \in \mathcal{I}_{t'}, S_0 = S_0' \cup \{v\},\ N_{G_t[X_t]}(v) \setminus S = \emptyset\} \\ & \cup \{\mathbf{r}_{t'}(S, S_0') \mid (S, S_0') \in \mathcal{I}_{t'}, S_0 = S_0' \setminus \{u\},\ u \in S_0', \\ & \qquad N_{G_t[X_t]}(v) \setminus S = \{u\}\}\ \big).\end{aligned}$$

Forget vertex. If v is the forget vertex of X_t and t' is the child of t, then for each $(S, S_0) \in \mathcal{I}_t$,

$$\mathbf{r}_t(S, S_0) \quad = \quad \min\{\mathbf{r}_{t'}(S', S_0') \mid (S', S_0') \in \mathcal{I}_{t'},\ S = S' \setminus \{v\},\ S_0 = S_0' \setminus \{v\}\}$$

Join. If t' and t'' are the children of t, then for each $(S, S_0) \in \mathcal{I}_t$,

$$\begin{aligned}\mathbf{r}(S, S_0) \quad = \quad & \min\{\mathbf{r}(S', S_0') + \mathbf{r}(S'', S_0'') - |S' \cap S''| \\ & \qquad \mid (S', S_0') \in \mathcal{I}_{t'}, (S'', S_0'') \in \mathcal{I}_{t''}, \\ & \qquad\qquad S = S' \cup S'',\ S_0 = S_0' \cap S_0'',\ X_t \setminus S \subseteq S_0' \cup S_0''\}.\end{aligned}$$

Let us analyze the running time of this algorithm. As, for each $t \in V(T)$, S and S_0 are disjoint subsets of X_t, we have that $|\mathcal{I}_t| \leq 3^{|X_t|}$. Note that if t is a leaf, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(1)$, if t is an introduce vertex or a forget vertex node, and t' is the child of t, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(|\mathcal{I}_{t'}| \cdot |X_t|)$, and if t is a join node, and t' and t'' are the two children of t, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(|\mathcal{I}_{t'}| \cdot |\mathcal{I}_{t''}| \cdot |X_t|)$.

We now show that for each $t \in V(T)$, the function $\mathbf{r}_t$ is correctly computed by the algorithm.

Leaf. This follows directly from the definition of $\mathbf{r}_t$.

Introduce vertex. Let v be the insertion vertex of X_t. As v is the insertion vertex, we have that $N_{G_t[X_t]}(v) = N_{G_t}(v)$, and so for each value we add to the set, we can find a witness of (S, S_0) of size bounded by this value.

Conversely, let $(S, S_0) \in \mathcal{I}_t$ and let $\widehat{S}$ be a witness. If $v \in S$, then $(S \setminus \{v\}, S_0) \in \mathcal{I}_{t'}$ and $\mathbf{r}(S \setminus \{v\}, S_0) \leq |\widehat{S}| - 1$, if $v \in S_0$ then $(S, S_0 \setminus \{v\}) \in \mathcal{I}_{t'}$ and $\mathbf{r}(S, S_0 \setminus \{v\}) \leq |\widehat{S}|$, and if $v \in X_t \setminus (S \cup S_0)$, then by definition v has a unique neighbor, say u, in $G_t \setminus \widehat{S}$, moreover $u \in X_t \setminus (S \cup S_0)$, v is the unique neighbor of u in $G_t \setminus \widehat{S}$, $(S, S_0 \cup \{u\}) \in \mathcal{I}_{t'}$, and $\mathbf{r}(S, S_0 \cup \{u\}) \leq |\widehat{S}|$.

Forget vertex. This also follows directly from the definition of $\mathbf{r}_t$.

Join. Let $(S', S'_0) \in \mathcal{R}_{t'}$ and let $(S'', S''_0) \in \mathcal{I}_{t''}$ with witnesses $\widehat{S}'$ and $\widehat{S}''$, respectively. If $S = S' \cup S''$ and $S'_0 \cup S''_0 = X_t \setminus S$, then the condition $X_t \setminus S \subseteq S'_0 \cup S''_0$ ensures that $G_t \setminus (\widehat{S}' \cup \widehat{S}'')$ has no vertex of degree at least two and so $\widehat{S}' \cup \widehat{S}''$ is a witness of $(S, S'_0 \cap S''_0) \in \mathcal{I}_t$ of size at most $\mathbf{r}_{t'}(S', S'_0) + \mathbf{r}_{t'}(S'', S''_0) - |S' \cap S''|$.

Conversely, let $(S, S_0) \in \mathcal{I}_t$ with witness $\widehat{S}$. If $\widehat{S}' = \widehat{S} \cap V(G_{t'})$ and $\widehat{S}'' = \widehat{S} \cap V(G_{t''})$, then by definition of $\widehat{S}$, $\widehat{S}'$ is a witness of some $(S', S'_0) \in \mathcal{I}_{t'}$, and $\widehat{S}''$ is a witness of some $(S'', S''_0) \in \mathcal{I}_{t''}$ such that $S = S' = S''$, $S'_0 \cup S''_0 = X_t \setminus S$, and $S_0 = S'_0 \cap S''_0$, and we have $\mathbf{r}_{t'}(S', S'_0) + \mathbf{r}_{t'}(S'', S''_0) - |S| \leq |\widehat{S}|$.

The following theorem summarizes the above discussion.

Theorem 4. *If a nice tree decomposition of G of width w is given, $\{P_3\}$-TM-Deletion can be solved in time $\mathcal{O}(9^w \cdot w \cdot n)$.*

4 A single-exponential algorithm for $\{P_4\}$-TM-Deletion

Similarly to what we did for $\{P_3\}$-TM-Deletion, we start with a structural definition of the graphs that exclude P_4 as a topological minor.

Lemma 2. *Let G be a graph. $P_4 \not\preceq_{\mathsf{tm}} G$ if and only if each connected component of G is either a C_3 or a star.*

Proof. First note that if each connected component of G is either a C_3 or a star, then $P_4 \not\preceq_{\mathsf{tm}} G$. Conversely, assume that $P_4 \not\preceq_{\mathsf{tm}} G$. Then each connected component of G of size at least 4 should contain at most 1 vertex of degree at least 2, hence such component is a star. On the other hand, the only graph on at most 3 vertices that is not a star is C_3. The lemma follows. $\square$

As we did for $\{P_3\}$-TM-Deletion, we present an algorithm using classical dynamic programming techniques over a tree decomposition of the input graph. Let G be an instance of $\{P_4\}$-Deletion, and let $((T, \mathcal{X}), r, \mathcal{G})$ be a nice tree decomposition of G.

We define, for each $t \in T$, the set $\mathcal{I}_t$ to be the set of each tuple $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-})$ such that $\{S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}\}$ is a partition of X_t and the function $\mathbf{r}_t : \mathcal{I}_t \to \mathbb{N}$ such that, for each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, $\mathbf{r}_t(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-})$ is the minimum ℓ such that there exists a triple $(\widehat{S}, \widehat{S}_*, \widehat{S}_{3-}) \subseteq V(G_t) \times V(G_t) \times V(G_t)$, called the *witness* of $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-})$, which satisfies the following properties:

- $\widehat{S}$, $\widehat{S}_*$, and $\widehat{S}_{3-}$ are pairwise disjoint,
- $\widehat{S} \cap X_t = S$, $\widehat{S}_* \cap X_t = S_*$, and $\widehat{S}_{3-} \cap X_t = S_{3-}$,
- $|\widehat{S}| \leq \ell$,
- $P_4 \not\preceq_{\mathsf{tm}} G_t \setminus \widehat{S}$,
- S_{1+} is a set of vertices of degree 0 in $G_t \setminus \widehat{S}$,
- each vertex of S_{1-} has a unique neighbor in $G_t \setminus \widehat{S}$ and this neighbor is in $\widehat{S}_*$,
- each connected component of $G_t[\widehat{S}_{3-}]$ is a C_3,
- there is no edge in $G_t \setminus \widehat{S}$ between a vertex of $\widehat{S}_{3-}$ and a vertex of $V(G_t) \setminus (\widehat{S} \cup \widehat{S}_{3-})$,
- there is no edge in $G_t \setminus \widehat{S}$ between a vertex of S_{3+} and a vertex of $V(G_t) \setminus (\widehat{S} \cup S_{3+})$, and
- there is no edge in $G_t \setminus \widehat{S}$ between two vertices of S_*.

Intuitively, $\widehat{S}$ corresponds to a partial solution in G_t. Note that, by Lemma 2, each component of $G_t \setminus \widehat{S}$ must be either a star or a C_3. With this in mind, $\widehat{S}_*$ is the set of vertices that are centers of a star in $G_t \setminus \widehat{S}$, S_{1+} is the set of leaves of a star that are not yet connected to a vertex of $\widehat{S}_*$, S_{1-} is the set of leaves of a star that are already connected to a vertex of $\widehat{S}_*$, $\widehat{S}_{3-}$ is the set of vertices that induce C_3's in G_t, and S_{3+} is a set of vertices that will induce C_3's when further edges will appear.

Note that with this definition, $\mathbf{tm}_{\mathcal{F}}(G) = \mathbf{r}_r(\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)$. For each $t \in V(T)$, we assume that we have already computed $\mathbf{r}_{t'}$ for each children t' of t, and we proceed to the computation of $\mathbf{r}_t$. We distinguish several cases depending on the type of node t.

Leaf. $\mathcal{I}_t = \{(\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset)\}$ and $\mathbf{r}_t(\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset) = 0$.

