# Backdoor Sets on Nowhere Dense SAT

**Daniel Lokshtanov**
University of California, Santa Barbara, CA, USA

**Fahad Panolan**
Indian Institute of Technology Hyderabad, India

**M. S. Ramanujan**
University of Warwick, Coventry, UK

## Abstract

For a satisfiable CNF formula $\phi$ and an integer $t$, a *weak backdoor set* to treewidth-$t$ is a set of variables such that there is an assignment to this set that reduces $\phi$ to a *satisfiable* formula that has an incidence graph of treewidth at most $t$. A natural research program in the work on fixed-parameter algorithms (FPT algorithms) for SAT is to delineate the tractability borders for the problem of detecting a small weak backdoor set to treewidth-$t$ formulas. In this line of research, Gaspers and Szeider (ICALP 2012) showed that detecting a weak backdoor set of size at most $k$ to treewidth-1 is W[2]-hard parameterized by $k$ if the input is an arbitrary CNF formula. Fomin, Lokshtanov, Misra, Ramanujan and Saurabh (SODA 2015), showed that if the input is $d$-CNF, then detecting a weak backdoor set of size at most $k$ to treewidth-$t$ is fixed-parameter tractable (parameterized by $k, t, d$). These two results indicate that sparsity of the input plays a role in determining the parameterized complexity of detecting weak backdoor sets to treewidth-$t$.

In this work, we take a major step towards characterizing the precise impact of sparsity on the parameterized complexity of this problem by obtaining algorithmic results for detecting small weak backdoor sets to treewidth-$t$ for input formulas whose incidence graphs belong to a *nowhere-dense graph class*. Nowhere density provides a robust and well-understood notion of sparsity that is at the heart of several advances on model checking and structural graph theory. Moreover, nowhere-dense graph classes contain many well-studied graph classes such as *bounded treewidth graphs, graphs that exclude a fixed (topological) minor and graphs of bounded expansion*.

Our main contribution is an algorithm that, given a formula $\phi$ whose incidence graph belongs to a fixed nowhere-dense graph class and an integer $k$, in time $f(t,k)|\phi|^{\mathcal{O}(1)}$, either finds a satisfying assignment of $\phi$, or concludes correctly that $\phi$ has no weak backdoor set of size at most $k$ to treewidth-$t$.

To obtain this algorithm, we develop a strategy that only relies on the fact that nowhere-dense graph classes are biclique-free. That is, for every nowhere-dense graph class, there is a $p$ such that it is contained in the class of graphs that exclude $K_{p,p}$ as a subgraph. This is a significant feature of our techniques since the class of biclique-free graphs also generalizes the class of graphs of bounded degeneracy, which are *incomparable* with nowhere-dense graph classes. As a result, our algorithm also generalizes the results of Fomin, Lokshtanov, Misra, Ramanujan and Saurabh (SODA 2015) for the special case of $d$-CNF formulas as input when $d$ is fixed. This is because the incidence graphs of such formulas exclude $K_{d+1,d+1}$ as a subgraph.

**2012 ACM Subject Classification** Theory of computation → Parameterized complexity and exact algorithms

**Keywords and phrases** Fixed-parameter Tractability, Satisfiability, Backdoors, Treewidth

**Digital Object Identifier** 10.4230/LIPIcs.ICALP.2022.91

**Category** Track A: Algorithms, Complexity and Games

**Funding** *Daniel Lokshtanov*: Supported by United States-Israel Binational Science Foundation (BSF) grant no. 2018302 and National Science Foundation (NSF) award CCF-2008838.
*Fahad Panolan*: Supported by Seed grant, IIT Hyderabad (SG/IITH/F224/2020-21/SG-79).
*M. S. Ramanujan*: Supported by Engineering and Physical Sciences Research Council (EPSRC) grants EP/V007793/1 and EP/V044621/1.



49th International Colloquium on Automata, Languages, and Programming (ICALP 2022).
Editors: Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff;
Article No. 91; pp. 91:1–91:20

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## 1 Introduction

A central concept in the design of fixed-parameter algorithms for SAT is the notion of *backdoor sets* introduced by Williams et al. [41], which was also introduced under a different name by Crama et al. [9]. These are sets of variables in the input formula whose instantiation can lead to a dramatic simplification of the formula, moving the formula into a class where SAT is easy to solve, such as *Horn formulas*. Backdoor sets provide a natural way to express the deviation of a SAT instance from well-studied "islands of tractability" (i.e., tractable cases of SAT) and allow us to study the question -"could we easily solve SAT instances that do not necessarily fall neatly within a known tractable fragment, yet are 'close' to them?"

To formalize this idea, fix a class of CNF formulas, call it $\mathcal{C}$, which is also called the *base* class. Let $\phi$ be the input instance of SAT. If $\phi$ is satisfiable, then a nonempty subset $B$ of the variables in $\phi$ is a *weak backdoor* of $\phi$ to the class $\mathcal{C}$ (or a weak $\mathcal{C}$-backdoor set) if there exists an assignment $\tau$ to the variables in $B$ such that the reduced instance $\phi[\tau]$ is a satisfiable formula in the class $\mathcal{C}$. We say that $B$ is a *strong backdoor* of $\phi$ to the class $\mathcal{C}$ (or strong $\mathcal{C}$-backdoor set) if, for every assignment $\tau$ to the variables in $B$, the reduced instance $\phi[\tau]$ is in $\mathcal{C}$. Clearly, if $\mathcal{C}$ is a class of formulas on which SAT can be efficiently solved and one knows a weak/strong $\mathcal{C}$-backdoor set of the input instance, then a straightforward way of checking satisfiability and computing a satisfying assignment of the input instance involves exploring all instantiations of the variables in the backdoor set and efficiently solving the simplified instances.

Over the last two decades, there has been an active theoretical research program on SAT – the design of algorithms for SAT by exploring the *parameterized complexity* of computing and utilizing small backdoor sets to an efficiently-solvable class of CNF formulas. The study of the parameterized complexity of the problem of finding small backdoor sets was formally initiated by Nishimura et al. [33] and over the past decade and a half, there has been an extensive study of backdoors for SAT in the realm of parameterized complexity (see [37, 39, 35, 26, 13, 21] for a partial list). We refer to the survey of Gaspers and Szeider [25] for an overview of many of these works. In recent years, the success brought to the parameterized complexity of SAT by the study of backdoor detection has been replicated for CSP [2, 4, 20, 17, 18] and Mixed-ILP [16]. We refer to the survey [22] for a more comprehensive picture of the state of the art on backdoors to CSPs.

A basic part of this research program exploring the parameterized complexity of SAT is to answer the question:

**For which input formulas is the detection of small backdoors to the class of bounded-treewidth formulas fixed-parameter tractable?**

Bounded-treewidth formulas are CNF formulas whose incidence graphs have bounded treewidth and it is well-known that SAT is efficiently solvable on these [10, 15, 38]. The study of computing small backdoors to bounded-treewidth formulas from the parameterized complexity perspective was initiated by Gaspers and Szeider in [23] and further studied in [24, 26]. Gaspers and Szeider [23] gave an FPT-approximation algorithm for the problem of computing a strong backdoor set to Acyclic formulas, which was later extended to bounded-treewidth formulas [26]. Specifically, they gave an algorithm that runs in time $f(k)|\phi|^{\mathcal{O}(1)}$, for some computable function $f$, and either detects a strong backdoor in $\phi$ of size at most $2^k$ to bounded-treewidth formulas or reports that there is no such strong backdoor of size at most $k$. Since approximate backdoor sets are also sufficient to solve SAT, they were able to conclude that SAT parameterized by the size of a smallest strong backdoor set to

bounded-treewidth formulas is fixed-parameter tractable *with no restrictions on the input*. On the other hand, the picture for weak backdoor sets is much less comprehensive and is the main motivation behind our work.

