UNIVERSITY OF LEEDS

This is a repository copy of *Subexponential Parameterized Algorithms for Hitting Subgraphs*.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/225445/

Version: Accepted Version

Proceedings Paper:
Lokshtanov, D., Panolan, F. orcid.org/0000-0001-6213-8687, Saurabh, S. et al. (2 more authors) (Accepted: 2025) Subexponential Parameterized Algorithms for Hitting Subgraphs. In: The 57th ACM Symposium on Theory of Computing (STOC 2025). The 57th ACM Symposium on Theory of Computing (STOC 2025), 23-27 Jun 2025, Prague, Czech Republic. ACM (In Press)

https://doi.org/10.1145/3717823.3718192

© Owner/Author | ACM 2025. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record will be published in The 57th ACM Symposium on Theory of Computing (STOC 2025), https://doi.org/10.1145/3717823.3718192.

Reuse
Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

White Rose
university consortium
Universities of Leeds, Sheffield & York

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

Subexponential Parameterized Algorithms for Hitting Subgraphs

Daniel Lokshtanov
University of California Santa Barbara
USA
daniello@ucsb.edu

Fahad Panolan
University of Leeds
UK
F.panolan@leeds.ac.uk

Saket Saurabh
The Institute of Mathematical Sciences, HBNI
India
saket@imsc.res.in

Jie Xue
New York University Shanghai
China
jiexue@nyu.edu

Meirav Zehavi
Ben-Gurion University
Israel
meiravze@bgu.ac.il

ABSTRACT

For a finite set $\mathcal{F}$ of graphs, the $\mathcal{F}$-Hitting problem aims to compute, for a given graph G (taken from some graph class $\mathcal{G}$) of n vertices (and m edges) and a parameter $k \in \mathbb{N}$, a set S of vertices in G such that $|S| \leq k$ and $G - S$ does not contain any subgraph isomorphic to a graph in $\mathcal{F}$. As a generic problem, $\mathcal{F}$-Hitting subsumes many fundamental vertex-deletion problems that are well-studied in the literature. The $\mathcal{F}$-Hitting problem admits a simple branching algorithm with running time $2^{O(k)} \cdot n^{O(1)}$, while it cannot be solved in $2^{o(k)} \cdot n^{O(1)}$ time on general graphs assuming the ETH, follows from the seminal work of Lewis and Yannakakis.

In this paper, we establish a general framework to design subexponential parameterized algorithms for the $\mathcal{F}$-Hitting problem on a broad family of graph classes. Specifically, our framework yields algorithms that solve $\mathcal{F}$-Hitting with running time $2^{O(k^c)} \cdot n + O(m)$ for a constant $c < 1$ on any graph class $\mathcal{G}$ that admits balanced separators whose size is (strongly) sublinear in the number of vertices and polynomial in the size of a maximum clique. Examples include all graph classes of polynomial expansion (e.g., planar graphs, bounded-genus graphs, minor-free graphs, etc.) and many important classes of geometric intersection graphs (e.g., map graphs, intersection graphs of any fat geometric objects, pseudo-disks, etc.). Our algorithms also apply to the *weighted* version of $\mathcal{F}$-Hitting, where each vertex of G has a weight and the goal is to compute the set S with a minimum weight that satisfies the desired conditions.

The core of our framework, which is our main technical contribution, is an intricate subexponential branching algorithm that reduces an instance of $\mathcal{F}$-Hitting (on the aforementioned graph classes) to $2^{O(k^c)}$ general hitting-set instances, where the Gaifman graph of each instance has treewidth $O(k^c)$, for some constant $c < 1$ depending on $\mathcal{F}$ and the graph class.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
STOC '25, June 23–27, 2025, Prague, Czechia
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1510-5/25/06
https://doi.org/10.1145/3717823.3718192

CCS CONCEPTS

• **Theory of computation → Parameterized complexity and exact algorithms**; **Graph algorithms analysis**.

KEYWORDS

Subgraph hitting, Subexponential paramterized algorithms, Separators, Generalized coloring numbers

ACM Reference Format:
Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. 2025. Subexponential Parameterized Algorithms for Hitting Subgraphs. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing (STOC '25), June 23–27, 2025, Prague, Czechia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3717823.3718192

1 INTRODUCTION

A vertex-deletion problem takes as input a graph G, and aims to delete from G a minimum number of vertices such that the resulting graph satisfies some property. In many vertex-deletion problems, the desired properties can be expressed as excluding a finite set of "forbidden" structures. This motivates the so-called $\mathcal{F}$-Hitting problem. Formally, for a finite set $\mathcal{F}$ of graphs (which represent the forbidden structures), the $\mathcal{F}$-Hitting problem is defined as follows.

> $\mathcal{F}$-Hitting **Parameter:** k
> **Input:** A graph G of n vertices and m edges (taken from some class $\mathcal{G}$) and a number $k \in \mathbb{N}$.
> **Goal:** Compute a set $S \subseteq V(G)$ of vertices such that $|S| \leq k$ and $G - S$ does not contain any graph in $\mathcal{F}$ as a subgraph.

As a generic problem, $\mathcal{F}$-Hitting subsumes many fundamental vertex-deletion problems that have been well-studied in the literature, such as Vertex Cover, Path Transversal [5, 25, 30], Short Cycle Hitting [23, 32, 33, 38], Component Order Connectivity [11, 24, 28], Degree Modulator [2, 21, 25], Treedepth Modulator [4, 15, 20], Clique Hitting [1, 16], Biclique Hitting [22], etc. We present in the appendix detailed descriptions of these problems as well as their background (see also the work [13]).

On general graphs (i.e., when the graph class $\mathcal{G}$ contains all graphs), the complexity of $\mathcal{F}$-Hitting is well-understood. For any (finite) $\mathcal{F}$, the problem admits a simple branching algorithm with running time $2^{O(k)} \cdot n^{O(1)}$ [6]. On the other hand, it follows from the reductions given in the work of Lewis and Yannakakis [31], $\mathcal{F}$-Hitting cannot be solved in $2^{o(k)} \cdot n^{O(1)}$ time or even $2^{o(n)}$ time for any $\mathcal{F}$ such that there are infinitely many graphs that do not contain any graphs in $\mathcal{F}$ as a subgraph[1], assuming the Exponential-Time Hypothesis. However, this does not rule out the existence of subexponential (parameterized) algorithms for $\mathcal{F}$-Hitting on restrictive graph classes. In fact, many special cases of $\mathcal{F}$-Hitting have been solved in subexponential time on various graph classes. For example, Vertex Cover, the simplest (nontrivial) case of $\mathcal{F}$-Hitting, was known to have $2^{o(k)} \cdot n^{O(1)}$-time algorithms on many graph classes, such as planar graphs [9], minor-free graphs [10], unit-disk graphs [18, 19], map graphs [9, 17, 19], and disk graphs [32]. Some other cases, such as Path Transversal, Clique Hitting, Component Order Connectivity, etc., were also known to admit subexponential parameterized algorithms on specific graph classes, mostly planar graphs, minor-free graphs, and (unit-)disk graphs [1, 9, 18, 19, 32]. Nevertheless, there has not been a comprehensive study on subexponential algorithms for the general $\mathcal{F}$-Hitting problem.

In this paper, we systematically study the $\mathcal{F}$-Hitting problem on a broad family of graph classes, in the context of subexponential parameterized algorithms. We show that on many graph classes where subexponential algorithms have been well-studied, $\mathcal{F}$-Hitting admits a subexponential parameterized algorithm for any $\mathcal{F}$. Before presenting our result in detail, we first discuss a difficulty in designing subexponential algorithms for $\mathcal{F}$-Hitting.

Difficulty of solving $\mathcal{F}$-Hitting in subexponential time. Compared to most vertex-deletion problems, designing subexponential algorithms for $\mathcal{F}$-Hitting seems especially challenging, due to lack of efficient algorithms for the problem on small-treewidth graphs. In fact, many fundamental vertex-deletion problems can be solved in $2^{O(t)}n^{O(1)}$ time or $t^{O(t)}n^{O(1)}$ time on graphs of treewidth t, by standard dynamic programming on tree decompositions. When the treewidth t is (strongly) sublinear, the running time is subexponential. Using this result as a cornerstone, a common approach for designing subexponential algorithms for such problems is to first reduce the treewidth t of the graph to sublinear, and then solve the problem in $2^{O(t)}n^{O(1)}$ time or $t^{O(t)}n^{O(1)}$ time. Unfortunately, this kind of approach does not work for $\mathcal{F}$-Hitting. Indeed, the best known algorithm for $\mathcal{F}$-Hitting on graphs of treewidth t, given by Cygan et al. [8], requires $2^{t^{f(\mathcal{F})}}n$ time for some function f. Even worse, Cygan et al. [8] also showed that such running time is necessary assuming the ETH. Therefore, even if the treewidth t is already sublinear, the algorithm for $\mathcal{F}$-Hitting can still run in superexponential time. This fact makes the task of solving $\mathcal{F}$-Hitting in subexponential time rather difficult and require new insights to the problem.

Other related work on $\mathcal{F}$-Hitting. The general $\mathcal{F}$-Hitting problem has been studied on graphs of bounded treewidth by Cygan et al. [8], who gave FPT algorithms (parameterized by treewidth) and lower bounds for the problem and its variants. Recently, Dvořák et al. [13] studied $\mathcal{F}$-Hitting in the perspective of approximation, showing that the local-search approach of Har-Peled and Quanrud [26] for Vertex Cover and Dominating Set on graph classes of polynomial expansion can be extended to obtain PTASes for $\mathcal{F}$-Hitting as well. They also gave a $(1+\varepsilon)$-approximation reduction from the problem on any bounded-expansion graph class to the same problem on bounded degree graphs within the class, resulting in lossy kernels for the problem. Bougeret et al. [3] considered the kernelization of $\mathcal{F}$-Hitting with various structural parameters. Finally, $\mathcal{F}$-Hitting is a special case of d-Hitting Set, and hence the algorithms for d-Hitting Set [7, 40] also apply to $\mathcal{F}$-Hitting. However, as the set system considered in $\mathcal{F}$-Hitting is implicitly defined and can have size $n^{O(\gamma)}$ where $\gamma = \max_{F \in \mathcal{F}} |V(F)|$, transferring the algorithms for d-Hitting Set to $\mathcal{F}$-Hitting usually results in an overhead of $n^{O(\gamma)}$.