Introduce vertex. If v is the insertion vertex of X_t and t' is the child of t, then, for each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$,

$$\begin{aligned}
\mathbf{r}_t(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \quad = \quad & \min \Big(\ \{\mathbf{r}_{t'}(S', S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) + 1 \\
& \quad \mid (S', S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{R}_{t'},\ S = S' \cup \{v\}\} \\
\cup \quad & \{\mathbf{r}_{t'}(S, S'_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \\
& \quad \mid (S, S'_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{R}_{t'}, \\
& \quad\quad S_{1+} = S'_{1+} \cup \{v\},\ N_{G_t[X_t \setminus S]}(v) = \emptyset\} \\
\cup \quad & \{\mathbf{r}_{t'}(S, S_{1+}, S'_{1-}, S_*, S_{3+}, S_{3-}) \\
& \quad \mid (S, S_{1+}, S'_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{R}_{t'}, \\
& \quad\quad S_{1-} = S'_{1-} \cup \{v\},\ z \in S_*,\ N_{G_t[X_t \setminus S]}(v) = \{z\}\} \\
\cup \quad & \{\mathbf{r}_{t'}(S, S'_{1+}, S'_{1-}, S'_*, S_{3+}, S_{3-}) \\
& \quad \mid (S, S'_{1+}, S'_{1-}, S'_*, S_{3+}, S_{3-}) \in \mathcal{R}_{t'}, \\
& \quad\quad S_* = S'_* \cup \{v\},\ N_{G_t[X_t \setminus S]}(v) \subseteq S'_{1+}, \\
& \quad\quad S_{1+} = S'_{1+} \setminus N_{G_t[X_t \setminus S]}(v),\ S_{1-} = S'_{1-} \cup N_{G_t[X_t \setminus S]}(v)\} \\
\cup \quad & \{\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_*, S'_{3+}, S_{3-}) \\
& \quad \mid (S, S_{1+}, S_{1-}, S_*, S'_{3+}, S_{3-}) \in \mathcal{R}_{t'}, \\
& \quad\quad S_{3+} = S'_{3+} \cup \{v\}, \\
& \quad\quad [N_{G_t[X_t \setminus S]}(v) = \emptyset] \text{ or} \\
& \quad\quad [z \in S'_{3+},\ N_{G_t[X_t \setminus S]}(v) = \{z\},\ N_{G_t[X_t \setminus S]}(z) = \{v\}]\} \\
\cup \quad & \{\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_*, S'_{3+}, S'_{3-}) \\
& \quad \mid (S, S_{1+}, S_{1-}, S_*, S'_{3+}, S_{3-}) \in \mathcal{R}_{t'}, \\
& \quad\quad S_{3+} = S'_{3+} \setminus \{z, z'\},\ S_{3-} = S'_{3-} \cup \{z, z', v\}, \\
& \quad\quad z, z' \in S'_{3+},\ N_{G_t[X_t \setminus S]}(v) = \{z, z'\}, \\
& \quad\quad N_{G_t[X_t \setminus S]}(z) = \{v, z'\},\ N_{G_t[X_t \setminus S]}(z') = \{v, z\}\} \ \Big).
\end{aligned}$$

Forget vertex. If v is the forget vertex of X_t and t' is the child of t, then, for each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$,

$$\begin{aligned}\mathbf{r}_t(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \;=\; & \min\{\mathbf{r}_{t'}(S', S_{1+}, S'_{1-}, S'_*, S_{3+}, S'_{3-}) \\ & \mid (S', S_{1+}, S'_{1-}, S'_*, S_{3+}, S'_{3-}) \in \mathcal{I}_{t'}, \\ & \;S = S' \setminus \{v\},\; S_{1-} = S'_{1-} \setminus \{v\}, \\ & \;S_* = S'_* \setminus \{v\},\; S_{3-} = S'_{3-} \setminus \{v\}\}.\end{aligned}$$

Join. If t' and t'' are the children of t, then for each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, $\mathbf{r}_t(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-})$ is

$$\begin{aligned}\min\{&\mathbf{r}_{t'}(S, S'_{1+}, S'_{1-}, S_*, S'_{3+}, S'_{3-}) + \mathbf{r}_{t'}(S, S''_{1+}, S''_{1-}, S_*, S''_{3+}, S''_{3-}) - |S| \\ & \mid (S, S'_{1+}, S'_{1-}, S_*, S'_{3+}, S'_{3-}) \in \mathcal{I}_{t'},\; (S, S''_{1+}, S''_{1-}, S_*, S''_{3+}, S''_{3-}) \in \mathcal{I}_{t''}, \\ & (S'_{1+} \cup S'_{1-}) \cap (S''_{3+} \cup S''_{3-}) = (S''_{1+} \cup S''_{1-}) \cap (S'_{3+} \cup S'_{3-}) = \emptyset, \\ & \forall v \in S'_{1-} \cap S''_{1-},\; \exists z \in S_* : N_{G_t[X_t \setminus S]}(v) = \{z\}, \\ & S_{1-} = (S'_{1-} \cup S''_{1-}),\; S_{1+} = S'_{1+} \cap S''_{1+}, \\ & \forall v \in S'_{3-} \cap S''_{3-}, \exists z, z' \in S'_{3-} \cap S''_{3-} : v, z, z' \text{ induce a } C_3 \text{ in } G_t[X_t \setminus S], \\ & S_{3-} = (S'_{3-} \cup S''_{3-}),\; S_{3+} = S'_{3+} \cap S''_{3+}\}.\end{aligned}$$

Let us analyze the running time of this algorithm. As, for each $t \in V(T)$, S, S_{1+}, S_{1-}, S_*, S_{3+}, and S_{3-} form a partition of X_t, we have that $|\mathcal{I}_t| \leq 6^{|X_t|}$. Note that if t is a leaf, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(1)$, if t is an introduce vertex or a forget vertex node, and t' is the child of t, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(|\mathcal{I}_{t'}| \cdot |X_t|)$, and if t is a join node, and t' and t'' are the two children of t, then $\mathbf{r}_t$ can be computed in time $\mathcal{O}(|\mathcal{I}_{t'}| \cdot |\mathcal{I}_{t''}| \cdot |X_t|)$.

We now show that for each $t \in V(T)$, $\mathbf{r}_t$ is correctly computed by the algorithm. For each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, it can be easily checked that each value ℓ we compute respects, $\mathbf{r}_t(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \leq \ell$. Conversely, we now argue that for each $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, the computed value ℓ is such that each witness $(\widehat{S}, \widehat{S}_*, \widehat{S}_{3-})$ of $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-})$ satisfies $\ell \leq |\widehat{S}|$. We again distinguish the type of node t.

Leaf. This follows directly from the definition of $\mathbf{r}_t$.

Introduce vertex. Let v be the insertion vertex of X_t, let $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{R}_t$, and let $(\widehat{S}, \widehat{S}_*, \widehat{S}_{3-})$ be a witness.

- If $v \in S$, then $(S \setminus \{v\}, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S \setminus \{v\}, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \leq |\widehat{S}| - 1$.

- If $v \in S_{1+}$, then v is of degree 0 in $G_t \backslash \widehat{S}$, hence $(S, S_{1+} \backslash \{v\}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+} \setminus \{v\}, S_{1-}, S_*, S_{3+}, S_{3-}) \leq |\widehat{S}|$.
- If $v \in S_{1-}$, then v has a unique neighbor that is in $\widehat{S}_*$. As v is the insertion vertex of X_t, it implies that $N_{G_t}(v) \subseteq S_*$, and so $(S, S_{1+}, S_{1-} \setminus \{v\}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-} \setminus \{v\}, S_*, S_{3+}, S_{3-}) \leq |\widehat{S}|$.
- If $v \in S_*$, then every neighbor of v is in S_{1-} and has degree 1 in $G_t \setminus \widehat{S}$. Thus, $(S, S_{1+} \cup N_{G_t[X_t \setminus S]}(v), S_{1-} \setminus N_{G_t[X_t \setminus S]}(v), S_* \setminus \{v\}, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+} \cup N_{G_t[X_t \setminus S]}(v), S_{1-} \setminus N_{G_t[X_t \setminus S]}(v), S_* \setminus \{v\}, S_{3+}, S_{3-}) \leq |\widehat{S}|$.
- If $v \in S_{3+}$, then $(S, S_{1+}, S_{1-}, S_*, S_{3+} \setminus \{v\}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_*, S_{3+} \setminus \{v\}, S_{3-}) \leq |\widehat{S}|$.
- If $v \in S_{3-}$, then there exist z and z' in S_{3-} such that $\{v, z, z'\}$ induce a C_3 in $G_t \setminus \widehat{S}$ and there is no edge in $G_t \setminus \widehat{S}$ between a vertex of $\{v, z, z'\}$ and a vertex of $V(G_t \setminus \widehat{S}) \setminus \{x, z, z'\}$. So $(S, S_{1+}, S_{1-}, S_*, S_{3+} \cup \{z, z'\}, S_{3-} \setminus \{x, z, z'\}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_*, S_{3+} \cup \{z, z'\}, S_{3-} \setminus \{x, z, z'\}) \leq |\widehat{S}|$.

Forget vertex. Let v be the forget vertex of X_t, let $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, and let $(\widehat{S}, \widehat{S}_*, \widehat{S}_{3-})$ be a witness. If v has degree 0 in $G_t \setminus \widehat{S}$, then $(S, S_{1+}, S_{1-}, S_* \cup \{v\}, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$ and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_* \cup \{v\}, S_{3+}, S_{3-}) \leq |\widehat{S}|$. If v has degree at least 1 in $G_t \setminus \widehat{S}$, then $N_{G_t \setminus \widehat{S}}(v) \cap S_{3+} = \emptyset$, as otherwise there would be an edge in $G_t \setminus \widehat{S}$ between a vertex of S_{3+} and a vertex of $V(G_t) \setminus (\widehat{S} \cup S_{3+})$. So, one of the following case occurs:

- $v \in \widehat{S}$, $(S \cup \{v\}, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$, and $\mathbf{r}_{t'}(S \cup \{v\}, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \leq |\widehat{S}|$,
- $v \in \widehat{S}_*$, $(S, S_{1+}, S_{1-}, S_* \cup \{v\}, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$, and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_* \cup \{v\}, S_{3+}, S_{3-}) \leq |\widehat{S}|$,
- $N_{G_t \setminus \widehat{S}}(v) \subseteq \widehat{S}_*$, $(S, S_{1+}, S_{1-} \cup \{v\}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_{t'}$, and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-} \cup \{v\}, S_*, S_{3+}, S_{3-}) \leq |\widehat{S}|$, or
- $v \in \widehat{S}_{3-}$, $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-} \cup \{v\}) \in \mathcal{I}_{t'}$, and $\mathbf{r}_{t'}(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-} \cup \{v\}) \leq |\widehat{S}|$

Join. Let $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) \in \mathcal{I}_t$, and let $(\widehat{S}, \widehat{S}_*, \widehat{S}_{3-})$ be a witness. Let t' and t'' be the two children of t. We define $\widehat{S}' = \widehat{S} \cap V(G_{t'})$, $\widehat{S}'' = \widehat{S} \cap V(G_{t''})$, $\widehat{S}'_* = \widehat{S}_* \cap V(G_{t'})$, $\widehat{S}''_* = \widehat{S}_* \cap V(G_{t''})$, $\widehat{S}'_{3-} \subseteq \widehat{S}_{3-} \cap V(G_{t'})$, and $\widehat{S}''_{3-} \subseteq \widehat{S}_{3-} \cap V(G_{t''})$, such that each connected component of $G_t[\widehat{S}'_{3-}]$ (resp. $G_t[\widehat{S}''_{3-}]$) is a C_3 and $G_{t'} \setminus (\widehat{S}' \cup \widehat{S}'_{3-})$ (resp. $G_{t''} \setminus (\widehat{S}'' \cup \widehat{S}''_{3-})$) is a forest). Then we define

- $S' = \widehat{S}' \cap X_t$,

- $S'_{1+} = S_{1+} \cup \{v \in S_{1-} \mid N_{G_t \setminus \widehat{S}}(v) \not\subseteq \widehat{S}'_*\}$,
- $S'_{1-} = \{v \in S_{1-} \mid N_{G_t \setminus \widehat{S}}(v) \subseteq \widehat{S}'_*\}$,
- $S'_* = S_* \cap V(G_{t'})$,
- $S'_{3-} = \widehat{S}'_{3-} \cap X_t$, and
- $S'_{3+} = S_{3+} \cup (S_{3-} \setminus S'_{3-})$.