### State of the art for detecting weak backdoors

In [23], Gaspers and Szeider studied the problem of detecting a weak backdoor of size at most $k$ to Acyclic formulas, i.e, formulas with incidence graphs of treewidth at most 1. They showed that this problem is W[2]-hard in general but fixed-parameter tractable (FPT) on $d$-CNF formulas for every fixed $d$. This implies that $d$-SAT parameterized by the size of a smallest weak backdoor set to Acyclic formulas is fixed-parameter tractable. This result was subsequently improved by Fomin et al. [13], who showed that $d$-SAT parameterized by the size of a smallest weak backdoor set to bounded-treewidth formulas is fixed-parameter tractable. The lack of progress in this direction since then, means that the state of the art on detecting weak backdoors to bounded-treewidth formulas continues to experience a huge gap between instances for which we know parameterized hardness results (i.e., general CNF formulas) and instances for which we have FPT algorithms (i.e., $d$-CNF). Where exactly does the tractability border lie? This brings us to the overarching goal of identifying maximal classes of CNF formulas for which SAT is FPT when parameterized by the size of a smallest weak backdoor set to the class of bounded-treewidth formulas.

### Our work

Since the incidence graphs of $d$-CNF formulas only have linearly-many edges in the number of clauses, a natural step towards the overarching goal outlined above is to first understand the effect that the "sparsity" of the input incidence graph has on the tractability of the problem. In particular, could one show that as long as the input has bounded degeneracy or is nowhere dense [32], then SAT is FPT parameterized by the size of a smallest weak backdoor set to bounded-treewidth formulas? Nowhere density yields a robust notion of a maximal sparse graph class and nowhere-dense graph classes have been at the heart of major developments in graph theory and model checking [31, 27, 29] in the last decade. Therefore, one could reasonably hope to harness the techniques developed in these efforts, towards our purpose. Starting from this point, in this paper, we obtain tractability results that take us simultaneously beyond the class of nowhere-dense graphs and graphs of bounded degeneracy. Towards this, we initiate the study of weak backdoor detection on *biclique-free formulas*. More precisely, we consider $K_{d,d}$-free formulas, which are CNF formulas where no set of $d$ clauses contain literals from $d$ variables in their common intersection. In graph-theoretic terms, the incidence graphs of $K_{d,d}$-free formulas exclude the complete bipartite graph $K_{d,d}$ as a subgraph. $K_{d,d}$-free graphs are very wide class of sparse graphs and include many well-studied graph classes such as *bounded treewidth graphs, graphs that exclude a fixed minor, graphs of bounded expansion, nowhere-dense graphs and graphs of bounded degeneracy*. That is, for any of the classes – bounded treewidth graphs, graphs that exclude a fixed minor, graphs of bounded expansion, nowhere-dense graphs and graphs of bounded degeneracy, there is a $p$ such that the class is contained in the class of $K_{p,p}$-free graphs (see Figure 1 for an illustration of the inclusion relation between these classes).

## 1.1 Our results

In order to present our results in the most general terms possible, we consider weak backdoor sets not just to bounded treewidth formulas, but what we call $\mathcal{F}$-minor-free formulas. Let $\mathcal{F}$ be a finite set of graphs. For a CNF formula $\phi$, a set $S$ of variables is called *a weak*

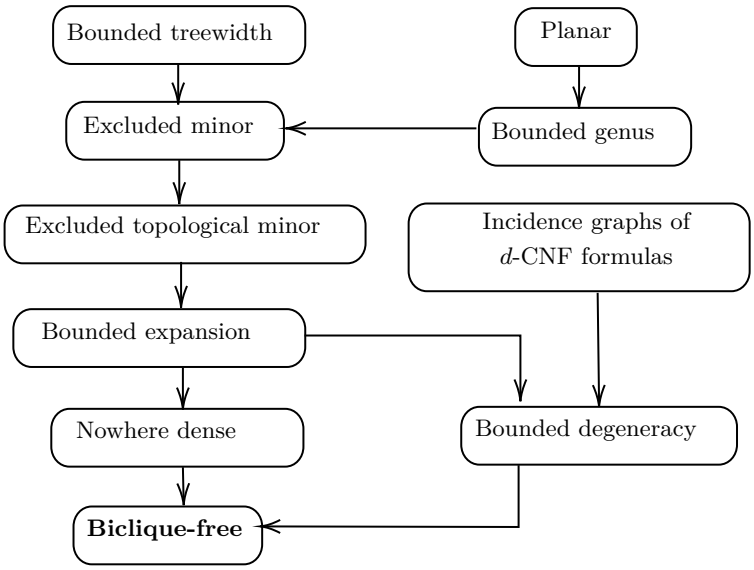


**Figure 1** Inclusion relation between the class of biclique-free graphs considered in this paper and well-studied graph classes in the literature (figure based on [40]). If there in arrow of the form $A \to B$, then class $A$ is a subclass of $B$.

*$\mathcal{F}$-minor-free backdoor set* of $\phi$ if it is a weak backdoor set of $\phi$ into the class of CNF formulas whose incidence graphs exclude every graph in $\mathcal{F}$ as a minor. Notice that a weak backdoor set of a formula into treewidth-$t$ formulas is also a weak $\mathcal{F}$-minor-free backdoor set for some $\mathcal{F}$ – take $\mathcal{F}$ to be precisely the set of forbidden minors for graphs of treewidth at most $t$. Note that the size of $\mathcal{F}$ depends only on $t$. The reason for studying the problem in this more general setting is that it allows us to also obtain an additional result from the proof techniques used for our main result. Moreover, due to the extensive work in parameterized complexity on connections between vertex-deletion problems to bounded treewidth graphs and vertex-deletion problems to minor-free graphs, there is a rich collection of powerful algorithmic techniques that we are able to draw from (see, for example, [14, 19, 13]).

We now state our central result formally and discuss our corollaries and further context.

▶ **Theorem 1.** *For every $d \in \mathbb{N}$, there is a computable function $f_d$ and an algorithm $\mathcal{A}_d$ that satisfies the following:*

1. *$\mathcal{A}_d$ takes as input a $K_{d,d}$-free formula $\phi$, a family $\mathcal{F}$ of graphs containing at least one planar graph, and an integer $k$.*
2. *$\mathcal{A}_d$ runs in time $f_d(\mathcal{F}, k) \cdot |\phi|^{\mathcal{O}(d)}$.*
3. *$\mathcal{A}_d$ either outputs a weak $\mathcal{F}$-minor-free-backdoor set of $\phi$ of size at most $k$, or concludes correctly that $\phi$ has no weak $\mathcal{F}$-minor-free-backdoor set of size at most $k$.*

Let us first make a few remarks regarding the time-complexity of the algorithm described in the above statement. First of all, in the running time of the above algorithm, the exponent in the term $|\phi|^{\mathcal{O}(d)}$ is independent of $k$ and $\mathcal{F}$. Secondly, as our main goal is to obtain a fixed-parameter tractability classification result, and as we have proved our result in such general terms, we have not attempted to optimize the function $f_d$. Indeed, due to the standard step of invoking Courcelle's theorem as a subroutine when the input treewidth is already bounded (see, for example, [23, 26]) one should not expect this algorithm to be practical *as presented*. However, our result is an important proof of concept demonstrating

fixed-parameter tractability for such a general class of inputs. We believe that optimizing the function $f_d$ is a more meaningful research question for specific subclasses of $K_{d,d}$-free formulas, e.g., formulas with incidence graphs that exclude a fixed graph as a minor.