1.1 Our result

Our main result is a general framework to design subexponential parameterized algorithms for $\mathcal{F}$-Hitting. In fact, our framework applies to the weighted version of $\mathcal{F}$-Hitting, where the vertices are associated with weights and we want to find the solution with the minimum total weight.

Weighted $\mathcal{F}$-Hitting **Parameter:** k

Input: A graph G of n vertices and m edges (taken from some class $\mathcal{G}$) together with a weight function $w : V(G) \to \mathbb{R}_{\geq 0}$, and a number $k \in \mathbb{N}$.

Goal: Compute a minimum-weight (under the function w) set $S \subseteq V(G)$ of vertices such that $|S| \leq k$ and $G - S$ does not contain any graph in $\mathcal{F}$ as a subgraph.

Another nice feature of our framework is that the algorithms it yields have running time not only subexponential in k but also linear in the size of the input graph G. More precisely, the running time of our algorithms is of the form $2^{O(k^c)} \cdot n + O(m)$ for some constant $c < 1$.

To present our result, we need to define the graph classes to which our framework applies. We define these graph classes in the language of *balanced separators*, which serve as a key ingredient in many efficient graph algorithms. For $\eta, \mu, \rho \geq 0$, we say a graph G admits *balanced (η, μ, ρ)-separators* if for every induced subgraph H of G, there exists $S \subseteq V(H)$ of size at most $\eta \cdot \omega^{\mu}(H) \cdot |V(H)|^{\rho}$ such that every connected component of $H - S$ contains at most $\frac{1}{2}|V(H)|$ vertices, where $\omega(H)$ denotes the size of a maximum clique in H. Throughout this paper, we denote by $\mathcal{G}(\eta, \mu, \rho)$ the class of all graphs admitting balanced (η, μ, ρ)-separators. Trivially, every graph admits balanced $(1, 0, 1)$-separators. Thus, we only consider the case $\rho < 1$. The main result of this paper is the following theorem.

Theorem 1. *Let $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ where $\eta, \mu \geq 0$ and $0 \leq \rho < 1$. Also, let $\mathcal{F}$ be a finite set of graphs. Then there exists a constant $c < 1$ (depending on η, μ, ρ, and $\mathcal{F}$) such that the Weighted $\mathcal{F}$-Hitting problem on $\mathcal{G}$ can be solved in $2^{O(k^c)} \cdot n + O(m)$ time.*

We now briefly discuss why the graph classes in Theorem 1 are interesting. The sub-classes of $\mathcal{G}(\eta, 0, \rho)$ for $\rho < 1$ are known

[1]Note that if there are only a finite number of graphs that do not contain any graphs in $\mathcal{F}$ as a subgraph, then $\mathcal{F}$-Hitting can be trivially solved in polynomial time.

as graph classes with *strongly sublinear separators*. Dvořák and Norin [14] showed that these classes are equivalent to graph classes of *polynomial expansion*, which is a well-studied special case of *bounded-expansion* graph classes introduced by Nešetřil and Ossona De Mendez [36, 37]. Important examples of polynomial-expansion graph classes include planar graphs, bounded-genus graphs, minor-free graphs, k-nearest neighbor graphs [35], greedy Euclidean spanners [29], etc. For a general $\mu \geq 0$, the classes $\mathcal{G}(\eta, \mu, \rho)$ with $\rho < 1$ in addition subsume a broad family of geometric intersection graphs. A *geometric intersection graph* is defined by a set of geometric objects in a Euclidean space, where the objects are the vertices and two vertices are connected by an edge if their corresponding objects intersect. Intersection graphs of any fat objects (i.e., convex geometric objects whose diameter-width ratio is bounded) and intersection graphs of pseudo-disks (i.e., topological disks in the plane satisfying that the boundaries of every pair of them are either disjoint or intersect twice) admit balanced (η, μ, ρ)-separators for $\rho < 1$ [13, 35]. These two families, in turn, cover many interesting cases of geometric intersection graphs, such as (unit-)disk graphs, ball graphs, hypercube graphs, map graphs, intersection graphs of non-crossing rectangles, etc. To summarize, the graph classes $\mathcal{G}(\eta, \mu, \rho)$ for $\rho < 1$ cover many common graph classes on which subexponential algorithms have been well-studied.

It is natural to ask whether the result in Theorem 1 can be extended to $\mathcal{F}$-Hitting with an infinite $\mathcal{F}$. A typical example is Feedback Vertex Set, where $\mathcal{F}$ consists of all cycles. Unfortunately, this seems impossible. Indeed, it was known [19] that Feedback Vertex Set does not admit subexponential FPT algorithms on unit-ball graphs with bounded ply, which belong to the class $\mathcal{G}(\eta, 0, \rho)$ for some η and $\rho < 1$. Another extension one may consider is to the Induced $\mathcal{F}$-Hitting problem, where the goal is to hit all *induced* subgraphs of G that are isomorphic to some graph in $\mathcal{F}$. Again, Theorem 1 is unlikely to hold for Induced $\mathcal{F}$-Hitting. Indeed, when $\mathcal{F}$ consists of a single edgeless graph of p vertices, Induced $\mathcal{F}$-Hitting (with $k = 0$) is equivalent to detecting an independent set of size p and thus an algorithm for Induced $\mathcal{F}$-Hitting on a graph class $\mathcal{G}$ with running time as in Theorem 1 would imply an FPT algorithm for Independent Set on $\mathcal{G}$. But Independent Set is W[1]-hard even on unit-disk graphs [7]. Note that while this rules out the possibility of extending Theorem 1 to Induced $\mathcal{F}$-Hitting, it might still be possible to obtain a weaker bound of $2^{O(k^c)} \cdot n^{f(\mathcal{F})}$ for $c < 1$ and some function f, which we leave as an interesting open question for future study.

1.2 Our framework and building blocks

In this section, we discuss our algorithmic framework for achieving Theorem 1 and its technical components. Let $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ where $\rho < 1$. In a high level, our framework solves a Weighted $\mathcal{F}$-Hitting instance (G, w, k) with $G \in \mathcal{G}$ through three general steps:

1. Reduce the size of the problem instance to $k^{O(1)}$ in linear time.
2. Reduce the $\mathcal{F}$-Hitting instance to a subexponential number of general (weighted) hitting-set instances, where the Gaifman graph of each instance has treewidth sublinear in k.
3. Solve each hitting-set instance efficiently using the sublinear treewidth of its Gaifman graph.

Among the three steps, the second one is the main step which is also the most difficult one, while the third step is standard. In what follows, we discuss these steps in detail.

To achieve Step 1, we give a polynomial kernel for the $\mathcal{F}$-Hitting problem on $\mathcal{G}$ that runs in *linear* time. Specifically, we show that one can compute in linear time an induced subgraph G' of the input graph G with size $k^{O(1)}$ such that solving the problem on G' is already sufficient for solving the problem on G. We say a set $S \subseteq V(G)$ is an *$\mathcal{F}$-hitting set* of G if $G - S$ does not contain any graph in $\mathcal{F}$ as a subgraph. Formally, we prove the following theorem.

Theorem 2. *Let $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ where $\eta, \mu \geq 0$ and $\rho < 1$. Also, let $\mathcal{F}$ be a finite set of graphs. There exists an algorithm that, for a given graph $G \in \mathcal{G}$ of n vertices and m edges together with a number $k \in \mathbb{N}$, computes in $k^{O(1)} \cdot n + O(m)$ time an induced subgraph G' of G with $|V(G')| = k^{O(1)}$ such that any $\mathcal{F}$-hitting set $S \subseteq V(G')$ of G' with $|S| \leq k$ is also an $\mathcal{F}$-hitting set of G.*

Note that polynomial kernels for d-Hitting Set are well-known [7]. Unfortunately, we cannot apply these kernels to obtain Theorem 2. The main reason is what we already mentioned in the introduction: in the $\mathcal{F}$-Hitting problem, the sets to be hit are implicit, and the number of these sets can be $n^{O(\gamma)}$, where $\gamma = \max_{F \in \mathcal{F}} |V(F)|$, so that we cannot afford to compute all of them. Besides this, another (less serious) difficulty here is that the instance obtained by the kernelization algorithm is required to be another $\mathcal{F}$-Hitting instance (whose underlying graph is an induced subgraph of G) rather than a general d-Hitting Set instance. Our approach for Theorem 2 is a variant of the sunflower-based kernel for d-Hitting Set, which can overcome these difficulties when properly combined with the linear-time first-order model checking algorithm of Dvořák et al. [12]. Thanks to Theorem 2, it suffices to design algorithms with subexponential *XP* running time, i.e., $n^{O(k^c)}$ time for $c < 1$.

Step 2 is the core of our framework. It is achieved by an intricate branching algorithm, which is the main technical contribution of this paper. Let $\mathcal{X}$ be a collection of sets (in which the elements belong to the same universe). The *Gaifman graph* of $\mathcal{X}$ is the graph with vertex set $\bigcup_{X \in \mathcal{X}} X$ where two vertices u and v are connected by an edge if $u, v \in X$ for some $X \in \mathcal{X}$. We say a set S *hits* $\mathcal{X}$ if $S \cap X \neq \emptyset$ for all $X \in \mathcal{X}$. The algorithm for Step 2 is stated in the following theorem.