Note that $(S', S'_{1+}, S'_{1-}, S'_*, S'_{3+}, S'_{3-}) \in \mathcal{I}'_t$. We define $(S'', S''_{1+}, S''_{1-}, S''_*, S''_{3+}, S''_{3-}) \in \mathcal{I}''_t$ similarly. Moreover we can easily check that

- $S = S' = S'', S_* = S'_* = S''_*$,
- $(S'_{1+} \cup S'_{1-}) \cap (S''_{3+} \cup S''_{3-}) = (S''_{1+} \cup S''_{1-}) \cap (S'_{3+} \cup S'_{3-}) = \emptyset$,
- $\forall v \in S'_{1-} \cap S''_{1-}, \exists z \in S_* : N_{G_t[X_t \setminus S]}(v) = \{z\}$,
- $\forall v \in S'_{3-} \cap S''_{3-}, \exists z, z' \in S'_{3-} \cap S''_{3-} : v, z, z'$ induce a C_3 in $G_t[X_t \setminus S]$,
- $(S, S_{1+}, S_{1-}, S_*, S_{3+}, S_{3-}) = (S, S'_{1+} \cap S''_{1+}, S'_{1-} \cup S''_{1-}, S_*, S'_{3+} \cap S''_{3+}, S'_{3-} \cup S''_{3-})$, and
- $\mathbf{r}_{t'}(S', S'_{1+}, S'_{1-}, S'_*, S'_{3+}, S'_{3-}) + \mathbf{r}_{t''}(S'', S''_{1+}, S''_{1-}, S''_*, S''_{3+}, S''_{3-}) - |S| \leq |\widehat{S}|$.

This concludes the proof of correctness of the algorithm. The following theorem summarizes the above discussion.

Theorem 5. *If a nice tree decomposition of G of width w is given, $\{P_4\}$-*Deletion *can be solved in time $\mathcal{O}(36^w \cdot w \cdot n)$.*

5 Single-exponential algorithms for $\{K_{1,s}\}$-TM-Deletion

Similarly to what we did before, we start with a (trivial) structural characterization of the graphs that exclude $K_{1,s}$, for some fixed integer s, as a topological minor.

Lemma 3. *Let s be a positive integer. A graph G contains $K_{1,s}$ as a topological minor if and only if it contains a vertex of degree at least s.*

Proof. Let s be a fixed integer. If a graph G contains a vertex v of degree at least s, then G contains $K_{1,s}$ as a subgraph and so, as a topological minor. If G contains $K_{1,s}$ as a topological minor, then it implies that there exist in G a vertex v and s paths of size at least two such that the intersection of any two of these paths contains precisely v. Thus v has degree at least s. □

Given a fixed integer $s \geq 1$, the s-Bounded-degree Vertex Deletion problem asks, given a graph G and an integer k, whether one can remove at most k vertices from G such that the remaining graph has maximum degree at most s. Lemma 3 implies that for every positive integer s, $\{K_{1,s}\}$-TM-Deletion is exactly $(s-1)$-Bounded-degree Vertex Deletion. For completeness, we provide a simple single-exponential algorithm parameterized by treewidth that solves s-Bounded-degree Vertex Deletion for any fixed integer $s \geq 1$.

Let $s \geq 1$ be a fixed integer, let G be an instance of s-Bounded-degree Vertex Deletion, and let $((T, \mathcal{X}), r, \mathcal{G})$ be a nice tree decomposition of G. We define, for each $t \in V(T)$, the set $\mathcal{I}_t = \{(S, f) \mid S \subseteq X_t, f : X_t \setminus S \to [0, s-1]\}$ and a function $\mathbf{r}_t : \mathcal{I}_t \to \mathbb{N}$ such that for each $(S, f) \in \mathcal{I}_t$, $\mathbf{r}(S, f)$ is the minimum ℓ such that there exists a set $\widehat{S} \subseteq V(G_t)$, called the *witness* of (S, f), that satisfies:

- $|\widehat{S}| \leq \ell$,
- $\widehat{S} \cap X_t = S$, and
- for each $v \in X_t \setminus S$, $\mathsf{deg}_{G_t \setminus \widehat{S}}(v) = f(v)$.

Note that with this definition, $\mathbf{tm}_{\mathcal{F}}(G) = \mathbf{r}_r(\emptyset, \varnothing)$. For each $t \in V(T)$, we assume that we have already computed $\mathbf{r}_{t'}$ for each children t' of t, and we proceed to the computation of $\mathbf{r}_t$. We distinguish several cases depending on the type of node t.

Leaf. $\mathcal{I}_t = \{(\emptyset, \varnothing)\}$ and $\mathbf{r}_t(\emptyset, \varnothing) = 0$.

Introduce vertex. If v is the insertion vertex of X_t and t' is the child of t, then for each $(S, f) \in \mathcal{I}_t$,

$$\begin{aligned} \mathbf{r}_t(S,f) \quad = \quad \min\big(& \{\mathbf{r}_{t'}(S', f) + 1 \mid (S', f) \in \mathcal{I}_{t'},\ S = S' \cup \{v\}\} \\ & \cup \{\mathbf{r}_{t'}(S, f') \mid (S, f') \in \mathcal{I}_{t'}, f(v) = \mathsf{deg}_{G[X_t \setminus S]}(v), \\ & \qquad \forall v' \in N_{G_t[X_t \setminus S]}(v),\ f(v') = f'(v') + 1, \\ & \qquad \forall v' \in X_{t'} \setminus (S \cup N_{G_t[X_t \setminus S]}(v)),\ f(v') = f'(v')\} \big). \end{aligned}$$

Forget vertex. If v is the forget vertex of X_t and t' is the child of t, then for each $(S, f) \in \mathcal{I}_t$,

$$\mathbf{r}_t(S,f) \quad = \quad \min\{\mathbf{r}_{t'}(S', f') \mid (S', f') \in \mathcal{I}_{t'},\ S = S' \setminus \{v\},\ \forall v' \in X_t \setminus S,\ f(v') = f'(v')\}.$$

Join. If t' and t'' are the children of t, then for each $(S, f) \in \mathcal{I}_t$,

$$\begin{aligned} \mathbf{r}(S,f) \quad = \quad \min\{ & \mathbf{r}(S, f') + \mathbf{r}(S, f'') - |S| \\ & \mid (S, f') \in \mathcal{I}_{t'}, (S, f'') \in \mathcal{I}_{t''}, \\ & \forall v \in X_t \setminus S,\ f(v) = f'(v) + f''(v) - \mathsf{deg}_{G_t[X_t \setminus S]}(v)\}. \end{aligned}$$

One can check that for each $t \in V(T)$, the set $\mathcal{I}_t$ is of size at most $(s+1)^{|X_t|}$: for each vertex in $X_t \setminus S$ there are s possible values for its degree, together with the choice of belonging to S or not for each vertex in X_t. Using the same argumentation as in the previous algorithms, we obtain the following theorem.

Theorem 6. *Let $s \geq 1$ be a fixed integer. If a nice tree decomposition of G of width w is given, $\{K_{1,s}\}$-TM-DELETION can be solved in time $\mathcal{O}((s+1)^{2w} \cdot w \cdot n)$.*

6 A single-exponential algorithm for $\{C_4\}$-TM-DELETION

As discussed before, in this section we use the dynamic programming techniques introduced by Bodlaender et al. [7] to obtain a single-exponential algorithm for $\{C_4\}$-TM-DELETION. It is worth mentioning that the $\{C_i\}$-TM-DELETION problem has been studied in digraphs from a non-parameterized point of view [20]. The algorithm we present solves the decision version of $\{C_4\}$-TM-DELETION: the input is a pair (G,k), where G is a graph and k is an integer, and the output is the boolean value $\mathbf{tm}_{\mathcal{F}}(G) \leq k$.

We give some definitions that will be used for the following algorithm. Given a graph G, we denote by $n(G) = |V(G)|$, $m(G) = |E(G)|$, $\mathsf{c}_3(G)$ the number of C_3's that are subgraphs of G, and $\mathsf{cc}(G)$ the number of connected components of G. We say that G satisfies the C_4-*condition* if the following conditions hold:

- G does not contain the $\mathsf{diamond}$ as a subgraph, and
- $n(G) - m(G) + \mathsf{c}_3(G) = \mathsf{cc}(G)$.

As in the case of P_3 and P_4, we state in Lemma 5 a structural characterization of the graphs that exclude C_4 as a (topological) minor. We first need an auxiliary lemma.