We now describe the main consequences of our result. As discussed, it is well-known that satisfiability of a $t$-tw formula (i.e., a CNF formula whose incidence graph has treewidth at most $t$) can be tested in linear time for every fixed $t$ and moreover, a satisfying assignment can be computed in the same time. Therefore, we can combine Theorem 1 with the fact that graphs of treewidth at most $t$ have a finite and computable set of forbidden minors that include the $(t+1) \times (t+1)$-grid, which is a planar graph. This leads us to the first FPT algorithm for SAT on $K_{d,d}$-free formulas (for fixed $d$) parameterized by $t + \mathsf{wb}_t(\phi)$. Here, $\mathsf{wb}_t(\phi)$ denotes the size of a smallest weak $t$-tw-backdoor set of $\phi$. The formal statement follows.

▶ **Theorem 2.** *For every $d \in \mathbb{N}$, there is a computable function $f_d$ and an algorithm that, given $t \in \mathbb{N}$, a $K_{d,d}$-free formula $\phi$ and an integer $k$, runs in time $f_d(t,k)|\phi|^{\mathcal{O}(d)}$ and either finds a satisfying assignment of $\phi$, or concludes correctly that $\phi$ has no weak $t$-tw-backdoor set of size at most $k$.*

An immediate corollary of the above theorem is that $d$-SAT is FPT parameterized by $t + \mathsf{wb}_t(\phi)$ for every fixed $d$. To obtain this, notice that a $d$-CNF formula is $K_{d+1,d+1}$-free.

A further consequence of the techniques used to prove Theorem 1 is in the case of weak backdoor sets to *nested formulas*. These form a subclass of $K_{3,3}$-free formulas and are known to have treewidth at most 3 (see preliminaries for the formal definition of nested formulas). However, this is not a precise characterization of these formulas, so we need a minor modification in the proof of Theorem 1 to obtain the following result.

▶ **Theorem 3.** *For every $d \in \mathbb{N}$, there is a computable function $f_d$ and an algorithm that, given $t \in \mathbb{N}$, a $K_{d,d}$-free formula $\phi$ and an integer $k$, runs in time $f_d(t,k)|\phi|^{\mathcal{O}(d)}$ and either finds a satisfying assignment of $\phi$, or concludes correctly that $\phi$ has no weak backdoor set of size at most $k$ to the class of nested formulas.*

### Hardness results for strong and weak backdoor detection

To complement our algorithmic results, we also show new hardness results that strengthen existing results. Towards this, we present a polynomial-time algorithm that, given a Set Cover instance $I = (U, \mathcal{S})$, numbers $d, d' \geq 2$ and a non-negative number $k$, outputs a CNF formula $\phi$ such that if $I$ has a set cover of size $k$, then $\phi$ has a strong/weak backdoor set of size $k$ to the class of empty formulas, which are formulas with no clauses. Conversely, if $\phi$ has a strong/weak backdoor set of size $k$ to any subclass of $\mathcal{K}_{d,d'}$-free formulas, then $I$ has a set cover of size $k$. This algorithm is effectively a reduction that implies that for any class $\mathcal{C}$ that is a superclass of empty formulas and a subclass of $\mathcal{K}_{d,d}$-free formulas for some $d \in \mathbb{N}$, the problem of detecting a strong/weak backdoor to $\mathcal{C}$ of size at most $k$ is at least as hard as solving Set Cover parameterized by the solution size (i.e., it is W[2]-hard).

In particular, notice that for every $t \geq 0$, there exists a $d \geq 2$ such that $t$-tw-CNF is a subclass of $K_{d,d}$-free formulas. Moreover, nested formulas are subclasses of $K_{3,3}$-free formulas. Hence, we obtain the following consequences.

▶ **Theorem 4.** *For every $t \geq 0$, family $\mathcal{F}$ that contains a planar graph but does not contain an independent set, and for each class $\mathcal{C} \in \{t$-tw formulas, nested formulas, $\mathcal{F}$-minor-free formulas$\}$, the problem of detecting a strong/weak backdoor set to $\mathcal{C}$ of size at most $k$ is $W[2]$-hard parameterized by $k$.*

**Table 1** Our results for weak backdoor detection in relation to the state of the art. Each row corresponds to a base class into which we are finding a backdoor set. Each column corresponds to a possible class of inputs.

| | Input: $d$-CNF | Input: $K_{d,d}$-free CNF | Input: CNF |
|---|---|---|---|
| Acyclic | FPT [23] | **FPT (Theorem 2)** | W[2]-hard [23] |
| Nested | **FPT** | **FPT (Theorem 3)** | W[2]-hard (**Theorem 4**) |
| tw-$t$ (for every $t$) | FPT [13] | **FPT (Theorem 2)** | W[2]-hard (**Theorem 4**) |
| Planar-$\mathcal{F}$ minor-free | **FPT** | **FPT** (Theorem 1) | W[2]-hard (**Theorem 4**) |

This theorem strengthens the hardness result of Gaspsers and Szeider [23] and shows that not only is it W[2]-hard to detect small weak backdoor sets (with no restriction on the input) to Acyclic formulas, but also to treewidth-$t$ formulas for every fixed $t$.

## 2 Preliminaries

### CNF Formulas

We consider propositional formulas in conjunctive normal form (CNF). In our algorithms to compute weak backdoor sets to bounded-treewidth formulas, we do not assume that the clauses exclude a pair of complementary literals. However, when the goal is to solve SAT, it can be assumed without loss of generality that the input formula does not contain a clause with a pair of complementary literals. For a CNF formula $\phi$, we use $\mathsf{var}(\phi)$ and $\mathsf{cla}(\phi)$ to refer to the sets of variables and clauses in $\phi$, respectively. We say that a variable $x$ is positive (negative) in a clause $C$ if $x \in C$ ($\overline{x} \in C$), and we write $\mathsf{var}(C)$ for the set of variables that are positive or negative in $C$, while we use $\mathsf{lit}(C)$ to denote the set of literals in $C$. For a set $\mathcal{C}$ of clauses, we use $\mathsf{var}(\mathcal{C})$ to denote $\bigcup_{C\in\mathcal{C}} \mathsf{var}(C)$ and call this set *the set of variables of* $\mathcal{C}$. Given a CNF formula $\phi$ and a truth assignment $\tau$, $\phi[\tau]$ denotes the *truth assignment reduct* of $\phi$ under $\tau$, which is the CNF formula obtained from $\phi$ by first removing all clauses that are satisfied by $\tau$ and second removing from the remaining clauses all literals $x, \overline{x}$ with $x \in \mathsf{var}(\tau)$. The assigned variables are also removed.

### Incidence graphs

The *incidence graph* of a CNF formula $\phi$, $\mathsf{inc}(\phi)$, is the bipartite graph whose vertices are the variables and clauses of $\phi$, and where vertices corresponding to a variable $x$ and a clause $C$ are adjacent if and only if $x \in \mathsf{var}(C)$. Further, an edge between a vertex corresponding to $x \in \mathsf{var}(\phi)$ and $C \in \mathsf{cla}(\phi)$ has the *polarity* (or *label*) $+$ if $x \in \mathsf{lit}(C)$ and $\overline{x} \notin \mathsf{lit}(C)$ and is labeled $-$ if $\overline{x} \in \mathsf{lit}(C)$ and $x \notin \mathsf{lit}(C)$. If $x, \overline{x} \in \mathsf{lit}(C)$, then this edge is labeled $\mathfrak{b}$. For an incidence graph $G$, we abuse notation and use $\mathsf{var}(G)$ to refer to the vertices of $G$ that correspond to variables in $\psi(G)$, and $\mathsf{cla}(G)$ to refer to the vertices of $G$ that correspond to clauses in $\psi(G)$. Also, for a vertex subset $Y \subseteq V(G)$, we continue to use the notations $\mathsf{var}(Y)$ and $\mathsf{cla}(Y)$ to refer to the sets $\mathsf{var}(G) \cap Y$ and $\mathsf{cla}(G) \cap Y$, respectively. We say that a graph is $K_{d,d}$*-free* if it does not contain $K_{d,d}$ as a subgraph. Notice that the $K_{d,d}$-free incidence graphs correspond to CNF formulas where no set of $d$ clauses contain literals from $d$ distinct variables in their common intersection.