Theorem 3. *Let $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ where $\eta, \mu \geq 0$ and $\rho < 1$. Also, let $\mathcal{F}$ be a finite set of graphs. Then there exists a constant $c < 1$ (depending on η, μ, ρ, and $\mathcal{F}$) such that for a given graph $G \in \mathcal{G}$ of n vertices and a parameter $k \in \mathbb{N}$, one can construct in $n^{O(k^c)}$ time $t = 2^{O(k^c)}$ collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ of subsets of $V(G)$ satisfying the following conditions.*

- *For any $S \subseteq V(G)$ with $|S| \leq k$, S is an $\mathcal{F}$-hitting set of G iff S hits $\mathcal{X}_i$ for some $i \in [t]$.*
- *The Gaifman graph of $\mathcal{X}_i$ has treewidth $O(k^c)$, for all $i \in [t]$.*
- *$|\mathcal{X}_i| = k^{O(1)}$ for all $i \in [t]$.*

The proof of Theorem 3 is highly nontrivial, which combines theories of sparse graphs, sunflowers, tree decomposition, branching algorithms, together with novel insights to the $\mathcal{F}$-Hitting problem itself. In the proof, we introduce interesting combinatorial results for the set systems considered in the $\mathcal{F}$-Hitting problem (i.e., the vertex sets of all subgraphs isomorphic to some graph in $\mathcal{F}$), which are of independent interest and might be useful for understanding the structure of such set systems. Using Theorem 2, one can straightforwardly improve the running time of Theorem 3 to $2^{O(k^c)} \cdot n + O(m)$, yielding the following result.

Corollary 4. *Let $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ where $\eta, \mu \geq 0$ and $\rho < 1$. Also, let $\mathcal{F}$ be a finite set of graphs. Then there exists a constant $c < 1$ (depending on η, μ, ρ, and $\mathcal{F}$) such that for a given graph $G \in \mathcal{G}$ of n vertices and a parameter $k \in \mathbb{N}$, one can construct in $2^{O(k^c)} \cdot n+O(m)$ time $t = 2^{O(k^c)}$ collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ of subsets of $V(G)$ satisfying the following conditions.*

- *For any $S \subseteq V(G)$ with $|S| \leq k$, S is an $\mathcal{F}$-hitting set of G iff S hits $\mathcal{X}_i$ for some $i \in [t]$.*
- *The Gaifman graph of $\mathcal{X}_i$ has treewidth $O(k^c)$, for all $i \in [t]$.*
- *$|\mathcal{X}_i| = k^{O(1)}$ for all $i \in [t]$.*

Proof. We first apply Theorem 2 on G and k to obtain the induced subgraph G'. Then we apply Theorem 3 on G' and k to obtain the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ of subsets of $V(G')$, which are also subsets of $V(G)$. Since $|V(G')| = k^{O(1)}$ by Theorem 2, the total time cost is $k^{O(1)} \cdot n+O(m)+k^{O(k^p)}$ for some constant $p < 1$. Choosing an arbitrary constant $c \in (p, 1)$, the running time is bounded by $2^{O(k^c)} \cdot n + O(m)$. The bounds on the sizes of $\mathcal{X}_1, \dots, \mathcal{X}_t$ and the treewidth of the Gaifman graphs directly follow from Theorem 3. It suffices to show that a set $S \subseteq V(G)$ with $|S| \leq k$ is an $\mathcal{F}$-hitting set of G iff S hits $\mathcal{X}_i$ for some $i \in [t]$. If S is an $\mathcal{F}$-hitting set of G, then $S \cap V(G')$ is an $\mathcal{F}$-hitting set of G'. Thus $S \cap V(G')$ hits $\mathcal{X}_i$ for some $i \in [t]$ by Theorem 3, which implies that S hits $\mathcal{X}_i$. On the other hand, if S hits $\mathcal{X}_i$ for some $i \in [t]$, then S is an $\mathcal{F}$-hitting set of G' by Theorem 3, which is in turn an $\mathcal{F}$-hitting set of G by Theorem 2 since $|S| \leq k$. ◻

Step 3 is achieved by standard dynamic programming on tree decomposition. For each hitting-set instance $\mathcal{X}_i$ obtained in Corollary 4, we build a tree decomposition of the Gaifman graph of $\mathcal{X}_i$ of width $O(k^c)$, which can be done using standard constant-approximation algorithms for treewidth [7]. Then we apply dynamic programming to compute a minimum-weight (under the weight function w) hitting set S_i for $\mathcal{X}_i$ satisfying $|S_i| \leq k$. Finally, we return the set S_i with the minimum total weight among $S_1, \dots, S_t$. The first condition in Corollary 4 guarantees that S_i is an optimal solution for the Weighted $\mathcal{F}$-Hitting instance (G, w, k).

2 OVERVIEW

Due to the limited space, in this paper, we only give an overview of our algorithms and the underlying ideas/techniques. The detailed proofs can be found in the full version of the paper [34].

The kernelization algorithm in Theorem 2 is relatively simple. So we only focus on our main technical result, the reduction algorithm in Theorem 3. Before the discussion, let us recall a standard notion called *weak coloring numbers*. Let G be a graph and σ be an ordering of $V(G)$. For $u, v \in V(G)$, we write $u <_\sigma v$ if u is before v under the ordering σ. The notations $>_\sigma$, $\leq_\sigma$, $\geq_\sigma$ are defined similarly. For an integer $r \geq 0$, u is *weakly r-reachable* from v *under σ* if there is a path π between v and u of length at most r such that u is the largest vertex on π under the ordering σ, i.e., $u \geq_\sigma w$ for all $w \in V(\pi)$. Let $\mathrm{WR}_r(G, \sigma, v)$ denote the set of vertices in G that are weakly r-reachable from v under σ. The *weak r-coloring number* of G *under σ* is defined as $\mathrm{wcol}_r(G, \sigma) = \max_{v \in V(G)} |\mathrm{WR}_r(G, \sigma, v)|$. Then the *weak r-coloring number* of G is defined as $\mathrm{wcol}_r(G) = \min_{\sigma \in \Sigma(G)} \mathrm{wcol}_r(G, \sigma)$ where $\Sigma(G)$ is the set of all orderings of $V(G)$. It is well-known [36, 41] that a graph class $\mathcal{G}$ is of bounded expansion iff there is a function $f : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{wcol}_r(G) \leq f(r)$ for all $G \in \mathcal{G}$ and all $r \in \mathbb{N}$.

For simplicity, we only discuss the algorithm of Theorem 3 in a special case where $\mu = 0$ (i.e., $\mathcal{G}$ is of polynomial expansion) and $\mathcal{F}$ only contains a single graph F. The same algorithm directly extends to the case where $\mathcal{F}$ consists of multiple graphs. Further extending it to a general $\mathcal{G} \subseteq \mathcal{G}(\eta, \mu, \rho)$ is also not difficult, by adapting some standard techniques. Note that if $\mathcal{G} \subseteq \mathcal{G}(\eta, 0, \rho)$, then for any fixed $r \in \mathbb{N}$, we have $\mathrm{wcol}_r(G) = O(1)$ for all $G \in \mathcal{G}$.

The first simple observation we have is that one can assume F is connected without loss of generality. To see this, consider a graph $G \in \mathcal{G}$. We say a graph is *F-free* if it does not contain any subgraph isomorphic to F. Let G^+ (resp., F^+) be the graph obtained from G (resp., F) by adding a new vertex with edges to all other vertices. Now $G^+ \in \mathcal{G}(\eta + 1, 0, \rho)$ and F^+ is connected. Furthermore, one can easily verify that for any $S \subseteq V(G)$, $G^+ - S$ is F^+-free iff $G - S$ is F-free. As long as Theorem 3 works for $\mathcal{G}(\eta + 1, 0, \rho)$ and $\mathcal{F}^+ = \{F^+\}$, we can apply it to compute the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ of subsets of $V(G^+)$ satisfying the desired conditions. Define $\mathcal{X}'_i = \{X \cap V(G) : X \in \mathcal{X}_i\}$ for $i \in [t]$. It turns out that $\mathcal{X}'_1, \dots, \mathcal{X}'_t$ are collections of subsets of $V(G)$ that satisfy the desired conditions for G and $\mathcal{F}$.

An *F-copy* in G refers to a pair (H, π) where H is a subgraph of G and $\pi : V(H) \to V(F)$ is an isomorphism between H and F. As is usual in hitting-set problems [27, 39], we consider and branch on *sunflowers* in the set system (which in our setting consists of the vertex sets of the F-copies in G). Recall that sets $V_1, \dots, V_r$ form a *sunflower* if there exists a set X such that $X \subseteq V_i$ for all $i \in [r]$ and $V_1 \backslash X, \dots, V_r \backslash X$ are disjoint; X is called the *core* of the sunflower and r is the size of the sunflower.

As the sets in our problem are vertex sets of subgraphs of G, it is more convenient to consider sunflowers with additional structures related to the graph. We say F-copies $(H_1, \pi_1), \dots, (H_r, \pi_r)$ in G form a *structured sunflower* if $V(H_1), \dots, V(H_r)$ form a sunflower with core X and $(\pi_1)_{|X} = \cdots = (\pi_r)_{|X}$; the core of the structured sunflower is the pair (X, f) where $f : X \to V(F)$ is the unique map satisfying $f = (\pi_1)_{|X} = \cdots = (\pi_r)_{|X}$. A pair (X, f) is a *heavy core* if it is the core of a structured sunflower of size $\gamma^{|X|}\delta$, where $\gamma = |V(F)|$ and δ is a parameter to be determined. The reason why we pick $\gamma^{|X|}\delta$ as the threshold will be clear later. Note that we do not require the F-copies in a structured sunflower to be distinct. Therefore, for an F-copy (H, π) in G, the pair $(V(H), \pi)$ is a heavy core, because it is the core of the structured sunflower $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ where $\Delta = \gamma^{|V(H)|}\delta = \gamma^\gamma\delta$ and $(H_1, \pi_1) = \cdots = (H_\Delta, \pi_\Delta) = (H, \pi)$.