Lemma 4. *Let n_0 be a positive integer. Assume that for each graph G' such that $1 \leq n(G') \leq n_0$, $C_4 \not\preceq_{\mathsf{tm}} G'$ if and only if G' satisfies the C_4-condition. If G is a graph that does not contain a* $\mathsf{diamond}$ *as a subgraph and such that $n(G) = n_0$, then $n(G) - m(G) + \mathsf{c}_3(G) \leq \mathsf{cc}(G)$.*

Proof. Let n_0 be a positive integer, and assume that for each graph G' such that $1 \leq n(G') \leq n_0$, $C_4 \not\preceq_{\mathsf{tm}} G'$ if and only if G satisfies the C_4-condition. Let G be a graph that does not contain a $\mathsf{diamond}$ as a subgraph and such that $n(G) = n_0$. Let $S \subseteq E(G)$ such that $C_4 \not\preceq_{\mathsf{tm}} G \setminus S$ and $\mathsf{cc}(G \setminus S) = \mathsf{cc}(G)$ (note that any minimal feedback edge set satisfies these conditions). We have, by hypothesis, that $G \setminus S$ satisfies the C_4-condition, so $n(G \setminus S) - m(G \setminus S) + \mathsf{c}_3(G \setminus S) = \mathsf{cc}(G \setminus S)$. Moreover, as G does not contain a $\mathsf{diamond}$ as a subgraph, each edge of G participates in at most one C_3, and thus

$\mathsf{c}_3(G) - \mathsf{c}_3(G \setminus S) \leq |S|$. As by definition $n(G) = n(G \setminus S)$ and $m(G) - m(G \setminus S) = |S|$, we obtain that $n(G) - m(G) + \mathsf{c}_3(G) \leq \mathsf{cc}(G \setminus S) = \mathsf{cc}(G)$. □

Lemma 5. *Let G be a non-empty graph. $C_4 \not\preceq_{\mathsf{tm}} G$ if and only if G satisfies the C_4-condition.*

Proof. Let G be a non-empty graph, and assume first that $C_4 \not\preceq_{\mathsf{tm}} G$. This directly implies that G does not contain the $\mathsf{diamond}$ as a subgraph. In particular, any two cycles of G, which are necessarily C_3's, cannot share an edge. Let S be a set containing an arbitrary edge of each C_3 in G. By construction, $G \setminus S$ is a forest. As in a forest F, we have $n(F) - m(F) = \mathsf{cc}(F)$, and S is defined such that $|S| = \mathsf{c}_3(G)$ because each edge of G participates in at most one C_3, we obtain that $n(G) - m(G) + \mathsf{c}_3(G) = \mathsf{cc}(G)$. Thus, G satisfies the C_4-condition.

Conversely, assume now that G satisfies the C_4-condition. We prove that $C_4 \not\preceq_{\mathsf{tm}} G$ by induction on $n(G)$. If $n(G) \leq 3$, then $n(G) < n(C_4)$ and so $C_4 \not\preceq_{\mathsf{tm}} G$. Assume now that $n(G) \geq 4$, and that for each graph G' such that $1 \leq n(G') < n(G)$, if G' satisfies the C_4-condition, then $C_4 \not\preceq_{\mathsf{tm}} G'$. We prove that this last implication is also true for G. Note that, as two C_3 cannot share an edge in G, we have that $\mathsf{c}_3(G) \leq \frac{m(G)}{3}$. This implies that the minimum degree of G is at most 2. Indeed, if each vertex of G had degree at least 3, then $m(G) \geq \frac{3}{2}n(G)$, which together with the relations $\mathsf{c}_3(G) \leq \frac{m(G)}{3}$ and $n(G) - m(G) + \mathsf{c}_3(G) = \mathsf{cc}(G)$ would imply that $\mathsf{cc}(G) \leq 0$, a contradiction. Let $v \in V(G)$ be a vertex with minimum degree. We distinguish two cases according to the degree of v.

If v has degree 0 or 1, then the graph $G \setminus \{v\}$ satisfies the C_4-condition as well, implying that $C_4 \not\preceq_{\mathsf{tm}} G \setminus \{v\}$. As v has degree at most one, it cannot be inside a cycle, hence $C_4 \not\preceq_{\mathsf{tm}} G$.

Assume that v has degree two and participates in a C_3. As G does not contain a $\mathsf{diamond}$ as a subgraph, $C_4 \preceq_{\mathsf{tm}} G$ if and only if $C_4 \preceq_{\mathsf{tm}} G \setminus \{v\}$. Moreover $n(G \setminus \{v\}) = n(G) - 1$, $m(G \setminus \{v\}) = m(G) - 2$, $\mathsf{c}_3(G \setminus \{v\}) = \mathsf{c}_3(G) - 1$, and $\mathsf{cc}(G \setminus \{v\}) = \mathsf{cc}(G)$. This implies that $G \setminus \{v\}$ satisfies the C_4-condition, hence $C_4 \not\preceq_{\mathsf{tm}} G \setminus \{v\}$, and therefore $C_4 \not\preceq_{\mathsf{tm}} G$.

Finally, assume that v has degree two and does not belong to any C_3. Using the induction hypothesis and Lemma 4, we have that $n(G \setminus \{v\}) - m(G \setminus \{v\}) + \mathsf{c}_3(G \setminus \{v\}) \leq \mathsf{cc}(G \setminus \{v\})$. As $n(G \setminus \{v\}) = n(G) - 1$, $m(G \setminus \{v\}) = m(G) - 2$, $\mathsf{c}_3(G \setminus \{v\}) = \mathsf{c}_3(G)$, v has degree two in G, and G satisfies the C_4-condition, we obtain that $\mathsf{cc}(G \setminus \{v\}) = \mathsf{cc}(G) - 1$. This implies that $G \setminus \{v\}$ satisfies the C_4-condition, and thus $C_4 \not\preceq_{\mathsf{tm}} G \setminus \{v\}$. Since v disconnects one of the connected components of G it cannot participate in a cycle of G, hence $C_4 \not\preceq_{\mathsf{tm}} G$. □

Lemma 6. *If G is a non-empty graph such that $C_4 \not\preceq_{\mathsf{tm}} G$, then $m(G) \leq \frac{3}{2}(n(G)-1)$.*

Proof. As $C_4 \not\preceq_{\mathsf{tm}} G$, by Lemma 5 G satisfies the C_4-condition. It follows that $\mathsf{c}_3(G) \leq \frac{1}{3}m(G)$. Moreover, as G is non-empty, we have that $1 \leq \mathsf{cc}(G)$. The lemma follows by using these inequalities in the equality $n(G) - m(G) + \mathsf{c}_3(G) = \mathsf{cc}(G)$. □

We now have all the tools needed to describe our algorithm. Recall that the basic ingredients of the rank-based approach of Bodlaender et al. [7] were given in Section 2. Let G be a graph and k be an integer. The algorithm we describe solves the decision version of $\{C_4\}$-TM-Deletion. This algorithm is based on the one given in [7, Section 3.5] for Feedback Vertex Set.

We define a new graph $G_0 = (V(G) \cup \{v_0\}, E(G) \cup E_0)$, where v_0 is a new vertex and $E_0 = \{\{v_0, v\} \mid v \in V(G)\}$. The role of v_0 is to artificially guarantee the connectivity of the solution graph, so that the machinery of Bodlaender et al. [7] can be applied. In the following, for each subgraph H of G_0, for each $Z \subseteq V(H)$, and for each $Z_0 \subseteq E_0 \cap E(H[Z])$, we denote by $H\langle Z, Z_0\rangle$ the graph $\big(Z, Z_0 \cup E\big(H[Z \setminus \{v_0\}]\big)\big)$.

Given a nice tree decomposition of G of width w, we define a nice tree decomposition $((T,\mathcal{X}), r, \mathcal{G})$ of G_0 of width $w+1$ such that the only empty bags are the root and the leaves and for each $t \in T$, if $X_t \neq \emptyset$ then $v_0 \in X_t$. Note that this can be done in linear time. For each bag t, each integers i, j, and ℓ, each function $\mathbf{s} : X_t \to \{0,1\}$, each function $\mathbf{s}_0 : \{v_0\} \times \mathbf{s}^{-1}(1) \to \{0,1\}$, each function $\mathbf{r} : E(G_t\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\rangle) \to \{0,1\}$, and for each partition $p \in \Pi(\mathbf{s}^{-1}(1))$, if $C_4 \not\preceq_{\mathsf{tm}} G_t\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\rangle$, we define:

$$
\begin{aligned}
\mathcal{E}_t(p,\mathbf{s},\mathbf{s}_0,\mathbf{r},i,j,\ell) \quad = \quad & \{(Z,Z_0) \mid (Z,Z_0) \in 2^{V_t} \times 2^{E_0 \cap E(G_t)} \\
& |Z| = i,\ |E(G_t\langle Z,Z_0\rangle)| = j,\ \mathsf{c}_3(G_t\langle Z,Z_0\rangle) = \ell, \\
& G_t\langle Z,Z_0\rangle \text{ does not contain the } \textsf{diamond} \text{ as a subgraph}, \\
& Z \cap X_t = \mathbf{s}^{-1}(1),\ Z_0 \cap (X_t \times X_t) = \mathbf{s}_0^{-1}(1), \\
& \qquad\qquad\qquad\qquad v_0 \in X_t \Rightarrow \mathbf{s}(v_0) = 1, \\
& \forall u \in Z \setminus X_t : \text{ either } t \text{ is the root or} \\
& \qquad \exists u' \in \mathbf{s}^{-1}(1) : u \text{ and } u' \text{ are connected in } G_t\langle Z,Z_0\rangle, \\
& \forall v_1,v_2 \in \mathbf{s}^{-1}(1) : p \sqsubseteq V_t[\{v_1,v_2\}] \Leftrightarrow v_1 \text{ and } v_2 \text{ are con-} \\
& \qquad\qquad\qquad\qquad \text{nected in } G_t\langle Z,Z_0\rangle, \\
& \forall e \in E(G_t\langle Z,Z_0\rangle) \cap \binom{\mathbf{s}^{-1}(1)}{2} : \mathbf{r}(e) = 1 \Leftrightarrow \ e \text{ is an edge} \\
& \qquad\qquad\qquad\qquad \text{of a } C_3 \text{ in } G_t\langle Z,Z_0\rangle\}
\end{aligned}
$$

$$
\mathcal{A}_t(\mathbf{s},\mathbf{s}_0,\mathbf{r},i,j,\ell) \quad = \quad \{p \mid p \in \Pi(\mathbf{s}^{-1}(1)),\ \mathcal{E}_t(p,\mathbf{s},\mathbf{s}_0,\mathbf{r},i,j,\ell) \neq \emptyset\}.
$$

Otherwise, i.e., if $C_4 \preceq_{\mathsf{tm}} G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle$, we define

$$
\mathcal{A}_t(\mathbf{s},\mathbf{s}_0,\mathbf{r},i,j,\ell) \quad = \quad \emptyset.
$$

Note that we do not need to keep track of partial solutions if $C_4 \preceq_{\mathsf{tm}} G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle$, as we already know they will not lead to a global solution. Moreover, if $C_4 \not\preceq_{\mathsf{tm}} G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle$, then by Lemma 6, $m(G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle) \leq \frac{3}{2}(n(G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle)-1)$.