### Nested formulas

A CNF formula $\phi$ is *nested* [28] precisely if the graph $\mathsf{inc}(\mathsf{univ}(\phi))$ is planar, where $\mathsf{univ}(\phi)$ is obtained from $\phi$ by adding any universal clause $c^\star$ containing a literal from each variable of $\phi$. Moreover, $\mathsf{inc}(\phi)$ has treewidth at most 3 if $\phi$ is nested. Hence, one can determine satisfiability of nested formulas in polynomial time. Nested formulas are a subclass of planar CNF formulas and so, are also a subclass of $K_{3,3}$-free formulas.

### Treewidth

Let $G$ be a graph. A *tree decomposition* of $G$ is a pair $(T, \mathcal{X} = \{X_t\}_{t \in V(T)})$ where $T$ is a tree and $\mathcal{X}$ is a collection of subsets of $V(G)$ such that: (i) $\forall e = (u, v) \in E(G)$, there is $t \in V(T) : \{u, v\} \subseteq X_t$, and (ii) $\forall v \in V(G)$, $T[\{t \mid v \in X_t\}]$ is a non-empty connected subtree of $T$. We call the vertices of $T$ *nodes* and the sets in $\mathcal{X}$ *bags* of the tree decomposition $(T, \mathcal{X})$. The *width* of $(T, \mathcal{X})$ is equal to $\max\{|X_t| - 1 \mid t \in V(T)\}$ and the *treewidth* of $G$ is the minimum width over all tree decompositions of $G$. A *nice tree decomposition* is a pair $(T, \mathcal{X})$ where $(T, \mathcal{X})$ is a tree decomposition such that $T$ is a rooted tree and the following conditions are satisfied: (i) Every node of the tree $T$ has at most two children; (ii) if a node $t$ has two children $t_1$ and $t_2$, then $X_t = X_{t_1} = X_{t_2}$; and (iii) if a node $t$ has one child $t_1$, then either $|X_t| = |X_{t_1}| + 1$ and $X_{t_1} \subset X_t$ (in this case we call $t_1$ an *introduce node*) or $|X_t| = |X_{t_1}| - 1$ and $X_t \subset X_{t_1}$ (in this case we call $t_1$ a *forget node*). It is possible to transform a given tree decomposition $(T, \mathcal{X})$ into a nice tree decomposition $(T', \mathcal{X}')$ in time $O(|V| + |E|)$ [3]. A set $X \subseteq V(G)$ is a treewidth-$t$ modulator for a graph $G$ if the treewidth of $G - X$ is at most $t$.

### Boundaried graphs

A $t$-boundaried graph is a triple $(G, B, \ell_G)$ where $G$ is a graph, $B \subset V(G)$ of size $t$ with each vertex $v \in B$ having a unique label $\ell_G(v) \in \{1, \ldots, t\}$. We refer to $B$ as the *boundary* of $G$. We often refer to a $t$-boundaried graph simply as $G$ with the function $\partial(G)$ returning the boundary of $G$ and the function $\ell_G$ denoting the labeling function on the boundary. In other cases, it will be useful to mention the boundary explicitly, in which case we will use $(G, B)$. When the size of the boundary is irrelevant to the discussion, we simply use "boundaried graph" instead of "$t$-boundaried graph".

Let $G = (X, C, E)$ be a boundaried incidence graph with boundary $\partial(G) = \{b_1, \ldots, b_{|\partial(G)|}\}$, where $\partial(G) \subseteq (X \cup C)$ and $b_i = \ell_G^{-1}(i)$. We use $\vartheta(G)$ to denote the $|\partial(G)|$-length binary word given as: $\vartheta(G)[i] = 0$ if $b_i \in X$ and $\vartheta(G)[i] = 1$ otherwise.

▶ **Definition 5** (Boundary types and mergeability). *We say that two boundaried incidence graphs $G_1 = (X_1, C_1, E_1)$ and $G_2 = (X_2, C_2, E_2)$ have the same* boundary type *if $\vartheta(G_1) = \vartheta(G_2)$. If $G_1$ and $G_2$ have the same boundary type, then we say that $G_1$ and $G_2$ are* mergeable *if for every $u_1$, $v_1 \in \partial(G_1)$ and $u_2$, $v_2 \in \partial(G_2)$ such that $\ell_{G_1}(u_1) = \ell_{G_2}(u_2)$ and $\ell_{G_1}(v_1) = \ell_{G_2}(v_2)$, at most one out of $(u_1, v_1)$ and $(u_2, v_2)$ is an edge in $G_1$ and $G_2$ respectively.*

### Gluing operation and compatibility of boundaried graphs

A pair of boundaried incidence graphs $G_1$ and $G_2$ that are mergeable can be "glued" together along their boundaries to obtain a new incidence graph, which we denote by $G_1 \oplus G_2$. The gluing operation takes the disjoint union of $G_1$ and $G_2$ and identifies the vertices of $\partial(G_1)$ and $\partial(G_2)$ with the same label. If there are vertices $u_1$, $v_1 \in \partial(G_1)$ and $u_2$, $v_2 \in \partial(G_2)$ such that $\ell_{G_1}(u_1) = \ell_{G_2}(u_2)$ and $\ell_{G_1}(v_1) = \ell_{G_2}(v_2)$ then $G$ has vertices $u$ formed by unifying $u_1$ and $u_2$ and $v$ formed by unifying $v_1$ and $v_2$. That is, $u_1$ and $u_2$ are merged to form the

supernode $u$ and $v_1$ and $v_2$ are merged to form the supernode $v$. The new vertices $u$ and $v$ are adjacent if $(u_1, v_1) \in E(G_1)$ or $(u_2, v_2) \in E(G_2)$. The polarities of the edges in $G_1$ and $G_2$ are inherited in $G_1 \oplus G_2$ in the natural way.

▶ **Definition 6** ($d$-compatibility of boundaried graphs)**.** *Let $G_1$ and $G_2$ be two $t$-boundaried $K_{d,d}$-free incidence graphs. We say that $G_1$ and $G_2$ are* $d$-compatible *if they are mergeable and further, $G_1 \oplus G_2$ is also a $K_{d,d}$-free graph.*

### Minors and nowhere-dense graph classes

Given an edge $e = (x, y)$ of a graph $G$, the graph $G/e$ is obtained from $G$ by contracting the edge $e$, that is, the endpoints $x$ and $y$ are replaced by a new vertex $v_{xy}$ which is adjacent to the original neighbors of $x$ and $y$ (except $x$ and $y$). A graph $H$ obtained by a sequence of edge-contractions is said to be a *contraction* of $G$. We denote it by $H \leq_c G$. A graph $H$ is a *minor* of a graph $G$ if $H$ is the contraction of some subgraph of $G$ and we denote it by $H \leq_m G$. We say that a graph $G$ is *$H$-minor-free* when it does not contain $H$ as a minor. We also say that a graph class $\mathcal{G}$ is *$H$-minor-free* (or, excludes $H$ as a minor) when all its members are $H$-minor-free. It is well-known [36] that if $H \leq_m G$ then $tw(H) \leq tw(G)$.