To get some basic idea about how $\mathcal{X}_1, \dots, \mathcal{X}_t$ in Theorem 3 are generated, let us first consider a trivial branching algorithm, which can generate $\mathcal{X}_1, \dots, \mathcal{X}_t$ satisfying the first condition in Theorem 3 without any guarantee on the running time, the number t, or the treewidth. Imagine there is some (unknown) $\mathcal{F}$-hitting set S of G. The branching algorithm essentially guesses whether each heavy core is hit by the solution or not. It maintains a set $U \subseteq V(G)$ and a collection $\mathcal{X}$ of subsets of $V(G)$. The vertices in U are "undeletable" vertices, namely, the vertices that are not supposed to be in S. On the other hand, the sets in $\mathcal{X}$ are supposed to be hit by S. Initially, $U = \emptyset$ and $\mathcal{X} = \emptyset$. Then it calls the function $\textsc{Branch}(U, \mathcal{X})$, which works as follows.

- Pick a heavy core (X, f) satisfying $X \not\subseteq U$ and $X \notin \mathcal{X}$. If such a heavy core does not exist and $G[U]$ is F-free, then create a new $\mathcal{X}_i = \{X \backslash U : X \in \mathcal{X}\}$, and return to the last level.
- Branch on (X, f) in two ways (i.e., guess whether S hits X or not):
  - "Yes" branch (guess $S \cap X \neq \emptyset$): recursively call $\textsc{Branch}(U, \mathcal{X} \cup \{X\})$.
  - "No" branch (guess $S \cap X = \emptyset$): recursively call $\textsc{Branch}(U \cup X, \mathcal{X})$.

Let $\mathcal{X}_1, \dots, \mathcal{X}_t$ be the collection generated by the above procedure. One can easily check that a subset $S \subseteq V(G)$ is an $\mathcal{F}$-hitting set of G iff S hits $\mathcal{X}_i$ for some $i \in [t]$. Indeed, if S is an $\mathcal{F}$-hitting set of G and the algorithm makes the right decision in each step (i.e., makes the "yes" decision whenever $S \cap X \neq \emptyset$ and the "no" decision whenever $S \cap X = \emptyset$), at the end of that branch a collection $\mathcal{X}_i = \{Y \backslash U : Y \in \mathcal{X}\}$ is created and S hits $\mathcal{X}_i$. On the other hand, if $S \subseteq V(G)$ is a set that hits some $\mathcal{X}_i$, then S hits all F-copies in G. Why? Note that $\mathcal{X}_i = \{X \backslash U : X \in \mathcal{X}\}$. At the point we create $\mathcal{X}_i$, we have $X \in \mathcal{X}$ for all heavy cores (X, f) with $X \not\subseteq U$, and in particular $V(H) \in \mathcal{X}$ for all F-copies (H, π) with $V(H) \not\subseteq U$. By assumption, S hits these F-copies. Furthermore, as $G[U]$ is F-free, there is no F-copy (H, π) with $V(H) \subseteq U$. Thus, S hits all F-copies in G.

Of course, this trivial branching procedure can only provide us some intuition about how the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ are generated. It guarantees neither sublinear treewidth of the Gaifman graphs of $\mathcal{X}_1, \dots, \mathcal{X}_t$ nor subexponential bound on t. Thus, we are still far from proving Theorem 3. In the following two sections, we shall focus on how to achieve sublinear treewidth (Section 2.1) and subexponential branching (Section 2.2), respectively. Both parts are quite technical. Interestingly, to obtain sublinear treewidth, we only need to slightly modify the trivial branching algorithm, and the main challenge lies in the proof of a structural lemma for the Gaifman graph of certain heavy cores (Lemma 5). For subexponential branching, however, we have to further elaborate the branching algorithm significantly, with an involved analysis.

In this overview, we ignore the requirements of Theorem 3 on the running time of the algorithm and the sizes of $\mathcal{X}_1, \dots, \mathcal{X}_t$. It turns out that these requirements can be achieved almost for free as long as the algorithm admits a subexponential branching tree.

2.1 How to achieve sublinear treewidth

Recall that we want the Gaifman graph of each $\mathcal{X}_i$ to have treewidth sublinear in k. To see how to achieve the sublinear treewidth bound, let us introduce some additional notions. For two heavy cores (X, f) and (Y, g), we write $(X, f) \prec (Y, g)$ if $X \subsetneq Y$ and $f = g_{|X}$, and write $(X, f) \preceq (Y, g)$ if $(X, f) \prec (Y, g)$ or $(X, f) = (Y, g)$. Clearly, $\prec$ is a partial order among all heavy cores. For a subset $U \subseteq V(G)$, we say a heavy core (X, f) is *U-minimal* if $X \not\subseteq U$ and for any heavy core (Y, g) with $(Y, g) \prec (X, f)$, we have $Y \subseteq U$. The key to achieve sublinear treewidth is the following important structural lemma for U-minimal heavy cores.

Lemma 5. *Suppose $\delta > \mathrm{wcol}_\gamma(G)$. Then for any subset $U \subseteq V(G)$ and any U-minimal heavy cores $(X_1, f_1), \dots, (X_r, f_r)$ in G, the Gaifman graph of $\{X_1 \backslash U, \dots, X_r \backslash U\}$ has treewidth $\delta^{O(1)} \cdot k^c$ for some constant $c < 1$, where k is the size of a minimum hitting set of $\{X_1 \backslash U, \dots, X_r \backslash U\}$. Here c and the constant hidden in $O(\cdot)$ only depend on F and the polynomial-expansion graph class $\mathcal{G}$ from which G is drawn.*

The proof of Lemma 5 is technical. Before giving a sketch of the proof, we first explain how this lemma helps us. Recall the branching procedure discussed before. The first observation is that we actually only need to branch on U-minimal heavy cores. Specifically, we require the heavy core (X, f) picked in the first step of $\textsc{Branch}(U, \mathcal{X})$ to be *U-minimal*. With this modification, the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ generated still satisfy the first condition in Theorem 3, because a set $S \subseteq V(G) \backslash U$ hits all heavy cores if and only if it hits all U-minimal heavy cores. More importantly, the Gaifman graph of each $\mathcal{X}_i$ has treewidth sublinear in the size of a minimum hitting set of $\mathcal{X}_i$, by Lemma 5. The second observation is that if the size of a minimum hitting set of a collection $\mathcal{X}_i$ is larger than k, then we can simply discard $\mathcal{X}_i$. This is because the first condition in Theorem 3 only considers $S \subseteq V(G)$ with $|S| \leq k$. Therefore, only keeping the collections $\mathcal{X}_i$ whose minimum hitting set has size at most k is sufficient. In this way, we can guarantee that the Gaifman graphs of all $\mathcal{X}_i$ have treewidth sublinear in k, as required in Theorem 3.

Proof sketch of Lemma 5. In the rest of this section, we provide a high-level overview for the proof of Lemma 5. We first need to show the following auxiliary lemma, which essentially states that whenever there are many heavy cores $(X_1, f_1), \dots, (X_p, f_p)$ forming a large sunflower, one can find certain smaller heavy cores inside each X_i. This lemma heavily relies on the threshold $\gamma^{|X|}\delta$ we chose for a heavy core (X, f).

Lemma 6. *Let $p = \gamma^\gamma \delta$. Suppose $(X_1, f_1), \dots, (X_p, f_p)$ are heavy cores in G satisfying that $|X_1| = \cdots = |X_p|$ and $X_1, \dots, X_p$ form a sunflower with core K where $(f_1)_{|K} = \cdots = (f_p)_{|K}$. Define $f : K \to V(F)$ as the unique map satisfying $f = (f_1)_{|K} = \cdots = (f_p)_{|K}$. Then for any $i \in [p]$ and any set C of connected components of $F - f(K)$, (X_i^C, f_i^C) is a heavy core in G where $X_i^C = K \cup (\bigcup_{C \in \mathcal{C}} f_i^{-1}(V(C)))$ and $f_i^C = (f_i)_{|X_i^C}$.*

Proof sketch. We only need to consider the pair (X_1^C, f_1^C). If C contains all connected components of $F - f(K)$, then $(X_1^C, f_1^C) = (X_1, f_1)$ and we are done. Otherwise, $|X_1^C| < |X_1| = \cdots = |X_p|$. Our goal is to find a structured sunflower $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ in G for $\Delta = \gamma^{|X_1^C|}\delta$ whose core is (X_1^C, f_1^C). We say an F-copy (H, π) is a *candidate* if $X_1^C \subseteq V(H)$ and $f_1^C = \pi_{|X_1^C}$. Then our task becomes finding Δ candidates whose vertex sets are disjoint outside X_1^C.

Where do these candidates come from? In fact, we can construct them from the structured sunflowers that witness the heavy cores $(X_1, f_1), \dots, (X_p, f_p)$. Let $A = f(K) \cup (\bigcup_{C \in \mathcal{C}} V(C))$ and $B = f(K) \cup (V(F)\backslash A)$. Then $V(F)$ is the disjoint union of $A\backslash B$, $f(K)$, and $B\backslash A$. Also, note that there is no edge in F between $A\backslash B$ and $B\backslash A$. Consider an F-copy (P, ϕ) in the structured sunflower that witnesses (X_1, f_1) and another F-copy (Q, ψ) in the structured sunflower that witnesses (X_i, f_i) for some $i \in [p]$. The key observation is the following: if $\phi^{-1}(A\backslash B) \cap \psi^{-1}(B\backslash A) = \emptyset$, then $H = P[\phi^{-1}(A)] \cup Q[\psi^{-1}(B)]$ is isomorphic to F with the isomorphism $\pi : V(H) \to V(F)$ defined as $\pi(v) = \phi(v)$ for $v \in \pi^{-1}(A)$ and $\pi(v) = \psi(v)$ for $v \in \pi^{-1}(B)$ — one can easily verify that π is well-defined and is an isomorphism — and furthermore (H, π) is a candidate (for convenience, we call $P[\phi^{-1}(A)]$ the *A-half* and $Q[\psi^{-1}(B)]$ the *B-half* of the candidate). We use this observation to construct the candidates. The structured sunflower that witnesses (X_1, f_1) has size $\gamma^{|X_1|}\delta > \Delta$, so we can take Δ F-copies $(P_1, \phi_1), \dots, (P_\Delta, \phi_\Delta)$ from it. Then we use $P_1[\phi_1^{-1}(A)], \dots, P_\Delta[\phi_\Delta^{-1}(A)]$ as the A-halves of the candidates, each of which will be "glued" with a B-half to obtain a complete candidate. We briefly discuss how to find the B-halves. Outside K, the B-halves should be disjoint from each other and also disjoint from the A-halves. Suppose we already found the B-halves for $P_1[\phi_1^{-1}(A)], \dots, P_{i-1}[\phi_{i-1}^{-1}(A)]$ and are now looking for the B-half for $P_i[\phi_i^{-1}(A)]$. We say a set $S \subseteq V(G)$ is *safe* if it is disjoint from all A-halves and the B-halves we have found. We first find $j \in [p]$ such that $X_j\backslash K$ is safe. Such an index exists since $X_1\backslash K, \dots, X_p\backslash K$ are disjoint and p is sufficiently large. Then we further find an F-copy (Q, ψ) in the structured sunflower that witnesses (X_j, f_j) such that $V(Q)\backslash X_j$ is safe. This is possible because the size of the structured sunflower is $\gamma^{|X_j|}\delta$, which is much larger than $\Delta = \gamma^{|X_1^C|}\delta$ as $|X_1^C| < |X_j|$. Now we use $Q[\psi^{-1}(B)]$ as the B-half for $P_i[\phi_i^{-1}(A)]$. In this way, we can successfully find all B-halves. □