Using the definition of $\mathcal{A}_r$, Lemma 5, and Lemma 6, we have that $\mathbf{tm}_{\{C_4\}}(G) \leq k$ if and only if for some $i \geq |V(G) \cup \{v_0\}| - k$ and some $j \leq \frac{2}{3}(i-1)$, we have $\mathcal{A}_r(\varnothing,\varnothing,\varnothing,i,j,1+j-i) \neq \emptyset$. For each $t \in V(T)$, we assume that we have already computed $\mathcal{A}_{t'}$ for each children t' of t, and we proceed to the computation of $\mathcal{A}_t$. As usual, we distinguish several cases depending on the type of node t.

Leaf. By definition of $\mathcal{A}_t$ we have $\mathcal{A}_t(\varnothing,\varnothing,\varnothing,0,0,0) = \{\emptyset\}$.

Introduce vertex. Let v be the insertion vertex of X_t, let t' be the child of t, let $\mathbf{s}$, $\mathbf{s}_0$, and $\mathbf{r}$ the functions defined as before, let $H = G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle$, and let d_3 be the number of C_3's of H that contain the vertex v.

- If $C_4 \preceq_{\mathsf{tm}} H$ or if $v = v_0$ and $\mathbf{s}(v_0) = 0$, then by definition of $\mathcal{A}_t$ we have that $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) = \emptyset$.
- Otherwise, if $\mathbf{s}(v) = 0$, then, by definition of $\mathcal{A}_t$, it holds that $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) = \mathcal{A}_{t'}(\mathbf{s}|_{X_{t'}}, \mathbf{s}_0|_{E_{t'}}, \mathbf{r}|_{E_{t'}}, i, j, \ell)$.
- Otherwise, if $v = v_0$, then by construction of the nice tree decomposition, we know that t' is a leaf of T and so $\mathbf{s} = \{(v_0, 1)\}$, $\mathbf{s}_0 = \mathbf{r} = \emptyset$, $j = \ell = i - 1 = 0$ and $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) = \mathsf{ins}(\{v_0\}, \mathcal{A}_{t'}(\varnothing, \varnothing, \varnothing, 0, 0, 0))$.
- Otherwise, we know that $v \neq v_0$, $\mathbf{s}(v) = 1$, $v_0 \in N_{G[\mathbf{s}^{-1}(1)]}(v)$, and $C_4 \not\preceq_{\mathsf{tm}} H$. As $\mathbf{s}(v) = 1$, we have to insert v and we have to make sure that all vertices of $N_H[v] \setminus \{v_0\}$ are in the same connected component of H. The only remaining choice is either we insert the edge $\{v, v_0\}$ or not. This is handle by the value of $\mathbf{s}_0(\{v_0, v\})$. So, by adding v, we add one vertex, $|N_H(v)|$ edges, and d_3 C_3's. We also have to take care not to introduce a $\mathsf{diamond}$. For this, the function $\mathbf{r}$ should be such that, for every edge e contained in a C_3's of H that contains the vertex v, $\mathbf{r}(e) = 1$. We define $\mathbf{r}' : E(H[X_{t'}]) \to \{0, 1\}$ such that for every edge $e \in E(H[X_{t'}])$ contained in a C_3's of H that contains the vertex v, $\mathbf{r}'(e) = 0$, and for each other edge e of $H[X_{t'}]$, $\mathbf{r}'(e) = \mathbf{r}(e)$. Therefore, we have that

$$\begin{aligned}\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) &= \\ &\mathsf{glue}(N_H[v], \mathsf{ins}(\{v\}, \mathcal{A}_{t'}(\mathbf{s}|_{X_{t'}}, \mathbf{s}_0|_{E_{t'}}, \mathbf{r}', i - 1, j - |N_H(v)|, \ell - d_3))).\end{aligned}$$

Forget vertex. Let v be the forget vertex of X_t, let t' be the child of t, and let $\mathbf{s}$, $\mathbf{s}_0$, and $\mathbf{r}$ the functions defined as before. For each function, we have a choice on how it can be extended in t', and we potentially need to consider every possible such extension. Note the number of vertices, edges, or C_3's is not affected. We obtain that

$$\begin{aligned}\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) \;=\; & A_{t'}(\mathbf{s} \cup \{(v, 0)\}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) \\ & \biguplus_{\substack{\mathbf{s}' : X_{t'} \to \{0,1\},\ \mathbf{s}'|_{X_t} = \mathbf{s},\ \mathbf{s}'(v) = 1 \\ \mathbf{s}'_0 : \{v_0\} \times \mathbf{s}'^{-1}(1) \to \{0,1\},\ \mathbf{s}'_0|_{X_t} = \mathbf{s}_0 \\ \mathbf{r}' : E(G_t\langle \mathbf{s}'^{-1}(1), \mathbf{s}_0'^{-1}(1) \rangle) \to \{0,1\},\ \mathbf{r}'|_{X_t} = \mathbf{r}}} \mathsf{proj}(\{v\}, A_{t'}(\mathbf{s}', \mathbf{s}'_0, \mathbf{r}', i, j, \ell)).\end{aligned}$$

Join. Let t' and t'' be the two children of t, let $\mathbf{s}$, $\mathbf{s}_0$, and $\mathbf{r}$ be the functions defined as before, let $H = G_t\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1) \rangle$, and let $S \subseteq E(H)$ be the set of edges that participate in a C_3 of H.

We join every compatible entries $A_{t'}(\mathbf{s}', \mathbf{s}_0', \mathbf{r}', i', j', \ell')$ and $A_{t''}(\mathbf{s}'', \mathbf{s}_0'', \mathbf{r}'', i'', j'', \ell'')$. For two such entries being compatible, we need $\mathbf{s}' = \mathbf{s}'' = \mathbf{s}$ and $\mathbf{s}_0' = \mathbf{s}_0'' = \mathbf{s}_0$. Moreover, we do not want the solution graph to contain a $\mathsf{diamond}$ as a subgraph, and for this we need $\mathbf{r}'^{-1}(1) \cap \mathbf{r}''^{-1}(1) = S$. Indeed, either H contains the $\mathsf{diamond}$ as a subgraph, and then $A_{t'}(\mathbf{s}', \mathbf{s}_0', \mathbf{r}', i', j', \ell') = A_{t''}(\mathbf{s}'', \mathbf{s}_0'', \mathbf{r}'', i'', j'', \ell'') = \{\emptyset\}$, or the $\mathsf{diamond}$ is created by joining two C_3's, one from t' and the other one from t'', sharing a common edge. This is possible only if $(\mathbf{r}'^{-1}(1) \cap \mathbf{r}''^{-1}(1)) \setminus S \neq \emptyset$. For the counters, we have to be careful in order not to count some element twice. We obtain that

$$\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, \mathbf{r}, i, j, \ell) = \biguplus_{\substack{\mathbf{r}', \mathbf{r}'' : E(H) \to \{0,1\}, \\ \mathbf{r}'^{-1}(1) \cap \mathbf{r}''^{-1}(1) = S \\ i' + i'' = i + |V(H)| \\ j' + j'' = j + |E(H)| \\ \ell' + \ell'' = \ell + \mathsf{c}_3(H)}} \mathsf{join}(A_{t'}(\mathbf{s}, \mathbf{s}_0, \mathbf{r}', i', j', \ell'), A_{t''}(\mathbf{s}, \mathbf{s}_0, \mathbf{r}'', i'', j'', \ell'')).$$

Theorem 7. $\{C_4\}$-TM-DELETION *can be solved in time* $2^{\mathcal{O}(\mathsf{tw})} \cdot n^7$.

Proof. The algorithm works in the following way. For each node $t \in V(T)$ and for each entry M of its table, instead of storing $\mathcal{A}_t(M)$, we store $\mathcal{A}'_t(M) = \mathsf{reduce}(\mathcal{A}_t(M))$ by using Theorem 3. As each of the operation we use preserves representation by Proposition 2, we obtain that for each node $t \in V(T)$ and for each possible entry M, $\mathcal{A}'_t(M)$ represents $\mathcal{A}_t(M)$. In particular, we have that $\mathcal{A}'_r(M) = \mathsf{reduce}(\mathcal{A}_r(M))$ for each possible entry M. Using the definition of $\mathcal{A}_r$, Lemma 5, and Lemma 6, we have that $\mathbf{tm}_{\{C_4\}}(G) \leq k$ if and only if for some $i \geq |V(G) \cup \{v_0\}| - k$ and some $j \leq \frac{2}{3}(i-1)$, we have $\mathcal{A}'_r(\varnothing, \varnothing, \varnothing, i, j, 1 + j - i) \neq \emptyset$.

We now focus on the running time of the algorithm. The size of the intermediate sets of weighted partitions, for a leaf node and for an introduce vertex node are upper-bounded by $2^{|\mathbf{s}^{-1}(1)|}$. For a forget vertex node, as in the big union operation we take into consideration a unique extension of $\mathbf{s}$, at most two possible extensions of $\mathbf{s}_0$, and at most $2^{|\mathbf{s}^{-1}(1)|}$ possible extensions for $\mathbf{r}$, we obtain that the intermediate sets of weighted partitions have size at most $2^{|\mathbf{s}^{-1}(1)|} + 2 \cdot 2^{|\mathbf{s}^{-1}(1)|} \cdot 2^{|\mathbf{s}^{-1}(1)|} \leq 2^{2|\mathbf{s}^{-1}(1)|+2}$. For a join node, as in the big union operation we take into consideration at most $2^{|E(H)|}$ possible functions $\mathbf{r}'$ and as many functions $\mathbf{r}''$, at most $n + |\mathbf{s}^{-1}(1)|$ choices for i' and i'', at most $\frac{3}{2}(n-1) + |E(H)|$ choices for j' and j'', and at most $\frac{1}{2}(n-1) + \frac{1}{3}|E(H)|$ choices for ℓ' and ℓ'', we obtain that the intermediate sets of weighted partitions have size at most $2^{|E(H)|} \cdot 2^{|E(H)|} \cdot (n + |\mathbf{s}^{-1}(1)|) \cdot (\frac{3}{2}(n-1) + |E(H)|) \cdot (\frac{1}{2}(n-1) + \frac{1}{3}|E(H)|) \cdot 4^{|\mathbf{s}^{-1}(1)|}$. As each time we can check the condition $C_4 \not\preceq_{\mathsf{tm}} H$, by Lemma 6 $m(H) \leq \frac{3}{2}(n(H) - 1)$, so we obtain that the intermediate sets of weighted partitions have size at most $6 \cdot n^3 \cdot 2^{5|\mathbf{s}^{-1}(1)|}$.