The notion of *shallow minors* is required to define nowhere-dense graph classes. The radius of a connected graph $G$ is the minimum over all vertices $v$ of $G$ of the maximum distance between $v$ and another vertex. For non-negative integer $r$, a graph $H$ is a shallow minor at depth $r$ of a graph $G$ if there exists a subgraph $X$ of $G$ whose connected components have radius at most $r$, such that $H$ is a simple graph obtained from $G$ by contracting each component of $X$ into a single vertex and then taking a subgraph.

We now recall the notion of *nowhere-dense* graph classes. Although the only property of a nowhere-dense graph class we require is that they are biclique-free, we give the formal definition here for the sake of completeness.

▶ **Definition 7.** *A class $\mathcal{C}$ of graphs is nowhere dense if there is a function $f : \mathbb{N} \to \mathbb{N}$ such that for every $r \geq 0$, $K_{f(r)}$ is* not *a shallow minor at depth $r$ of the graph $G$, for every $G \in \mathcal{C}$.*

▶ **Proposition 8.** (see Fact 1, [40]) *Let $\mathcal{C}$ be a nowhere-dense class of graphs and let $f$ be the corresponding function as described in the above definition. Then, there is a number $t$ depending only on $f$ such that for any $G \in \mathcal{C}$, $G$ does not contain $K_{t,t}$ as a subgraph.*

### Unbreakability

Roughly speaking, a graph is unbreakable if it is not possible to "break" it into two large parts by removing only a small number of vertices. We now give the formal definition. A pair $(X, Y)$ where $X \cup Y = V(G)$ is a *separation* if there is no edge with one endpoint in $X \setminus Y$ and the other in $Y \setminus X$. The order of $(X, Y)$ is $|X \cap Y|$. If there exists a separation $(X, Y)$ of order at most $c$ such that $|X \setminus Y| \geq s$ and $|Y \setminus X| \geq s$, called an *$(s, c)$-witnessing separation*, then $G$ is *$(s, c)$-breakable*. Otherwise, $G$ is *$(s, c)$-unbreakable*. The following lemma states that it is possible to determine (approximately) whether a graph is unbreakable or not using a fixed-parameter algorithm, and lemmas similar to it can be found in [5, 30].

▶ **Proposition 9** ([30])**.** *There exists an algorithm that, given $s, c \in \mathbb{N}$ and a graph $G$, runs in time $2^{O(c(s+c))} \cdot n^3 \log n$ and either returns an $(\frac{s}{2^c}, c)$-witnessing separation or correctly concludes that $G$ is $(s, c)$-unbreakable.*

▶ **Observation 10.** *Let $t, \delta, \tau^\star \in \mathbb{N}$, where $t \leq 2\tau^\star$. Let $(G, Z)$ be a $t$-boundaried graph and let $(X, Y)$ be a $(\delta, \tau^\star)$-witnessing separation. Then, one of the pairs $(G[X], (X \cap Y) \cup (Z \cap X))$ or $(G[Y], (X \cap Y) \cup (Z \cap Y))$ is a $t'$-boundaried graph, where $t' \leq 2\tau^\star$.*

By repeatedly invoking the above observation, one can "zoom in" to a separation of small order in the graph such that one of the sides is large and induces an unbreakable subgraph.

▶ **Lemma 11.** *Let $\delta, \delta^\star, \tau^\star \in \mathbb{N}$ be such that $\delta^\star = 2^{\tau^\star} \cdot \delta$. Let $(X, Y)$ be a separation of order at most $2\tau^\star$ in a graph $G$ such that $|X| \geq \delta$. There is an algorithm that, given $G, X, Y$, runs in time $2^{\mathcal{O}(\tau^\star \cdot \delta^\star)} \cdot n^{\mathcal{O}(1)}$ and returns a separation $(\widetilde{X}, \widetilde{Y})$ of order at most $2\tau^\star$ in $G$ such that $|\widetilde{X}| \geq \delta$ and $G[\widetilde{X}]$ is $(\delta^\star, \tau^\star)$-unbreakable.*

▶ Remark 12. Roughly speaking, the notions of boundaried incidence graphs, mergeability, and $d$-compatibility allows us to split our original input (formula's) incidence graph along a small balanced separator (of size at most some $t$) and treat the two sides of the separator as $t$-boundaried $d$-compatible graphs and focus our attention on one of these two graphs in order to make further progress. The notion of unbreakability plus Lemma 11 gives us a kind of "base case" where we are able to stop splitting our instance along such small separators and use the additional structure imposed by unbreakability to make progress.

## 3 Algorithm for Weak Planar-$\mathcal{F}$-minor-free-Backdoor Detection

In this section, we discuss the proof of Theorem 1. We begin with the following definitions and assumptions.

- Recall that $d$ is fixed, i.e., it is a constant for us.
- We assume that $k > d$. Otherwise, we could simply guess (and verify) a weak-backdoor set of size at most $k$ in time $|\phi|^{\mathcal{O}(d)}$. Here, $k$ is the integer taken as input by the algorithm $\mathcal{A}$ described in Theorem 1.
- We assume that $\mathcal{F}$ is a family of graphs containing at least one planar graph and use $\eta$ to denote a computable upper bound on the treewidth of $\mathcal{F}$-minor-free graphs.
- Define $\tau(k, d) = k^{d+1}$, $\tau'(\eta, k, d) = \tau(k, d) + 2\eta + 2$.
- We pick $\delta(\eta, k, d)$ to be a large enough function of $\tau'(\eta, k, d)$ to be decided later.
- Set $\delta'(\eta, k, d) = 2^{\tau'(\eta,k,d)} \cdot \delta(\eta, k, d)$.
- Let $\Delta(\eta, k, d) = \delta'(\eta, k, d) + 2\tau'(\eta, k, d)$.
- For ease of readability, we omit the arguments of these functions when they are clear from the context, e.g., we replace $\Delta(\eta, k, d)$ with $\Delta$.

### 3.1 Overview of our algorithm

We first briefly describe the ideas used by Fomin et al. [13] in their randomized FPT algorithm to detect weak $\eta$-tw-backdoor sets when the input formula is $d$-CNF. We then outline the similarities and main differences with our algorithm.

#### Algorithm of Fomin et al [13]

Let $G$ be the input incidence graph. Say that a set $X \subseteq V(G)$ is an $\eta$-modulator if $\mathsf{tw}(G - X) \leq \eta$. Fomin et al. observed that if $X$ is a set of variables that form a weak $\eta$-tw-backdoor, then the set $N(X)$ of their neighbors in the incidence graph is an $\eta$-modulator. Note that $|N(X)|$ could be arbitrarily large compared to $|X|$. They then give a preprocessing procedure that ensures that for every $\eta$-tw-backdoor set $X$, the set $N(X)$ is incident to a large fraction of the edges in $G$. Therefore, if one picks an edge uniformly at random, then the clause endpoint of the edge belongs to $N(X)$ with constant probability. Further, since the clauses are of constant size $d$, we have that a randomly chosen variable from the clause belongs to $X$ with a constant probability, some $f(d, \eta)$. The algorithm then simply branches

on the chosen variable $x$ in the usual way – in one branch, the formula is simplified by setting $x$ to 1 and recurse, and in the other branch, $x$ is set to 0 and the residual instance is recursed upon.