Note that the above lemma directly implies the "sparseness" of U-minimal heavy cores outside U: every vertex in $V(G)\backslash U$ hits at most $\delta^{O(1)}$ (distinct) U-minimal heavy cores in G. Why? Suppose a vertex $v \in V(G)\backslash U$ hits too many U-minimal heavy cores. By the sunflower lemma and Pigeonhole principle, among these U-minimal heavy cores, we can find $(X_1, f_1), \dots, (X_p, f_p)$ satisfying the condition in the lemma. The core K of the sunflower $X_1, \dots, X_p$ is nonempty as $v \in \bigcap_{i=1}^p X_i = K$. Also, $K \not\subseteq U$, since $v \in K\backslash U$. Applying the lemma with $\mathcal{C} = \emptyset$, we see that (K, f) is a heavy core, which contradicts the U-minimality of $(X_1, f_1), \dots, (X_p, f_p)$ because $(K, f) \prec (X_i, f_i)$ for all $i \in [p]$. We omit the calculation for the maximum number of U-minimal heavy cores v can hit, but the number turns out to be $\delta^{O(1)}$.

Now we sketch the proof of Lemma 5. Let $(X_1, f_1), \dots, (X_r, f_r)$ be as in the lemma, and k be the size of a minimum hitting set of $\{X_1\backslash U, \dots, X_r\backslash U\}$. As argued above, each vertex can hit $\delta^{O(1)}$ sets in $\{X_1\backslash U, \dots, X_r\backslash U\}$, which implies $|\bigcup_{i=1}^r (X_i\backslash U)| = \delta^{O(1)}k$. Fix an ordering σ of the vertices of G such that $\mathrm{wcol}_\gamma(G, \sigma) = \mathrm{wcol}_\gamma(G)$. Let G' be a supergraph of G obtained by adding edges to connect pairs of vertices in which one is weakly γ-reachable (under σ) from the other, i.e., $V(G') = V(G)$ and $E(G') = E(G) \cup \{(u, v) : u \in \mathrm{WR}_\gamma(G, \sigma, v)\}$. It turns out that the graph G' also admits strongly sublinear separators, which implies the treewidth of $G'[\bigcup_{i=1}^r (X_i\backslash U)]$ is sublinear in $|\bigcup_{i=1}^r (X_i\backslash U)| = \delta^{O(1)}k$, i.e., bounded by $\delta^{O(1)}k^c$ for some $c < 1$. Let (T, β) be a minimum-width tree decomposition of $G'[\bigcup_{i=1}^r (X_i\backslash U)]$. Our goal is to modify (T, β) to a tree decomposition of the Gaifman graph G^* of $\{X_1\backslash U, \dots, X_r\backslash U\}$, without increasing its width too much. The modification is done as follows. For each $i \in [r]$, we pick a node $t_i \in V(T)$ such that $\beta(t_i) \cap (X_i\backslash U) \neq \emptyset$. For two nodes $t, t' \in V(T)$, denote by $\pi_{t,t'}$ as the (unique) path in T connecting t and t'. Then for each $t \in V(T)$ and each $i \in [r]$, define $\beta_i^*(t)$ as the set of all vertices $v \in X_i\backslash U$ such that t is on the path $\pi_{t_i,t'}$ for some node $t' \in V(T)$ with $v \in \beta(t')$. Set $\beta^*(t) = \bigcup_{i=1}^r \beta_i^*(t)$ for all $t \in V(T)$.

It is easy to verify that (T, β^*) is a tree decomposition of G^*. The tricky part is to bound its width. We want $|\beta^*(t)| = \delta^{O(1)} \cdot |\beta(t)|$ for every $t \in V(T)$. Fix a node $t \in V(T)$. Let $I^* = \{i \in [r] : \beta_i^*(t) \neq \emptyset\}$. Since $|\beta_i^*(t)| \leq |X_i\backslash U| \leq \gamma$, we have $|\beta^*(t)| \leq \gamma|I^*|$ and thus it suffices to show $|I^*| = \delta^{O(1)} \cdot |\beta(t)|$. Now let $I = \{i \in [r] : \beta(t) \cap (X_i\backslash U) \neq \emptyset\}$. We have seen that each vertex in $\beta(t)$ can hit at most $\delta^{O(1)}$ sets in $\{X_1\backslash U, \dots, X_r\backslash U\}$. So $|I| = \delta^{O(1)} \cdot |\beta(t)|$ and we only need to show $|I^*\backslash I| = \delta^{O(1)} \cdot |\beta(t)|$.

The high-level plan for bounding $|I^*\backslash I|$ is to apply a charging argument as follows. For each $i \in I^*\backslash I$, we shall pick two vertices $v_i, v_i' \in X_i\backslash U$ and charge i to a set $Y_i \subseteq X_i$ satisfying that $f_i(Y_i)$ separates $f_i(v_i)$ and $f_i(v_i')$ in F, i.e., $f_i(v_i)$ and $f_i(v_i')$ belong to different connected components of $F - f_i(Y_i)$. By a careful construction, we can guarantee that the number of distinct Y_i's is small. Then using Lemma 6, we can show that each set $Y \subseteq V(G)$ does not get charged too many times. These two conditions together bound the size of $I^*\backslash I$.

Consider an index $i \in I^*\backslash I$. We have $\beta_i^*(t) \neq \emptyset$ but $\beta(t) \cap (X_i\backslash U) = \emptyset$. By the choice of t_i, $\beta(t_i) \cap (X_i\backslash U) \neq \emptyset$ and so we pick a vertex $v_i \in \beta(t_i) \cap (X_i\backslash U)$. On the other hand, as $\beta_i^*(t) \neq \emptyset$, we can pick another vertex $v_i' \in \beta_i^*(t) \subseteq X_i\backslash U$. Note that $v_i, v_i' \notin \beta(t)$, since $\beta(t) \cap (X_i\backslash U) = \emptyset$. By the properties of a tree decomposition, the nodes $s \in T$ with $v_i \in \beta(s)$ (resp., $v_i' \in \beta(s)$) are connected in T, and we call the subtree of T formed by these nodes the *v_i-area* (resp., *v_i'-area*) for convenience. Why do v_i, v_i' appear in $\beta_i^*(t)$ but not $\beta(t)$? The only reason is that the v_i-area and the v_i'-area belong to different connected components in the forest $T - \{t\}$. We can show that if $\delta > \mathrm{wcol}_\gamma(G)$, then this situation happens only when $f_i(X_i \cap \beta(t))$ separates $f_i(v_i)$ and $f_i(v_i')$ in F. We omit the details of this argument. Now a natural idea is to directly set $Y_i = X_i \cap \beta(t)$. But this seems a bad idea, as the number of distinct Y_i's cannot be bounded with this definition. Therefore, we need to construct Y_i from $X_i \cap \beta(t)$ with an additional step as follows. Let Π be the set of all simple paths in F from $f_i(v_i)$ to a vertex in $f_i(X_i \cap \beta(t))$ in which all internal nodes are in $V(F)\backslash f_i(X_i \cap \beta(t))$. For every $u \in X_i$, we include u in Y_i if there exists $\pi \in \Pi$ such that u is the largest vertex (under the ordering σ) in $f_i^{-1}(V(\pi))$. It turns out that $f_i(Y_i)$ also separates $f_i(v_i)$ and $f_i(v_i')$ in F. Furthermore, Y_i has a very nice property: $Y_i \subseteq \mathrm{WR}_\gamma(G, \sigma, v)$ for some $v \in \beta(t)$. Again, we omit the proof of this property in this overview.