Moreover, for each node $t \in V(T)$, the function reduce will be called as many times as the number of possible entries, i.e., at most $2^{\mathcal{O}(w)} \cdot n^3$ times. Thus, using Theorem 3, $\mathcal{A}'_t$ can be computed in time $2^{\mathcal{O}(w)} \cdot n^6$. The theorem follows by taking into account the linear number of nodes in a nice tree decomposition. $\square$

7 A single-exponential algorithm for {paw}-TM-Deletion

Again, we start with a simple structural characterization of the simple graphs that exclude the paw as a topological minor; recall the paw graph in Figure 2.

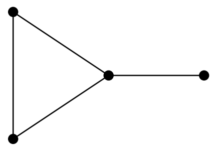


Figure 2: The paw graph.

Lemma 7. *Let G be a simple graph.* paw $\not\preceq_{\mathsf{tm}} G$ *if and only if each connected component of G is either a cycle or a tree.*

Proof. It is easy to see that neither a cycle nor a tree contain the paw as a topological minor. Let G be a graph such that paw $\not\preceq_{\mathsf{tm}} G$. Let us assume w.l.o.g. that G is connected. If G does not contains a cycle, then it is a tree. Otherwise, let C be a chordless cycle in G. If G contains a vertex v that is not in C then, as G is connected, there exists a path from v to C containing at least two vertices. This is not possible, as it would imply that G contains the paw as a topological minor. As C is chordless and G is simple, we obtain that G is exactly the cycle C, and the lemma follows. $\square$

We present an algorithm that solves the decision version of {paw}-TM-Deletion. As the algorithm that we presented for $\{C_4\}$-TM-Deletion in Section 6, this algorithm is based on the one given in [7, Section 3.5] for Feedback Vertex Set. Let G be a graph and k be an integer. The idea of the following algorithm is to partition $V(G)$ into three sets. The first one will be the solution set S, the second one will be a set F of vertices that induces a forest, and the third one will be a set C of vertices that induces a collection of cycles. If we can partition our graph into three such sets (S, F, C) such that there is no edge between a vertex of F and a vertex of C and such that $|S| \leq k$, then, using Lemma 7, we know that $\mathbf{tm}_{\{\mathsf{paw}\}}(G) \leq k$. On the other hand, if such a partition does not exist, we know that $\mathbf{tm}_{\{\mathsf{paw}\}}(G) > k$. The main idea of this algorithm is to combine classical dynamic programming techniques in order to verify that C induces a collection of cycles, and the rank-based approach in order to verify that F induces a forest.

As for $\{C_4\}$-TM-Deletion, we define a new graph $G_0 = (V(G) \cup \{v_0\}, E(G) \cup E_0)$, where v_0 is a new vertex and $E_0 = \{\{v_0, v\} \mid v \in V(G)\}$. We recall that for each subgraph H of G_0, for each $Z_1 \subseteq V(H)$, and for each $Y \subseteq E_0 \cap E(H[Z_1])$, we denote by $H\langle Z_1, Y\rangle$ the graph $\big(Z_1, Y \cup E\big(H[Z_1 \setminus \{v_0\}]\big)\big)$.

Given a nice tree decomposition of G of width w, we define a nice tree decomposition $((T, \mathcal{X}), r, \mathcal{G})$ of G_0 of width $w + 1$ such that the only empty bags are the root and the leaves and for each $t \in T$, if $X_t \neq \emptyset$ then $v_0 \in X_t$. Note that this can be done in linear time. For each bag t, each integers i, j, and ℓ, each function $\mathbf{s} : X_t \to \{0, 1, 2_0, 2_1, 2_2\}$, each function $\mathbf{s}_0 : \{v_0\} \times \mathbf{s}^{-1}(1) \to \{0, 1\}$, and each partition $p \in \Pi(\mathbf{s}^{-1}(1))$, we define:

$$\begin{aligned}
\mathcal{E}_t(p, \mathbf{s}, \mathbf{s}_0, i, j, \ell) \quad = \quad & \{(Z_1, Z_2, Y) \mid (Z_1, Z_2, Y) \in 2^{V_t} \times 2^{V_t} \times 2^{E_0 \cap E(G_t)},\ Z_1 \cap Z_2 = \emptyset, \\
& |Z_1| = i,\ |Z_2| = \ell, |E(G_t[Z_1 \setminus \{v_0\}]) \cup Y| = j, \\
& \forall e \in E_0 \cap E_t,\ \mathbf{s}_0(e) = 1 \Leftrightarrow e \in Y, \\
& \forall v \in Z_2 \cap X_t,\ \mathbf{s}(v) = 2_z \text{ with } z = \mathsf{deg}_{G_t[Z_2]}(v), \\
& \forall v \in Z_2 \setminus X_t,\ \mathsf{deg}_{G_t[Z_2]}(v) = 2, \\
& Z_1 \cap X_t = \mathbf{s}^{-1}(1),\ v_0 \in X_t \Rightarrow \mathbf{s}(v_0) = 1, \\
& G_t\big\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\big\rangle \text{ is a forest}, \\
& \forall u \in Z_1 \setminus X_t : \text{ either } t \text{ is the root or} \\
& \qquad \exists u' \in \mathbf{s}^{-1}(1) : u \text{ and } u' \text{ are connected in } G_t\langle Z_1, Y\rangle, \\
& \forall v_1, v_2 \in \mathbf{s}^{-1}(1) : p \sqsubseteq V_t[\{v_1, v_2\}] \Leftrightarrow v_1 \text{ and } v_2 \text{ are con-} \\
& \qquad\qquad \text{nected in } G_t\langle Z_1, Y\rangle, \\
& \forall (u, v) \in (Z_1 \setminus \{v_0\}) \times Z_2,\ \{u, v\} \notin E(G_t)\}
\end{aligned}$$

$$\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) \quad = \quad \{p \mid p \in \Pi(\mathbf{s}^{-1}(1)),\ \mathcal{E}_t(p, \mathbf{s}, \mathbf{s}_0, i, j, \ell) \neq \emptyset\}.$$

In the definition of $\mathcal{E}_t$, the sets Z_1 (resp. Z_2) correspond to the set F (resp. C) restricted to G_t. The vertex v_0 and the set Y exist to ensure that F will be connected.

By Lemma 7, we have that the given instance of $\{\mathsf{paw}\}$-TM-Deletion is a Yes-instance if and only if for some i and ℓ, $i+\ell \geq |V(G) \cup \{v_0\}| - k$ and $\mathcal{A}_r(\varnothing, \varnothing, i, i-1, \ell) \neq \emptyset$. For each $t \in V(T)$, we assume that we have already computed $\mathcal{A}_{t'}$ for every children t' of t, and we proceed to the computation of $\mathcal{A}_t$. As usual, we distinguish several cases depending on the type of node t.

Leaf. By definition of $\mathcal{A}_t$, we have $\mathcal{A}_t(\varnothing, \varnothing, 0, 0, 0) = \{\emptyset\}$.

Introduce vertex. Let v be the insertion vertex of X_t, let t' be the child of t, let $\mathbf{s} : X_t \to \{0, 1, 2_0, 2_1, 2_2\}$, $\mathbf{s}_0 : \{v_0\} \times \mathbf{s}^{-1}(1) \to \{0, 1\}$, and let $H = G_t\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\rangle$.

- If $v = v_0$ and $\mathbf{s}(v_0) \in \{0, 2_0, 2_1, 2_2\}$ or if H contains a cycle, then by definition of $\mathcal{A}_t$ we have that $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \emptyset$.
- Otherwise, if $v = v_0$, then by construction of the nice tree decomposition, we know that t' is a leaf of T and so $\mathbf{s} = \{(v_0, 1)\}$, $j = \ell = i - 1 = 0$ and $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \mathsf{ins}(\{v_0\}, \mathcal{A}_{t'}(\varnothing, \varnothing, 0, 0, 0))$.
- Otherwise, if $\mathbf{s}(v) = 0$, then, by definition of $\mathcal{A}_t$, it holds that $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \mathcal{A}_{t'}(\mathbf{s}|_{X_{t'}}, \mathbf{s}|_{E_{t'}}, i, j, \ell)$.
- Otherwise, if $\mathbf{s}(v) = 2_z$, $z \in \{0, 1, 2\}$, then let $Z_2' = N_{G_t[X_t]}(v) \setminus \mathbf{s}^{-1}(0)$. If $Z_2' \not\subseteq \mathbf{s}^{-1}(\{2_1, 2_2\})$ or $|Z_2'| \neq z$ then $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \emptyset$. Otherwise $Z_2' \subseteq \mathbf{s}^{-1}(\{2_1, 2_2\})$ and $|Z_2'| = z$, and with $\mathbf{s}' : X_{t'} \to \{0, 1, 2_0, 2_1, 2_2\}$ defined such that $\forall v' \in X_{t'} \setminus Z_2'$, $\mathbf{s}'(v') = \mathbf{s}(v')$ and for each $v' \in Z_2'$ such that $\mathbf{s}(v') = 2_{z'}$, $z' \in \{1, 2\}$, $\mathbf{s}'(v') = 2_{z'-1}$. It holds that $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \mathcal{A}_{t'}(\mathbf{s}', \mathbf{s}_0, i, j, \ell - 1)$.
- Otherwise, we know that $v \neq v_0$, $\mathbf{s}(v) = 1$, and $v_0 \in N_{G[\mathbf{s}^{-1}(1)]}(v)$. First, if $N_{G_t[X_t]}(v) \setminus \mathbf{s}^{-1}(0) \not\subseteq \mathbf{s}^{-1}(1)$, then $\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \emptyset$. Indeed, this implies that the cycle part and the forest part are connected. As $\mathbf{s}(v) = 1$, we have to insert v in the forest part and we have to make sure that all vertices of $N_H[v]$ are in the same connected component of H. The only remaining choice is to insert the edge $\{v, v_0\}$ or not. Again, this is handled by the function $\mathbf{s}_0$. By adding v, we add one vertex and $|N_H(v)|$ edges in the forest part. Therefore, we have that

$$\begin{aligned} \mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) &= \\ \mathsf{glue}(N_H[v], \mathsf{ins}(\{v\}, &\mathcal{A}_{t'}(\mathbf{s}|_{X_{t'}}, \mathbf{s}_0|_{E_{t'}}, i - 1, j - |N_H(v)|, \ell))). \end{aligned}$$