The main obstacle in our setting is that simply obtaining a clause in $N(X)$ is not good enough since clauses can be arbitrarily large. This motivates us to look for a "domination core" within the incidence graph. That is, if we find sufficiently many clause vertices in $N(X)$, then it is possible to identify a small subset of variables that intersect $X$. However, being able to locate the required number of clause vertices in $N(X)$ is far from obvious even though we have access to the preprocessing rule of Fomin et al. For instance, each time we locate a vertex in $N(X)$ and remove it, the preprocessing rule could kick in again and even remove vertices of $X$. A similar issue arises in the recent work of Choudhary et al. [6] for Feedback Vertex Set on linear hypergraphs (whose incidence graphs are $K_{2,2}$-free). As a result, they needed to design problem specific preprocessing rules that heavily utilize the fact that they are dealing with deletion to graphs of treewidth 1. Moreover, their approach only works when the input incidence graph is $K_{2,2}$-free and extending it to our far more general setting appears to be a challenging task.

### Our approach

To overcome the obstacle described above, we take a different route and use the notion of unbreakability [5] to argue that if we "zoom into" a subgraph that is large enough, has high treewidth, is separated from the rest of the graph by a small boundary and is unbreakable, then it is possible to identify sufficiently many vertices from $N(X)$ and a *domination core* can be extracted from this [34, 11, 12]. This is the main technical contribution of the paper and we believe that this approach can find further applications.

Our algorithm relies on Lemma 11 and Lemmas 13-16. Out of these, the next three lemmas correspond to the three main cases that we will encounter in our algorithm. In the following three lemmas, let $G^\star$ denote the input incidence graph, let $G$ and $H$ be $K_{d,d}$-free $t$-boundaried $d$-compatible incidence graphs ($t \leq 2\tau'$) with the same boundary type such that $G^\star = G \oplus H$. Moreover, suppose that $|\partial(G)| \leq 2\tau'$ and $G$ is $(\delta', \tau')$-unbreakable. Recall that $\partial(G)$ and $\partial(H)$ denote the boundaries of $G$ and $H$ respectively. Moreover, we use $n$ to denote the number of vertices in the incidence graph under consideration.

The first lemma states that if $G$ has sufficiently high tree-width, then it is possible to compute a small family of clause-sets, each containing few clauses, such that if $G^\star$ contains a small variable set $S^\star$ such that deleting its neighborhood reduces the treewidth of $G$ significantly, then in at least one of these clause-sets, every clause contains a variable of $S^\star$. In graph-theoretic terms, it shows that one can compute a small family of small clause-vertex sets, such that in at least one of these small clause-vertex sets, every vertex is adjacent to $S^\star$.

▶ **Lemma 13** (Case 1: High treewidth piece)**.** *Suppose* $\mathsf{tw}(G) > \Delta$*. Let* $S^\star \subseteq \mathsf{var}(G^\star)$ *be such that* $|S^\star| \leq k$ *and* $\mathsf{tw}(G^\star - N_{G^\star}[S^\star]) \leq \eta$*. There is an algorithm that, given* $G^\star$*,* $G$ *and* $H$ *as input, runs in time* $(\delta')^{\mathcal{O}(d\tau')}n^{\mathcal{O}(1)}$*, and outputs a family* $\mathscr{Q}$ *of subsets of* $\mathsf{cla}(G^\star)$ *of size* $\tau$ *each, with the following properties.*

1. $|\mathscr{Q}| = (\delta')^{\mathcal{O}(d\tau')}$*, and*
2. *there exists* $\mathcal{Q} \in \mathscr{Q}$ *such that each clause in* $\mathcal{Q}$ *contains a variable from* $S^\star$*.*

The following lemma is used to handle a situation similar to that above, with the only difference being that the treewidth of $G$ is not too high. On the other hand, it is also high enough to ensure that any weak $\mathcal{F}$-minor-free-backdoor set of $G^\star$ intersects $G$ in at least one vertex. The proof of this lemma uses Arnborg et al.'s extension [1] of Courcelle's Theorem and the fact that the existence of a small weak $\mathcal{F}$-minor-free-backdoor set is CMSO-expressible (see, for example, [23]). We refer the reader to [7, 1, 8] for a detailed introduction to CMSO.

▶ **Lemma 14** (Case 2: Moderately-high treewidth piece). *Suppose that* $\eta < \mathsf{tw}(G - \partial(G)) \leq \Delta$. *We can compute a set* $S \subseteq \mathsf{var}(G)$ *in time* $g(\eta, k, d) \cdot n^{\mathcal{O}(1)}$ *for some computable function* $g$, *such that the following hold.*

1. $S$ *has size at most* $g(\eta, k, d)$, *and*
2. $S$ *has a non-empty intersection with a weak* $\mathcal{F}$*-minor-free-backdoor set of size at most* $k$ *for* $G \oplus H$ *(if one exists).*

The next lemma shows that if $G$ has low-treewidth but has a large number of vertices, then one can obtain a strictly smaller, equivalent instance by closely following the graph-replacement arguments used by Fomin et al. [13], which in turn was inspired by ideas used in [19] for kernelization.

▶ **Lemma 15** (Case 3: Low treewidth piece). *Suppose that* $|V(G)| \geq \delta$ *and* $\mathsf{tw}(G - \partial(G)) \leq \eta$. *Then, there is a* $K_{d,d}$*-free incidence graph* $G'$ *such that the following hold.*

1. $|V(G')| < |V(G^\star)|$.
2. *For every* $\gamma \leq k$, $\psi(G^\star)$ *has a weak* $\mathcal{F}$*-minor-free-backdoor set* $S$ *of size at most* $\gamma$ *if and only if* $\psi(G')$ *has a weak* $\mathcal{F}$*-minor-free-backdoor set* $S'$ *of size at most* $\gamma$.
3. *Given* $G^\star, G$ *and* $H$, *one can obtain* $G'$ *in time* $\widehat{g}(\mathcal{F}, k, d) \cdot n^{\mathcal{O}(1)}$ *for some computable function* $\widehat{g}$.

Having stated the three central lemmas that we will be using, we next demonstrate an immediate application of Lemma 13, following which we give the proof of Theorem 1 assuming the correctness of the three central lemmas.

Recall that Lemma 13 provides us a family of subsets of $\mathsf{cla}(G^\star)$ such that there is one subset where each clause contains a variable from $S$. In the following lemma we explain how to get a variable in $S$ from such a family. The proof the lemma uses the concept of *domination core*, which is used in the study of parameterized complexity of dominating set on $K_{d,d}$-free graphs [34, 11, 12]. Recall that $k > d$.

▶ **Lemma 16** (Domination core). *There is an algorithm that, given a set* $Z_C \subseteq \mathsf{cla}(\phi)$ *of* $\tau$ *clauses of a* $K_{d,d}$*-free formula* $\phi$, *each of whose variable set intersects some fixed variable set* $S \subseteq \mathsf{var}(\phi)$ *of size at most* $k$, *runs in polynomial time and outputs a set* $Z_V$ *of at most* $d$ *variables that intersects* $S$.

**Proof.** Let $Z_C^0 = Z_C$. We define $v_1, \ldots, v_r \in \mathsf{var}(\phi)$ and $\mathcal{C}_1, \ldots, \mathcal{C}_r \subseteq Z_C$ as follows. For every $i \in [r]$, we select $v_i$ to be the lexicographically first variable disjoint from the previously selected variables, that occurs most frequently among the clauses in the set $Z_C^{i-1}$. We set $Z_C^i := N_{\mathsf{inc}(\phi)}(v_i) \cap Z_C^{i-1}$. We now observe that for every $i \in [d]$, either $S$ intersects $\{v_1, \ldots, v_i\}$ or $|Z_C^i| \geq |Z_C^{i-1}|/k$. This is because $S$ has size at most $k$ and every clause in $Z_C$ contains a variable of $S$. In particular, we have that either $S$ intersects $Z_V = \{v_1, \ldots, v_d\}$ or it must be the case that $|Z_C^d| \geq k$ (since $|Z_C^0| \geq k^{d+1}$), implying the existence of a $K_{d,d}$ in $\mathsf{inc}(\phi)$, which is a contradiction. This completes the proof of the lemma. ◀

We are now ready to complete the proof of Theorem 1 assuming Lemmas 11-16. We restate it here for the reader's convenience.