With the above construction, how many distinct Y_i's can there be? For each $v \in \beta(t)$, we have $|\mathrm{WR}_\gamma(G, \sigma, v)| \leq \mathrm{wcol}_\gamma(G, \sigma) = \mathrm{wcol}_\gamma(G)$. Thus, the nice property of each Y_i and the fact $|Y_i| \leq \gamma$ guarantee that the number of distinct Y_i's is at most $\mathrm{wcol}_\gamma^\gamma(G) \cdot |\beta(t)|$, which is $O(|\beta(t)|)$. Now it suffices to bound the number of times a set $Y \subseteq V(G)$ gets charged. The intuition is the following. Assume there are too many indices $i \in I^* \backslash I$ that are charged to the same set Y, in order to deduce a contradiction. Then among the heavy cores (X_i, f_i) corresponding to the indices charged to Y, we can find $p = \gamma^\gamma \delta$ of them satisfying the conditions in Lemma 6, by the sunflower lemma and Pigeonhole principle. Without loss of generality, suppose they are $(X_1, f_1), \dots, (X_p, f_p)$, where $X_1, \dots, X_p$ form a sunflower with core K and $f = (f_1)_{|K} = \cdots = (f_p)_{|K}$. Then $Y = Y_1 = \cdots = Y_p$. Applying Lemma 6 with $C = \emptyset$, we see that (K, f) is a heavy core in G. Since $(K, f) \prec (X_i, f_i)$ for all $i \in [p]$, we must have $K \subseteq U$, for otherwise $(X_1, f_1), \dots, (X_p, f_p)$ are not U-minimal. As all $i \in [p]$ are charged to Y, we have $Y \subseteq \bigcap_{i=1}^p X_i = K$. Recall the vertices $v_i, v_i' \in X_i \backslash U$ we picked when constructing Y_i. We just consider v_1 and v_1'. Neither $f(v_1)$ nor $f(v_1')$ is contained in $f(K)$, because $K \subseteq U$ and $v_i, v_i' \in X_i \backslash U$. Let C (resp., C') be the connected component of $F - f(K)$ containing v_1 (resp., v_1'). Note that $C \neq C'$. Indeed, $f(v_1)$ and $f(v_1')$ lie in different connected components of $F - f_1(Y_1) = F - f_1(Y)$ and thus lie in different connected components of $F - f(K)$, since $f_1(Y) \subseteq f_1(K) = f(K)$. Set $\mathcal{C} = \{C\}$ and let $(X_1^{\mathcal{C}}, f_1^{\mathcal{C}})$ as defined in Lemma 6. Lemma 6 shows that $(X_1^{\mathcal{C}}, f_1^{\mathcal{C}})$ is a heavy core. Observe that $X_1^{\mathcal{C}} \nsubseteq U$, since $v_1 \in X_1^{\mathcal{C}} \backslash U$. On the other hand, $v_1' \notin X_1^{\mathcal{C}}$, as $C' \notin \mathcal{C}$. Thus, $X_1^{\mathcal{C}} \subsetneq X_1$ and $(X_1^{\mathcal{C}}, f_1^{\mathcal{C}}) \prec (X_1, f_1)$. But this contradicts the fact that (X_1, f_1) is U-minimal. As a result, we see that Y cannot get charged too many times. This bounds $|I^* \backslash I|$ and hence bounds $|\beta^*(t)|$. The final bound we achieve is $|\beta^*(t)| = \delta^{O(1)} \cdot |\beta(t)|$ for all $t \in V(T)$. As the width of (T, β) is $\delta^{O(1)} k^c$ for $c < 1$, the width of (T, β^*) is also $\delta^{O(1)} k^c$, so is the treewidth of G^*. This completes the overview of the proof of Lemma 5.

2.2 How to do subexponential branching

Next, we discuss how to achieve the subexponential bound on the number t of collections generated by our algorithm. Note that t is at most the number of leaves of the branching tree. So the key here is to have a branching tree with subexponential size.

To get some intuition, consider a stage of our branching procedure, where we are branching on a U-minimal heavy core (X, f). There is a structured sunflower $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ that witnesses (X, f), where $\Delta = \gamma^{|X|} \delta$. When we make the "yes" decision for (X, f), what we gain is that the size of $\mathcal{X}$ increases by 1. When we make the "no" decision for (X, f), we add all vertices in X to U, and by doing this we also gain something: originally the F-copies $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ can be hit by a single vertex in $X \backslash U$, but after the vertices in X are added to U, we have to use Δ vertices outside U to hit these F-copies since $V(H_1), \dots, V(H_\Delta)$ form a sunflower with core X. So ideally, this could make the size of a minimum $\mathcal{F}$-hitting set of G contained in $V(G) \backslash U$ increase by $\Delta > \delta$. (Clearly, this is not the case in general. But let us assume it is true just for explaining the intuition.) It turns out that, by a corollary of Lemma 6, the size of $\mathcal{X}$ on any successful branch path cannot exceed $\delta^{O(1)} k$. As such, the number of "yes" decisions along a successful path in the branching tree is at most $\theta_{\mathsf{yes}} = \delta^{O(1)} k$. On the other hand, the number of "no" decisions along a successful path is at most $\theta_{\mathsf{no}} = k/\delta$. Indeed, every "no" decision increases the size of a minimum $\mathcal{F}$-hitting set of G contained in $V(G) \backslash U$ by at least δ, and when we need more than k vertices in $V(G) \backslash U$ to hit all F-copies in G, we know that the current path is not successful. Thus, during the branching procedure, if we have made more than θ_{yes} "yes" decisions (resp., θ_{no} "no" decisions), we can stop branching further. In this way, the branching tree has size $\binom{\theta_{\mathsf{yes}} + \theta_{\mathsf{no}}}{\theta_{\mathsf{no}}} = (\delta k)^{O(k/\delta)}$, which is subexponential in k when setting $\delta = k^\varepsilon$ for a sufficiently small $\varepsilon > 0$. This is the intuition about where the subexponential bound comes from. Of course, the analysis does not actually work, because we cheated when bounding the number of "no" decisions. In fact, the branching tree of our current algorithm does not have a subexponential size, and we have to further elaborate it.

The above intuition has been used (implicitly) in several subexponential branching algorithms for general hitting set [27, 39], which aim to sparsify the input set system. A crucial reason why these algorithms have subexponential-size branching trees is that, during the branching procedure, they keep cleaning out "redundant" sets from the set system and only consider large sunflowers formed by the sets that survive. In our setting, an F-copy (H, π) is *U-redundant* for $U \subseteq V(G)$ if there exists another F-copy (H', π') such that $V(H') \nsubseteq U$ and $V(H') \backslash U \subsetneq V(H) \backslash U$. Note that if (H', π') is hit by a set $S \subseteq V(G) \backslash U$, then (H, π) must also be hit by S. Therefore, if U is the "undeletable" set maintained in the branching procedure, intuitively, U-redundant F-copies are useless and can be ignored (because they will be anyway hit as long as the non-U-redundant ones are hit). As aforementioned, the branching algorithms of [27, 39] keep cleaning out the redundant sets and only branch on the cores of large sunflowers formed by the sets that are non-redundant. By doing this, they can guarantee that the number of "no" decisions along any successful path is sublinear (which in turn implies the subexponential bound on the size of the branching tree). Unfortunately, this is not a good idea for our problem. Of course, in each step of our branching algorithm, we can choose to branch on only the heavy cores witnessed by a structured sunflower formed by non-U-redundant F-copies. This can still give us the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ satisfying the first condition in Theorem 3. The main issue is the treewidth bound: Lemma 5 heavily relies on the fact that $(X_1, f_1), \dots, (X_r, f_r)$ are *U-minimal* heavy cores. If we restrict ourselves to heavy cores witnessed by non-U-redundant F-copies, we cannot guarantee the heavy cores in $\mathcal{X}$ to be U-minimal during the branching procedure. Indeed, there can exist heavy cores (X, f) and (Y, g) with $(X, f) \prec (Y, g)$ such that (Y, g) can be witnessed by non-U-redundant F-copies but (X, f) cannot. Thus, at some stage of the branching, it might happen that every U-minimal heavy core cannot be witnessed by non-U-redundant F-copies while there are still (non-U-minimal) heavy cores witnessed by non-U-redundant F-copies; in this situation, we are forced to consider heavy cores that are not U-minimal (and possibly add them to $\mathcal{X}$). This entirely ruins the sublinear treewidth of the Gaifman graphs.

However, the insight that "U-redundant F-copies can be ignored" still turns out to be useful. But we have to be more careful about which heavy cores should be considered and which can be ignored. As aforementioned, only branching on the heavy cores witnessed by non-U-redundant F-copies does not work. Thus, we try to loosen the condition of "being witnessed by non-U-redundant F-copies" as follows. We say a heavy core (X, f) is *U-redundant* if *every* F-copy (H, π) with $(X, f) \preceq (V(H), \pi)$ is U-redundant. Note that for a heavy core, "being not U-redundant" is strictly weaker than "being witnessed by non-U-redundant F-copies". Furthermore, the former condition has a nice *hereditary* property (which the latter condition does not have): for heavy cores (X, f) and (Y, g) such that $(X, f) \prec (Y, g)$, if (Y, g) is not U-redundant, then (X, f) is also not U-redundant. This property is important and allows us to avoid the issue we had before (when trying to branch on heavy cores witnessed by non-U-redundant F-copies). Now we say a heavy core (X, f) is *U-active* if it is U-minimal and *not* U-redundant. We modify our algorithm so that it only branches on U-active heavy cores. We show that the collections $\mathcal{X}_1, \dots, \mathcal{X}_t$ generated still satisfy both conditions in Theorem 3 after this modification. Clearly, the treewidth bound for the Gaifman graphs of $\mathcal{X}_1, \dots, \mathcal{X}_t$ still holds, because U-active heavy cores are U-minimal. Also, we still have that if $S \subseteq V(G)$ is an $\mathcal{F}$-hitting set of G, then it hits some $\mathcal{X}_i$. The only part slightly different is to show that if S hits some $\mathcal{X}_i$, then it is an $\mathcal{F}$-hitting set of G. Consider the point we generate $\mathcal{X}_i$. At this point, $X \in \mathcal{X}$ for all U-active heavy cores (X, f), $G[U]$ is F-free, and we set $\mathcal{X}_i = \{X \backslash U : X \in \mathcal{X}\}$. Without loss of generality, we only need to consider the case $S \subseteq V(G) \backslash U$, as the sets in $\mathcal{X}_i$ are disjoint from U. Assume S hits $\mathcal{X}_i$ but S is not an $\mathcal{F}$-hitting set of G, for the sake of contradiction. Then there must exist a non-U-redundant F-copy (H, π) with $S \cap V(H) = \emptyset$. Indeed, as $S \subseteq V(G) \backslash U$, if S hits all non-U-redundant F-copies in G, then it is an $\mathcal{F}$-hitting set of G. Now $(V(H), \pi)$ is a heavy core, which is also not U-redundant. As S hits $\mathcal{X}_i$ (thus hits $\mathcal{X}$) and $\mathcal{X}$ contains all U-active heavy cores, we know that $(V(H), \pi)$ is not U-active, which further implies it is not U-minimal. Hence, there exists a U-minimal heavy core (X, f) such that $(X, f) \prec (V(H), \pi)$. By the hereditary property of U-redundancy, (X, f) is also not U-redundant and is thus U-active. It follows that $X \in \mathcal{X}$ and $X \backslash U \in \mathcal{X}_i$. However, as $S \cap V(H) = \emptyset$, we have $S \cap X = \emptyset$, which contradicts our assumption that S hits $\mathcal{X}_i$. Thus, if S hits $\mathcal{X}_i$, S must be an $\mathcal{F}$-hitting set of G.