Forget vertex. Let v be the forget vertex of X_t, let t' be the child of t, and let $\mathbf{s} : X_t \to \{0, 1, 2_0, 2_1, 2_2\}$. As a vertex from the collection of cycles can be removed only if it has exactly two neighbors, we obtain that

$$\begin{aligned} \mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) &= A_{t'}(\mathbf{s} \cup \{(v, 0)\}, \mathbf{s}_0, i, j, \ell) \\ &\uplus \mathsf{proj}(\{v\}, A_{t'}(\mathbf{s} \cup \{(v, 1)\}, \mathbf{s}_0 \cup \{(\{v_0, v\}, 0)\}, i, j, \ell)) \\ &\uplus \mathsf{proj}(\{v\}, A_{t'}(\mathbf{s} \cup \{(v, 1)\}, \mathbf{s}_0 \cup \{(\{v_0, v\}, 1)\}, i, j, \ell)) \\ &\uplus A_{t'}(\mathbf{s} \cup \{(v, 2_2)\}, \mathbf{s}_0, i, j, \ell). \end{aligned}$$

Join. Let t' and t'' be the two children of t, let $\mathbf{s} : X_t \to \{0,1,2_0,2_1,2_2\}$, $\mathbf{s}_0 : \{v_0\} \times \mathbf{s}^{-1}(1) \to \{0,1\}$, and let $H = G_t\langle \mathbf{s}^{-1}(1), \mathbf{s}_0^{-1}(1)\rangle$. Given three functions $\mathbf{s}^*, \mathbf{s}', \mathbf{s}'' : X_t \to \{0,1,2_0,2_1,2_2\}$, we say that $\mathbf{s}^* = \mathbf{s}' \oplus \mathbf{s}''$ if for each $v \in \mathbf{s}^{-1}(\{0,1\})$, $\mathbf{s}^*(v) = \mathbf{s}'(v) = \mathbf{s}''(v)$, and for each $v \in X_t$ such that $\mathbf{s}^*(v) = 2_z$, $z \in \{0,1,2\}$, there exist $z', z'' \in \{0,1,2\}$ such that $\mathbf{s}'(v) = 2_{z'}$, $\mathbf{s}''(v) = 2_{z''}$, and $z = z' + z'' - \mathsf{deg}_{G_t[X_t \setminus \mathbf{s}^{-1}(0)]}(v)$.

We join every compatible entries $A_{t'}(\mathbf{s}', \mathbf{s}'_0, i', j', \ell')$ and $A_{t''}(\mathbf{s}'', \mathbf{s}''_0, i'', j'', \ell'')$. For two such entries being compatible, we need $\mathbf{s}' \oplus \mathbf{s}''$ to be defined and $\mathbf{s}'_0 = \mathbf{s}''_0$. We obtain that

$$\mathcal{A}_t(\mathbf{s}, \mathbf{s}_0, i, j, \ell) = \biguplus_{\substack{\mathbf{s}', \mathbf{s}'' : X_t \to \{0,1,2_0,2_1,2_2\}, \\ \mathbf{s} = \mathbf{s}_1 \oplus \mathbf{s}_2 \\ i' + i'' = i + |V(H)| \\ j' + j'' = j + |E(H)| \\ \ell' + \ell'' = \ell + |\mathbf{s}^{-1}(\{2_0,2_1,2_2\})|}} \mathsf{join}(A_{t'}(\mathbf{s}', \mathbf{s}_0, i', j', \ell'), A_{t''}(\mathbf{s}'', \mathbf{s}_0, i'', j'', \ell'')).$$

Theorem 8. $\{\mathsf{paw}\}$-TM-Deletion *can be solved in time* $2^{\mathcal{O}(\mathsf{tw})} \cdot n^7$.

Proof. The algorithm works in the following way. For each node $t \in V(T)$ and for each entry M of its table, instead of storing $\mathcal{A}_t(M)$, we store $\mathcal{A}'_t(M) = \mathsf{reduce}(\mathcal{A}_t(M))$ by using Theorem 3. As each of the operations we use preserves representation by Proposition 2, we obtain that for each node $t \in V(T)$ and for each possible entry M, $\mathcal{A}'_t(M)$ represents $\mathcal{A}_t(M)$. In particular, we have that $\mathcal{A}'_r(M) = \mathsf{reduce}(\mathcal{A}_r(M))$ for each possible entry M. Using the definition of $\mathcal{A}_r$ and Lemma 7, we have that $\mathbf{tm}_{\{\mathsf{paw}\}}(G) \leq k$ if and only if for some i and ℓ, $i + \ell \geq |V(G) \cup \{v_0\}| - k$ and $\mathcal{A}'_r(\varnothing, \varnothing, i, i-1, \ell) \neq \emptyset$.

We now focus on the running time of the algorithm. The size of the intermediate sets of weighted partitions for a leaf node and for an introduce vertex node, are upper-bounded by $2^{|\mathbf{s}^{-1}(1)|}$. For a forget vertex node, we take the union of four sets of size $2^{|\mathbf{s}^{-1}(1)|}$, so the intermediate sets of weighted partitions have size at most $4 \cdot 2^{|\mathbf{s}^{-1}(1)|}$. For a join node, as in the big union operation we take into consideration at most $5^{|X_t|}$ possible functions $\mathbf{s}'$, as many functions $\mathbf{s}''$, at most $n + |\mathbf{s}^{-1}(1)|$ choices for i' and i'', at most $n + |\mathbf{s}^{-1}(1)|$ choices for j' and j'' (as H is always a forest), and at most $n + |\mathbf{s}^{-1}(\{2_0,2_1,2_2\}|$ choices for ℓ' and ℓ'', we obtain that the intermediate sets of weighted partitions have size at most $25^{|X_t|} \cdot (n + |\mathbf{s}^{-1}(1)|)^2 \cdot (n + |\mathbf{s}^{-1}(\{2_0,2_1,2_2\}|) \cdot 4^{|\mathbf{s}^{-1}(1)|}$. We obtain that the intermediate sets of weighted partitions have size at most $(n + |X_t|)^3 \cdot 100^{|X_t|}$. Moreover, for each node $t \in V(T)$, the function reduce will be called as many times as the number of possible entries, i.e., at most $2^{\mathcal{O}(w)} \cdot n^3$ times. Thus, using Theorem 3, $\mathcal{A}'_t$ can be computed in time $2^{\mathcal{O}(w)} \cdot n^6$. The theorem follows by taking into account the linear number of nodes in a nice tree decomposition. $\square$

8 A single-exponential algorithm for {chair}-TM-Deletion

As in the previous cases that we solved in single-exponential time, we start with a structural characterization of the graphs that exclude the chair as a topological minor (hence, as a minor as well).

Lemma 8. *Let G be a graph.* chair $\not\preceq_{\mathsf{tm}} G$ *if and only if every connected component of G of size at least five is a path, a cycle, or a star.*

Proof. It is straightforward to check that a path, a cycle, a star, or a graph of size at most 4 do not contain the chair as a topological minor. Conversely, let G be a connected graph of size at least five that excludes the chair as a topological minor. Let P be a longest path of G. We denote by p_1 and p_2 the endpoints of this path P. It is easy to see that if $|V(P)| \leq 3$, then G has to be a star. Assume now that $|V(P)| \geq 4$. Then we have that $V(P) = V(G)$. Indeed, assume that $V(P) \neq V(G)$ and let $v \in V(G) \setminus V(P)$ be a vertex adjacent with a vertex of $V(P)$ in G. This vertex should exist by the connectivity of G. By maximality of P, v cannot be adjacent to p_1 or p_2 and as chair $\not\preceq_{\mathsf{tm}} G$, v cannot be adjacent to an internal vertex of the path P. Thus, $|V(P)| = |V(G)| \geq 5$. Moreover, as chair $\not\preceq_{\mathsf{tm}} G$, neither p_1 nor p_2 can be adjacent to an internal vertex of the path P, and so, $E(P) \subseteq E(G) \subseteq E(P) \cup \{p_1, p_2\}$. Thus, G is either a path or a cycle. □

With Lemma 8 at hand, an algorithm for {chair}-TM-Deletion running in time $2^{\mathcal{O}(\mathsf{tw})} \cdot n^{\mathcal{O}(1)}$ can be obtained using standard dynamic programming techniques. For this, as we did for $\{P_3\}$-TM-Deletion, $\{P_4\}$-TM-Deletion, and $\{K_{1,s}\}$-TM-Deletion, $s \in \mathbb{N}$, we label the vertices on each bag. Each label carries two types of information. On the one hand, it indicates whether a vertex is in a collection of paths or cycles, a collection of stars, a clique of size 4, a paw, or a diamond; note that these are all the possible graphs on at most 4 vertices (see Figure 1). On the other hand, it also indicates in which "state" a vertex is with regard to the already computed graph, which will become clear below.

As the number of distinct labels needed for the algorithm for {chair}-TM-Deletion is quite large and the algorithm itself is not complicated, we will avoid the formal description of it. Namely, as we will discuss, we need 17 distinct labels on the vertices and three distinct labels on the edges. We only describe how to label the vertices for each type of component. Note that, for instance, a P_3 is both a path and a star. We do not consider this as an issue, and we will just have two types of components that can become P_3's.