▶ **Theorem 1.** *For every* $d \in \mathbb{N}$, *there is a computable function* $f_d$ *and an algorithm* $\mathcal{A}_d$ *that satisfies the following:*

1. $\mathcal{A}_d$ *takes as input a* $K_{d,d}$*-free formula* $\phi$, *a family* $\mathcal{F}$ *of graphs containing at least one planar graph, and an integer* $k$.
2. $\mathcal{A}_d$ *runs in time* $f_d(\mathcal{F}, k) \cdot |\phi|^{\mathcal{O}(d)}$.
3. $\mathcal{A}_d$ *either outputs a weak* $\mathcal{F}$*-minor-free-backdoor set of* $\phi$ *of size at most* $k$, *or concludes correctly that* $\phi$ *has no weak* $\mathcal{F}$*-minor-free-backdoor set of size at most* $k$.

**Proof.** We focus on solving the decision version of the problem. This is because, if the input is a yes-instance, then a weak $\mathcal{F}$-minor-free backdoor set of size at most $k$ can be computed using self-reducibility. Let $I = (\phi, \mathcal{F}, k)$ denote the input instance and let $G$ be the incidence graph of $\phi$. Suppose that $G$ has treewidth at most $\Delta$. Then, one can use Courcelle's theorem to solve the problem (see, for example, [23, 26]). Hence, we may assume that $G$ has treewidth $> \Delta$. In particular, $|V(G)| > \Delta$. Now, using Proposition 9, we either conclude that $G$ is $(\delta', \tau')$-unbreakable or we compute a $(\delta, \tau')$-witnessing separation.

In the first case, we define $\partial(G) = \emptyset$ and $H$ to be an edgeless graph with $\partial(H) = \emptyset$. We then invoke Lemma 13 to compute a set $\mathscr{Q}$ of size $\delta'^{\mathcal{O}(d\tau')}$ comprising sets of $\tau$ clauses each such that for some $\mathcal{Q} \in \mathscr{Q}$, each clause in $\mathcal{Q}$ contains a variable in some fixed weak backdoor set $S^\star$ (if one exists). Recall that $\mathsf{inc}(\phi) - N_{\mathsf{inc}(\phi)}[S^\star]$ has treewidth at most $\eta$, satisfying the premise of Lemma 13. We then invoke Lemma 16 on every $\mathcal{Q} \in \mathscr{Q}$ to compute a set $Z_V$ of at most $d \cdot \delta'^{\mathcal{O}(d\tau')}$ variables that intersects $S^\star$. For every $x \in Z_V$ and $\alpha \in \{0, 1\}$, we recursively solve the problem on the instance $I_{x,\alpha} = (\phi[x = \alpha], k - 1)$. We conclude that $I$ is a yes-instance if and only if there is an $x \in Z_V$ and $\alpha \in \{0, 1\}$ such that $I_{x,\alpha}$ is a yes-instance.

Now, consider the second case, i.e., the case where we compute a $(\delta, \tau')$-witnessing separation $(X, Y)$ for $G$. Notice that this separation satisfies the premises of Lemma 11 (by taking the same $\delta$, taking $\delta^\star$ be $\delta'$ and $\tau^\star$ to be $\tau'$). Hence, we execute the algorithm of this lemma to compute a separation $(\widetilde{X}, \widetilde{Y})$ of order at most $2\tau'$ in $G$ such that $|\widetilde{X}| \geq \delta$ and $G[\widetilde{X}]$ is $(\delta', \tau')$-unbreakable. Let $\widetilde{G} = G[\widetilde{X}]$ and let $\widetilde{H} = G[\widetilde{Y}]$. At this point, we have that $|V(\widetilde{G})| \geq \delta$, $|\partial(\widetilde{G})| \leq 2\tau'$ and $\widetilde{G}$ is $(\delta', \tau')$-unbreakable.

Consider the following three cases depending on whether the "piece" $\widetilde{G} - \partial(\widetilde{G})$ has high treewidth or moderately-high treewidth or low treewidth. In each case, we will either strictly reduce the size of the instance or compute a bounded set of variables upon which to branch.

**Case 1: $\mathsf{tw}(\widetilde{G} - \partial(\widetilde{G})) > \Delta$.** We invoke Lemma 13 with $G := \widetilde{G}$ and $H := \widetilde{H}$ to compute $\mathscr{Q}$ and then Lemma 16 on the sets in $\mathscr{Q}$ as described above. This gives us a set $Z_V$ of at most $d \cdot \delta'^{\mathcal{O}(d\tau')}$ variables that intersects a weak $\mathcal{F}$-minor-free-backdoor set $S^\star$ of $\phi$ of size at most $k$. Now, for every $x \in Z_V$ and $\alpha \in \{0, 1\}$, we recursively solve the problem on the instance $I_{x,\alpha} = (\phi[x = \alpha], \mathcal{F}, k - 1)$. We conclude that $I$ is a yes-instance if and only if there is an $x \in Z_V$ and $\alpha \in \{0, 1\}$ such that $I_{x,\alpha}$ is a yes-instance.

**Case 2: $\eta < \mathsf{tw}(\widetilde{G} - \partial(\widetilde{G})) \leq \Delta$.** We invoke Lemma 14 with $G := \widetilde{G}$ and $H := \widetilde{H}$ to compute a set $S \subseteq \mathsf{var}(G)$ of size at most $g(\eta, k, d)$ that has a non-empty intersection with a weak $\mathcal{F}$-minor-free-backdoor set of $G$ of size at most $k$ (if one exists). For every $x \in S$ and $\alpha \in \{0, 1\}$, we recursively solve the problem on the instance $I_{x,\alpha} = (\phi[x = \alpha], \mathcal{F}, k-1)$. We conclude that $I$ is a yes-instance if and only if there is an $x \in S$ and $\alpha \in \{0, 1\}$ such that $I_{x,\alpha}$ is a yes-instance.

**Case 3: $\mathsf{tw}(\widetilde{G} - \partial(\widetilde{G})) \leq \eta$.** We invoke Lemma 15 with $G := \widetilde{G}$ and $H := \widetilde{H}$ to compute an equivalent instance $(\phi', \mathcal{F}, k)$ such that $|\phi'| < |\phi|$ and recurse on $(\phi', \mathcal{F}, k)$.

This completes the description of our algorithm. Each step of the algorithm has running time $\chi(\eta, d, k)$ for some computable function $\chi$ and at the end of each, we either make $2d \cdot \delta'^{\mathcal{O}(d\tau')}$ recursive calls with a strictly smaller budget $k$ or strictly reduce the size of the instance. Hence, the running time follows. The correctness is derived from that of the three main lemmas (Lemmas 13, 14, 15) and the domination core lemma (Lemma 16). ◀

It remains to prove the three main lemmas. Lemma 13 captures the new technical insight at the heart of this work, and so we focus on that lemma in this extended abstract. We also give a proof sketch of Lemma 14 and as the proof of Lemma 15 closely mirrors the algorithm of Fomin et al. [13], we omit the details of this lemma in this extended abstract.