Now we see that only branching on U-active heavy cores can still give us the collections satisfying the conditions in Theorem 3. This yields a smaller branching tree, as U-active heavy cores form a subset of U-minimal heavy cores. However, this has not yet given us a sublinear bound on the number of "no" decisions. We need the last elaboration on our branching algorithm. Still, we keep branching on U-active heavy cores (X, f). When making a "yes" decision, we recursively call Branch$(U, \mathcal{X} \cup \{X\})$ as before. The changes are made to the "no" decisions. When making a "no" decision, instead of simply adding the vertices in X to U, we further guess an additional set P of vertices that are not in the (unknown) solution S and add the vertices in P to U as well. As such, at each stage, we have one "yes" decision and *multiple* "no" decisions corresponding to different choices of P. The choices of P are as follows. Take F-copies $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ that witness (X, f), where $\Delta = \gamma^{|X|}\delta$. Suppose $C_1, \dots, C_t$ are the connected components of $F - f(X)$. Define $V_{i,j} = \pi_j^{-1}(V(C_i))$ for $(i, j) \in [t] \times [\Delta]$, which is the copy of C_i in H_j. For each $i \in [t]$, we pick at most γ sets in $\{V_{i,1}, \dots, V_{i,\Delta}\}$ and include them in P. Formally, let

$$\mathcal{P} = \left\{ \bigcup_{i=1}^{t} \bigcup_{j \in J_i} V_{i,j} : J_1, \dots, J_t \in \mathcal{J} \right\},$$

where $\mathcal{J} = \{J \subseteq [\Delta] : |J| \leq \gamma\}$. Then the choices of P are just the sets in $\mathcal{P}$. Specifically, for each $P \in \mathcal{P}$, we recursively call Branch$(U \cup X \cup P, \mathcal{X})$, which corresponds to a "no" decision. Note that $|\mathcal{P}| = \Delta^{O(\gamma)} = \delta^{O(1)}$. Thus, the degree of the branching tree becomes $\delta^{O(1)}$, but this does not influence the entire size of the branching tree too much, as $\delta = k^\varepsilon$. Surprisingly, with such a twist, we can in fact make the number of "no" decisions sublinear in k.

Finally, we briefly discuss our analysis for the number of "no" decisions. Unfortunately, as our branching algorithm is already rather different from the ones in [27, 39], their arguments is not applicable here. Instead, we use a very different analysis, which takes advantage of the graph structure, or more specifically, the bounded weak coloring number of G, as well as the assumption that the graph F is connected. Consider an $\mathcal{F}$-hitting set S of G. When branching on a (U-active) heavy core (X, f), we define *S-correct* decisions as follows. If $S \cap X \neq \emptyset$, then the "yes" decision is the only S-correct decision. If $S \cap X = \emptyset$, then a "no" decision is S-correct iff its corresponding set $P \in \mathcal{P}$ satisfies **(i)** $S \cap P = \emptyset$ and **(ii)** $S \cap P' \neq \emptyset$ for any $P' \in \mathcal{P}$ with $P \subsetneq P'$ (in other words, the set P we guess is a maximal set in $\mathcal{P}$ that is disjoint from S). We say a path in the branching tree from the root is *S-successful* if every decision on the path is S-correct. It is clear that at any node of an S-successful path, the sets U and $\mathcal{X}$ always satisfy that $S \cap U = \emptyset$ and S is a hitting set of $\mathcal{X}$.

Let S be an $\mathcal{F}$-hitting set of G with $|S| \leq k$. Our goal is to show that along any S-successful path in the branching tree, the number of "no" decisions is bounded by $O(k/\delta)$. This is done by a subtle charging argument. Fix an ordering σ of $V(G)$ such that $\mathrm{wcol}_\gamma(G, \sigma) = \mathrm{wcol}_\gamma(G)$. For $v \in V(G)$, define $\lambda_S(v) = |\{u \in S : v \in \mathrm{WR}_\gamma(G, \sigma, u)\}|$. Then we define a set

$$R = \{v \in V(G) : \lambda_S(v) \geq \delta - \gamma - \mathrm{wcol}_\gamma(G)\}.$$

We have $\sum_{v \in V(G)} \lambda_S(v) = \sum_{u \in S} |\mathrm{WR}_\gamma(G, \sigma, u)| \leq \mathrm{wcol}_\gamma(G, \sigma) \cdot |S| = \mathrm{wcol}_\gamma(G) \cdot |S|$. By an averaging argument, we deduce that $|R| \leq \frac{\mathrm{wcol}_\gamma(G)}{\delta - \gamma - \mathrm{wcol}_\gamma(G)} \cdot |S|$. Note that G is taken from a graph class of polynomial expansion and thus $\mathrm{wcol}_\gamma(G) = O(1)$. Therefore, if we choose δ much larger than $\gamma + \mathrm{wcol}_\gamma(G)$, then $|R| = O(|S|/\delta) = O(k/\delta)$. Our plan is (essentially) to charge every "no" decision on the S-successful path to a vertex in R, with the guarantee that each vertex in R only gets charged $O(1)$ times. If this can be done, then the number of "no" decisions is $O(k/\delta)$.

Observation 7. *Let (X, f) be a heavy core in G, and $x \in X$ be the largest vertex under σ. If $S \cap X = \emptyset$, then $x \in R$. Furthermore, if $\delta > \mathrm{wcol}_\gamma(G)$, then $x \in \mathrm{WR}_\gamma(G, \sigma, u)$ for all $u \in X$.*

Proof sketch. There exist F-copies $(H_1, \pi_1), \dots, (H_\Delta, \pi_\Delta)$ such that $V(H_1), \dots, V(H_\Delta)$ form a sunflower with core X, where $\Delta = \gamma^{|X|}\delta$. Since $S \cap X = \emptyset$ and S is an $\mathcal{F}$-hitting set, there exists $u_i \in S \cap (V(H_i) \backslash X)$ for all $i \in [\Delta]$. Note that $u_1, \dots, u_\Delta$ are distinct, as $V(H_1) \backslash X, \dots, V(H_\Delta) \backslash X$ are disjoint. Using the assumptions that

F is connected and x is the largest vertex in X, we can deduce that $x \in \mathrm{WR}_\gamma(G, \sigma, u_i)$ for at least $\Delta - \mathrm{wcol}_\gamma(G)$ indices $i \in [\Delta]$. As $u_1, \dots, u_\Delta \in S$, this implies $\lambda_S(x) \geq \Delta - \mathrm{wcol}_\gamma(G) \geq \delta - \gamma - \mathrm{wcol}_\gamma(G)$. Thus, $x \in R$. The second statement also follows easily from the facts that F is connected and x is the largest vertex in X. $\square$

Consider a "no" decision on the S-successful path, and let (X, f) be the heavy core on which the decision was made. At the time we made the decision for (X, f), we have $X \not\subseteq U$. Pick an arbitrary vertex $y \in X \backslash U$. As we made a "no" decision for (X, f), we have $S \cap X = \emptyset$. The above observation then implies $X \cap R \cap \mathrm{WR}_\gamma(G, \sigma, y) \neq \emptyset$, since it contains the largest vertex in X. We then charge the "no" decision for (X, f) to the *smallest* vertex in $X \cap R \cap \mathrm{WR}_\gamma(G, \sigma, y)$.

It is non-obvious that why each vertex in R only gets charged $O(1)$ times. The intuition is roughly as follows. Assume there are too many "no" decisions charged to the same vertex in R. Let $(X_1, f_1), \dots, (X_r, f_r)$ be the heavy cores these "no" decisions are made on, where r is very large. By the sunflower lemma, we may assume that $X_1, \dots, X_r$ form a sunflower, without loss of generality. Suppose the decisions for $(X_1, f_1), \dots, (X_r, f_r)$ are made in order. After we made the "no" decisions for $(X_1, f_1), \dots, (X_{r-1}, f_{r-1})$, the vertices in $X_1, \dots, X_{r-1}$ are all added to U. In addition, each "no" decision here also adds a set P of vertices to U, where P is maximal among all choices that are disjoint from S (because the "no" decision is S-correct). We somehow show that using these vertices added to U, for every F-copy (H, π) with $(X_r, f_r) \preceq (V(H), \pi)$, we can construct another F-copy (H', π') satisfying $V(H') \backslash U \subsetneq V(H) \backslash U$; the construction of H' is done by carefully replacing a part of H that is not totally contained in U with an isomorphic one that consists of those vertices added to U (of course this construction relies on our charging rule as well). It then follows that when we branch on (X_r, f_r), every F-copy (H, π) with $(X_r, f_r) \preceq (V(H), \pi)$ is U-redundant and hence (X_r, f_r) is U-redundant. But this contradicts the fact that we only branch on U-active heavy cores, and thus each vertex in R cannot get charged too many times. As a result, the number of "no" decisions on any S-successful path is bounded by $O(k/\delta)$.

According to the above discussion, we can set $\theta_{\mathrm{no}} = O(k/\delta)$ as the budget for "no" decisions, which makes the branching tree have subexponential size. This completes the overview of how we achieve the subexponential bounds in Theorem 3.

3 CONCLUSION

In this paper, we studied the (Weighted) $\mathcal{F}$-Hitting problem, in which we are given a (vertex-weighted) graph G together with a parameter k, and the goal is to compute a set $S \subseteq V(G)$ of vertices (with minimum total weight) such that $|S| \leq k$ and $G - S$ does not contain any graph in $\mathcal{F}$ as a subgraph. We gave a general framework for designing subexponential FPT algorithms for Weighted $\mathcal{F}$-Hitting on a large family of graph classes that admit "small" separators relative to the graph size and the maximum clique size. Such graph classes include all classes of polynomial expansion and many important classes of geometric intersection graphs, on which subexponential algorithms are widely studied. The algorithms obtained from our framework runs in $2^{O(k^c)} \cdot n + O(m)$ time, where $n = |V(G)|$ and $m = |E(G)|$. The technical core of our framework is a subexponential branching algorithm that reduces an instance of $\mathcal{F}$-Hitting (on the aforementioned graph classes) to $2^{O(k^c)}$ instances of the general hitting-set problem, where the Gaifman graph of each instance has treewidth $O(k^c)$, for some constant $c < 1$.