Let us proceed to the description of the labels. First, we use one label to indicate whether a vertex belongs in the solution or not. For the collection of paths or cycles, we use three labels 0, 1, and 2, corresponding to the current degree of this vertex. For a collection of stars, we use three labels, c, 0, 1 where c labels a vertex that is a center of a star, 0 labels a vertex that is leaf of a star that is not connected to a center yet, and 1 labels a leaf of a star that is already connected to a center. For the clique of size 4, we only need two labels 0 and 1, where 0 means that the vertex should be in a K_4 but we do not know which one yet, and 1 means that we already found in which K_4 the vertex is. The crucial argument for this is the fact that if four vertices induce a K_4, then by the properties of a tree decomposition there exists a bag that contains the four vertices. For the paw, we use six labels c_0, c_1, d_f, d_w, ℓ_0, and ℓ_1. We call *leaf* the only vertex of the paw of degree 1. The label ℓ_0 (resp. ℓ_1) corresponds to a leaf of a paw that has degree 0 (resp. 1) in the currently processed graph. The label c_0 (resp. c_1) corresponds to a vertex of the cycle of a paw that is not (resp. is) connected to a leaf and for which we do not know yet the three vertices of the cycle. The label d_f corresponds to a vertex of the cycle of a paw for which we already know the three vertices of the cycle and that is *full*, i.e., it cannot be connected to a leaf anymore. The label d_w corresponds to a vertex of the cycle of a paw for which we know the three vertices of the cycle and that is waiting for a leaf to be connected to.

Dealing with the diamond is a bit more complicated. We see the diamond as two C_3 glued by an edge called *chord*. The crucial argument is that for each C_3, there exists a bag that contains the three vertices of the cycle and we "only" need to remember which edge of the cycle is the chord of the diamond. We use two labels for the vertices a_0 and a_1, and, this time, we also use three labels on the edges b_0, b_1, and b_2. Note that in a diamond, the number of edges is linear in the number of vertices and so, we can afford to label the edges. Namely, a_0 is used for vertices that can still be in a C_3 and a_1 is used for vertices that cannot be in a new C_3 anymore. The label b_0 is used for edges that are not in a C_3 yet, the label b_1 is used for chords for which we only found one of the two C_3's, and the label b_2 is used for edges that cannot be used anymore. Note that the endpoints of a chord belong to two C_3's, so when detecting the first one, these endpoints will remain labeled a_0 but the chord will be now labeled b_1.

Using all these labels, and updating them in a bottom-up fashion in a tree decomposition in a standard way, we obtain the following theorem.

Theorem 9. *If a nice tree decomposition of G of width w is given,* {chair}-Deletion *can be solved in time $2^{\mathcal{O}(w)} \cdot n$.*

9 A single-exponential algorithm for {banner}-TM-Deletion

Similarly as before, we start with a structural characterization of the graphs that exclude the banner as a (topological) minor.

Lemma 9. *Let G be a graph with at least five vertices.* banner $\not\preceq_{\mathsf{tm}} G$ *if and only if G is a cycle or $C_4 \not\preceq_{\mathsf{tm}} G$.*

Proof. Let G be a connected graph with $|V(G)| \geq 5$. It is straightforward to see that if G is a cycle or if $C_4 \not\preceq_{\mathsf{tm}} G$, then banner $\not\preceq_{\mathsf{tm}} G$. Conversely, assume that G excludes the banner as a minor and contains a cycle C of size at least four. As we can assume that G is connected and excludes the banner as a minor, we have that $V(G) = V(C)$. Indeed, if there exists a vertex $v \in V(G) \setminus V(C)$, then there exists a vertex $v' \in V(G') \setminus V(C)$ that is a neighbor of a vertex of $V(C)$, and so the graph $G[V(C) \cup \{v'\}]$ contains the banner as a topological minor. By assumption, we have that $|V(C)| = |V(G)| \geq 5$. Assume now for contradiction that G is *not* a cycle, that is, that $E(G) \setminus E(C)$ contains an edge e. Then, as $|V(C)| \geq 5$, there exists a cycle C' containing e, such that $4 \leq |V(C')| < |V(C)|$, and so G contains the banner as a minor, a contradiction. The lemma follows. $\square$

The idea of the algorithm is, as we did for {paw}-TM-Deletion (see Section 7), to use the structural properties given by Lemma 9 and to combine the rank-based approach and the standard dynamic programming techniques. As for {chair}-TM-Deletion (see Section 8), we will need, in particular, to label vertices that appear in components of size at most 4 and in components that are cycles. As for $\{C_4\}$-TM-Deletion (see Section 6), we also use one label for the solution and another label for the components that exclude C_4 as a minor. This means that, for this algorithm, we need 15 labels for the vertices and three labels for the edges. Because of this, again we do not provide the full formal description of the algorithm, and instead we provide a high-level description of how it works, reusing what we presented previously.

Namely, we explain how, starting from the algorithm for $\{C_4\}$-TM-Deletion given in Section 6, we obtain the desired algorithm for {banner}-TM-Deletion. As we did for {paw}-TM-Deletion, we modify the function $\mathbf{s} : X_t \to \{0,1\}$ to a function $\mathbf{s} : X_t \to \{0,1\} \cup L$, where L corresponds to the set of labels needed for detecting if a component is of size at most 4 or a cycle. We also add a function $\mathbf{r}_d$ for the labeling of the edges that appear in a diamond. Then, as we did for {paw}-TM-Deletion, when doing the dynamic programming operations, we use the rank-based approach for the elements of $\mathbf{s}^{-1}(1)$ and standard dynamic programming operations for the elements of $\mathbf{s}^{-1}(L)$. Doing this, we obtain the following theorem.

Theorem 10. *If a nice tree decomposition of G of width w is given,* $\{\mathsf{banner}\}$-Deletion *can be solved in time* $2^{\mathcal{O}(w)} \cdot n^{\mathcal{O}(1)}$.

10 Conclusions and further research

We presented single-exponential algorithms for $\{H\}$-TM-Deletion taking as parameter the treewidth of the input graph, when $H \in \{P_3, P_4, K_{1,i}, C_4, \mathsf{paw}, \mathsf{chair}, \mathsf{banner}\}$. These algorithms, combined with the results of [4,5], settle completely the complexity of $\{H\}$-M-Deletion when H is planar and connected; see Figure 1 for an illustration.

Concerning the topological minor version, in order to establish a dichotomy for $\{H\}$-TM-Deletion when H is planar and connected, it remains to obtain algorithms in time $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})})$ for the graphs H with maximum degree at least four, like the gem or the dart (see Figure 1), as for those graphs the algorithm in time $\mathcal{O}^*(2^{\mathcal{O}(\mathsf{tw} \cdot \log \mathsf{tw})})$ given in [4] cannot be applied.

Our algorithms for $\{C_i\}$-Deletion given here and in [4] may also be used to devise approximation algorithms for hitting or packing long cycles in a graph (in the spirit of [9] for other patterns), by using the fact that cycles of length at least i satisfy the Erdős-Pósa property [15]. We did not focus on optimizing the degree of the polynomials or the constants involved in our algorithms. Concerning the latter, one could use the framework by Lokshtanov et al. [19] to prove lower bounds based on the *Strong* ETH.

References

[1] Z. Bai, J. Tu, and Y. Shi. An improved algorithm for the vertex cover P_3 problem on graphs of bounded treewidth. *CoRR*, abs/1603.09448, 2016.

[2] J. Baste, I. Sau, and D. M. Thilikos. Optimal algorithms for hitting (topological) minors on graphs of bounded treewidth. In *Proc. of the 12th International Symposium on Parameterized and Exact Computation (IPEC)*, volume 89 of *LIPIcs*, pages 4:1–4:12, 2017.

[3] J. Baste, I. Sau, and D. M. Thilikos. A complexity dichotomy for hitting small planar minors parameterized by treewidth. In *Proc. of the 13th International Symposium on Parameterized and Exact Computation (IPEC)*, volume 115 of *LIPIcs*, pages 2:1–2:13, 2018.

[4] J. Baste, I. Sau, and D. M. Thilikos. Hitting minors on bounded treewidth graphs. I. General upper bounds. Manuscript submitted for publication. The contents correspond to those in Section 3 of `https://arxiv.org/abs/1704.07284`, 2019.

[5] J. Baste, I. Sau, and D. M. Thilikos. Hitting minors on bounded treewidth graphs. III. Lower bounds. Manuscript submitted for publication. The contents correspond to those in Section 5 of `https://arxiv.org/abs/1704.07284`, 2019.

[6] B. Bergougnoux and M. M. Kanté. Rank based approach on graphs with structured neighborhood. *CoRR*, abs/1805.11275, 2018.

[7] H. L. Bodlaender, M. Cygan, S. Kratsch, and J. Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Information and Computation*, 243:86–111, 2015.

[8] H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov, and M. Pilipczuk. A $c^k n$ 5-Approximation Algorithm for Treewidth. *SIAM Journal on Computing*, 45(2):317–378, 2016.

[9] D. Chatzidimitriou, J. Raymond, I. Sau, and D. M. Thilikos. An $O(\log OPT)$-Approximation for Covering and Packing Minor Models of Θ_r. *Algorithmica*, 80(4):1330–1356, 2018.

[10] B. Courcelle. The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs. *Information and Computation*, 85(1):12–75, 1990.

[11] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms.* Springer, 2015.

[12] M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving Connectivity Problems Parameterized by Treewidth in Single Exponential Time. In *Proc. of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 150–159, 2011.

[13] R. Diestel. *Graph Theory*, volume 173. Springer-Verlag, 4th edition, 2010.

[14] R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity.* Texts in Computer Science. Springer, 2013.

[15] S. Fiorini and A. Herinckx. A tighter Erdős-Pósa function for long cycles. *Journal of Graph Theory*, 77(2):111–116, 2014.

[16] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh. Efficient computation of representative families with applications in parameterized and exact algorithms. *Journal of the ACM*, 63(4):29:1–29:60, 2016.

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

[18] T. Kloks. *Treewidth. Computations and Approximations.* Springer-Verlag LNCS, 1994.

[19] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[20] D. Paik, S. M. Reddy, and S. Sahni. Deleting vertices to bound path length. *IEEE Transactions on Computers*, 43(9):1091–1096, 1994.

[21] J. Tu, L. Wu, J. Yuan, and L. Cui. On the vertex cover P_3 problem parameterized by treewidth. *Journal of Combinatorial Optimization*, 34(2):414–425, 2017.