We now propose several open problems for future research. First, as mentioned in the introduction, one can study the "induced" variant of the $\mathcal{F}$-Hitting problem, called Induced $\mathcal{F}$-Hitting, which aims to delete k vertices from an input graph G such that the resulting graph does not contain any $F \in \mathcal{F}$ as an *induced* subgraph. It is interesting to ask whether Induced $\mathcal{F}$-Hitting can be solved in $2^{O(k^c)} \cdot n^{f(\mathcal{F})}$ time for $c < 1$ and some function f on the graph classes considered in this paper. For more restrictive graph classes (such as graph classes of polynomial expansion), one can even seek for algorithms with running time $2^{O(k^c)} \cdot n$ for $c < 1$. Second, the constant c in the running time $2^{O(k^c)} \cdot n + O(m)$ of our algorithms is only slightly smaller than 1. How to further improve the running time would be a natural problem. Finally, one can investigate the problem with parameters other than the solution size k, such as the treewidth t (or other width parameters) of the input graph G. Although Cygan et al. [8] showed that solving $\mathcal{F}$-Hitting requires $2^{t^{f(\mathcal{F})}} \cdot n$ time for some function f, this hardness result was proved on general graphs. An interesting open question here is whether $\mathcal{F}$-Hitting on the graph classes considered in this paper can be solved in $2^{O(t^c)} \cdot n$ time for a constant c *independent* of $\mathcal{F}$, or even subexponential time in t.

ACKNOWLEDGMENTS

The research of Saket Saurabh was partially supported by the European Research Council (ERC) grant no. 819416, and Swarnajayanti Fellowship no. DST/SJF/MSA01/2017-18. The research of Meirav Zehavi was partially supported by the European Research Council (ERC) grant no. 101039913. The authors would like to thank the anonymous reviewers for their insightful comments on this paper.

REFERENCES

[1] Gaétan Berthe, Marin Bougeret, Daniel Gonçalves, and Jean-Florent Raymond. 2024. Kick the Cliques. In *19th International Symposium on Parameterized and Exact Computation (IPEC 2024)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 13–1.
[2] Nadja Betzler, Robert Bredereck, Rolf Niedermeier, and Johannes Uhlmann. 2012. On bounded-degree vertex deletion parameterized by treewidth. *Discrete Applied Mathematics* 160, 1-2 (2012), 53–60.
[3] Marin Bougeret, Bart MP Jansen, and Ignasi Sau. 2024. Kernelization Dichotomies for Hitting Subgraphs under Structural Parameterizations. In *51st International Colloquium on Automata, Languages and Programming*.
[4] Marin Bougeret and Ignasi Sau. 2019. How much does a treedepth modulator help to obtain polynomial kernels beyond sparse graphs? *Algorithmica* 81, 10 (2019), 4043–4068.
[5] Boštjan Brešar, Marko Jakovac, Ján Katrenič, Gabriel Semanišin, and Andrej Taranenko. 2013. On the vertex k-path cover. *Discrete Applied Mathematics* 161, 13-14 (2013), 1943–1949.
[6] Leizhen Cai. 1996. Fixed-Parameter Tractability of Graph Modification Problems for Hereditary Properties. *Inf. Process. Lett.* 58, 4 (1996), 171–176.
[7] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms.* Springer.
[8] Marek Cygan, Dániel Marx, Marcin Pilipczuk, and Michał Pilipczuk. 2017. Hitting forbidden subgraphs in graphs of bounded treewidth. *Information and Computation* 256 (2017), 62–82.
[9] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. 2005. Fixed-parameter algorithms for (k, r)-center in planar graphs and map graphs. *ACM Trans. Algorithms* 1, 1 (2005), 33–47.

[10] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. 2005. Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs. *Journal of the ACM (JACM)* 52, 6 (2005), 866–893.

[11] Pål Grønås Drange, Markus Dregi, and Pim van't Hof. 2016. On the computational complexity of vertex integrity and component order connectivity. *Algorithmica* 76, 4 (2016), 1181–1202.

[12] Zdeněk Dvořák, Daniel Král, and Robin Thomas. 2013. Testing first-order properties for subclasses of sparse graphs. *Journal of the ACM (JACM)* 60, 5 (2013), 1–24.

[13] Zdeněk Dvořák, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. 2023. Efficient Approximation for Subgraph-Hitting Problems in Sparse Graphs and Geometric Intersection Graphs. *arXiv preprint arXiv:2304.13695* (2023).

[14] Zdeněk Dvořák and Sergey Norin. 2016. Strongly Sublinear Separators and Polynomial Expansion. *SIAM Journal on Discrete Mathematics* 30, 2 (2016), 1095–1101.

[15] Eduard Eiben, Diptapriyo Majumdar, and MS Ramanujan. 2022. On the Lossy Kernelization for Connected Treedepth Deletion Set. In *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer, 201–214.

[16] Samuel Fiorini, R Krithika, NS Narayanaswamy, and Venkatesh Raman. 2018. Approximability of clique transversal in perfect graphs. *Algorithmica* 80, 8 (2018), 2221–2239.

[17] Fedor V Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2019. Decomposition of Map Graphs with Applications. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.

[18] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2019. Finding, Hitting and Packing Cycles in Subexponential Time on Unit Disk Graphs. *Discrete & Computational Geometry* 62, 4 (2019), 879–911.

[19] Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. 2018. Excluded Grid Minors and Efficient Polynomial-Time Approximation Schemes. *Journal of the ACM (JACM)* 65, 2 (2018), 10:1–10:44.

[20] Jakub Gajarský, Petr Hliněný, Jan Obdržálek, Sebastian Ordyniak, Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, and Somnath Sikdar. 2017. Kernelization using structural parameters on sparse graph classes. *J. Comput. System Sci.* 84 (2017), 219–242.

[21] Robert Ganian, Fabian Klute, and Sebastian Ordyniak. 2021. On structural parameterizations of the bounded-degree vertex deletion problem. *Algorithmica* 83, 1 (2021), 297–336.

[22] Lito Julius Goldmann. 2021. *Parameterized Complexity of Modifying Graphs to be Biclique-free.* Ph. D. Dissertation. Institute of Software.

[23] Marina Groshaus, Pavol Hell, Sulamita Klein, Loana Tito Nogueira, and Fábio Protti. 2009. Cycle transversals in bounded degree graphs. *Electronic Notes in Discrete Mathematics* 35 (2009), 189–195.

[24] Daniel Gross, Monika Heinig, Lakshmi Iswara, L William Kazmierczak, Kristi Luttrell, John T Saccoman, and Charles Suffel. 2013. A survey of component order connectivity models of graph theoretic networks. *WSEAS Transactions on Mathematics* 12 (2013), 895–910.

[25] Anupam Gupta, Euiwoong Lee, Jason Li, Pasin Manurangsi, and Michał Włodarczyk. 2019. Losing treewidth by separating subsets. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 1731–1749.

[26] Sariel Har-Peled and Kent Quanrud. 2017. Approximation algorithms for polynomial-expansion and low-density graphs. *SIAM J. Comput.* 46, 6 (2017), 1712–1744.

[27] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. 2001. Which problems have strongly exponential complexity? *J. Comput. System Sci.* 63, 4 (2001), 512–530.

[28] Mithilesh Kumar and Daniel Lokshtanov. 2017. A $2lk$ Kernel for l-Component Order Connectivity. In *11th International Symposium on Parameterized and Exact Computation (IPEC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[29] Hung Le and Cuong Than. 2022. Greedy Spanners in Euclidean Spaces Admit Sublinear Separators. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 3287–3310.

[30] Euiwoong Lee. 2017. Partitioning a graph into small pieces with applications to path transversal. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 1546–1558.

[31] John M Lewis and Mihalis Yannakakis. 1980. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. System Sci.* 20, 2 (1980), 219–230.

[32] Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. 2022. Subexponential Parameterized Algorithms on Disk Graphs (Extended Abstract). In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2005–2031.

[33] Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. 2023. A framework for approximation schemes on disk graphs. In *34rd Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM.

[34] Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, Jie Xue, and Meirav Zehavi. 2024. Subexponential Parameterized Algorithms for Hitting Subgraphs. *arXiv preprint arXiv:2409.04786* (2024).

[35] Gary L Miller, Shang-Hua Teng, William Thurston, and Stephen A Vavasis. 1997. Separators for sphere-packings and nearest neighbor graphs. *Journal of the ACM (JACM)* 44, 1 (1997), 1–29.

[36] Jaroslav Nešetřil and Patrice Ossona De Mendez. 2008. Grad and classes with bounded expansion I. Decompositions. *European Journal of Combinatorics* 29, 3 (2008), 760–776.

[37] Jaroslav Nešetřil and Patrice Ossona De Mendez. 2008. Grad and classes with bounded expansion II. Algorithmic aspects. *European Journal of Combinatorics* 29, 3 (2008), 777–791.

[38] Michał Pilipczuk. 2011. Problems parameterized by treewidth tractable in single exponential time: A logical approach. In *International Symposium on Mathematical Foundations of Computer Science*. Springer, 520–531.

[39] Rahul Santhanam and Srikanth Srinivasan. 2012. On the limits of sparsification. In *Automata, Languages, and Programming: 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I 39*. Springer, 774–785.

[40] David P Williamson and David B Shmoys. 2011. *The design of approximation algorithms*. Cambridge university press.

[41] Xuding Zhu. 2009. Colouring graphs with bounded generalized colouring number. *Discrete Mathematics* 309, 18 (2009), 5562–5568.

Received 2024-11-04; accepted 2025-02-01