## 3.2 Handling unbreakable boundaried instances

### Protrusions

For a graph $G$ and $S \subseteq V(G)$, we define $\partial_G(S)$ as the set of vertices in $S$ that have a neighbor in $V(G) \setminus S$. For a set $S \subseteq V(G)$ the *neighborhood* of $S$ is $N_G(S) = \partial_G(V(G) \setminus S)$. When it is clear from the context, we omit the subscripts. An *$r$-protrusion* in a graph $G$ is a set $X \subseteq V$ such that $|\partial(X)| \leq r$ and $\mathsf{tw}(G[X]) \leq r$. The *size* of a protrusion is the number of vertices in it. An $\alpha$-cover in $G$ is a set $S$ such that $\sum_{v \in S} d(v)) \geq \alpha \cdot \sum_{v \in V(G)} d(v) = 2\alpha|E(G)|$. Here, $d(v)$ denotes the degree of the vertex $v$.

We begin with the following lemma that states that if a graph does not have large $r$-protrusions and there is a set $Z$ of vertices whose deletion results in a graph of constant treewidth, then a constant fraction of the edge-set is incident on $Z$.

▶ **Lemma 17.** *Let $r = 2\eta + 2$, and $\alpha = \frac{1}{18(\eta+1)^2}$. Let $J$ be a graph and $Z \subseteq V(J)$ such that $\mathsf{tw}(J - Z) \leq \eta$. If $J$ has no $r$-protrusion of size at least $r'$, then $Z$ is an $\frac{\alpha}{r'}$-cover of $J$.*

Lemma 17 can be inferred directly from the proof of Lemma 3 in [14], although it does not state concrete values for the various parameters involved in the statement. The following lemma provides a procedure to extract, from a given $K_{d,d}$-free incidence graph with no *large* protrusions, a small set of clauses such that at least one of the clauses in this set contains a backdoor variable.

▶ **Lemma 18** (Protrusion-free instances)**.** *Let $J$ be a $K_{d,d}$-free incidence graph and let $k, r' \in \mathbb{N}$. Let $S \subseteq \mathsf{var}(J)$ be a non-empty inclusion-wise minimal set such that $|S| \leq k$ and $\mathsf{tw}(J - N[S]) \leq \eta$. Moreover, suppose that $J$ has no $(2\eta + 2)$-protrusion of size at least $r'$. There is an algorithm that, given $J$, runs in polynomial time, and outputs a subset $\mathcal{D}$ of $\mathsf{cla}(J)$ such that $|\mathcal{D}| = (k \cdot \eta \cdot r')^{\mathcal{O}(d)}$, and there exists a clause $C \in \mathcal{D}$ that contains a variable from $S$.*

**Proof.** Since $J$ is an incidence graph, $S$ is a variable set, and $\mathsf{tw}(J - N[S]) \leq \eta$, it follows that $\mathsf{tw}(J - N(S)) \leq \eta$. Invoking Lemma 17 with $Z = N(S)$, we infer that $N(S)$ is an $\frac{\alpha'}{r'}$-cover of $J$, where $\alpha' = \frac{1}{18(\eta+1)^2}$. Let $\alpha = \frac{\alpha'}{k \cdot r'}$, $g = \frac{\alpha}{2d}$, and $h = \prod_{i=0}^{d-1}(\alpha - ig)$. Let $X = \mathsf{var}(J)$ and $\mathcal{C} = \mathsf{cla}(J)$. Now we define a weight function $w$ on $\mathcal{C}$ as follows. For every $C \in \mathcal{C}$, $w(C) = \frac{d(C)}{m}$, where $m$ is the number of edges in $J$ and $d(C)$ denotes the degree of the vertex $C$ in $J$. Extending the weight function to sets in the natural way, we conclude that $w(\mathcal{C}) = \sum_{C \in \mathcal{C}} w(C) = 1$ by definition. It will be useful to keep in mind that picking an edge of $J$ uniformly at random and picking the clause-vertex that appears as one of the endpoints of this edge corresponds to picking a vertex $C \in \mathcal{C}$ with probability $w(C)$.

Next, we identify sufficiently small sets $Y \subseteq X$ and $\mathcal{D}' \subseteq \mathcal{C}$ and prove that either $S \cap Y \neq \emptyset$ or $N_J(S) \cap \mathcal{D}' \neq \emptyset$. The sets $\mathcal{D}'$ and $Y$ are defined as follows: $\mathcal{D}'$ is the set of vertices in $\mathcal{C}$ whose weight is greater than $g$ and $Y$ is the set of vertices in $X$ such that it has no neighbor in $\mathcal{D}'$, and the sum of weights on its neighbors is at least $\alpha$. Formally,

$$\mathcal{D}' = \{y \in \mathcal{C} \mid w(y) > g\}$$

$$Y = \{x \in X \setminus N(\mathcal{D}') \mid w(N(x)) \geq \alpha\}$$

Notice that for every $x \in Y$, $|N(x) \setminus \mathcal{D}'| \geq 2d$. This is because $w(N(x)) \geq \alpha = 2gd$ and every clause in $\mathcal{C} \setminus \mathcal{D}'$ has weight at most $g$.

▷ Claim 19. Either $N(S) \cap \mathcal{D}' \neq \emptyset$ or $S \cap Y \neq \emptyset$.

We next claim that the sets $\mathcal{D}'$ and $Y$ are sufficiently small for our purposes.

▷ Claim 20. $|\mathcal{D}'| \leq 36(\eta+1)^2 d \cdot k \cdot r'$ and if $N(S) \cap \mathcal{D}' = \emptyset$, then $|Y| \leq d \cdot (36k \cdot r'(\eta+1)^2)^d$.

Given the above two claims, the algorithm is straightforward. We first construct $\mathcal{D}'$ and $Y$. If $|Y| \leq d \cdot (36k \cdot r'(\eta+1)^2)^d$, then we output a set $\mathcal{D}$ which is the union of $\mathcal{D}'$ and a set of clauses comprising one arbitrary neighbor of each vertex in $Y$. Otherwise, we simply output $\mathcal{D} = \mathcal{D}'$. From Claim 20, it follows that the size of $\mathcal{D}$ is bounded by $(k \cdot \eta \cdot r')^{\mathcal{O}(d)}$. Moreover, from both claims above, we conclude that some clause in $\mathcal{D}$ contains a variable in $S$. Since computing $\mathcal{D}'$ and $Y$ can be done in polynomial time, this algorithm runs in polynomial time as required. This completes the proof of the lemma. ◀

In the following, let $G$ and $H$ be $K_{d,d}$-free $t$-boundaried $d$-compatible incidence graphs ($t \leq 2\tau'$) with the same boundary type, $|\partial(G)| \leq 2\tau'$ and $G$ is $(\delta', \tau')$-unbreakable. Recall that $\partial(G)$ and $\partial(H)$ denote the boundaries of $G$ and $H$ respectively. Throughout this section, we denote by $G^\star$, the graph $G \oplus H$.

▶ **Lemma 13** (Case 1: High treewidth piece). *Suppose* $\mathsf{tw}(G) > \Delta$. *Let* $S^\star \subseteq \mathsf{var}(G^\star)$ *be such that* $|S^\star| \leq k$ *and* $\mathsf{tw}(G^\star - N_{G^\star}[S^\star]) \leq \eta$. *There is an algorithm that, given* $G^\star$, $G$ *and* $H$ *as input, runs in time* $(\delta')^{\mathcal{O}(d\tau')} n^{\mathcal{O}(1)}$, *and outputs a family* $\mathscr{Q}$ *of subsets of* $\mathsf{cla}(G^\star)$ *of size* $\tau$ *each, with the following properties.*

1. $|\mathscr{Q}| = (\delta')^{\mathcal{O}(d\tau')}$, *and*
2. *there exists* $\mathcal{Q} \in \mathscr{Q}$ *such that each clause in* $\mathcal{Q}$ *contains a variable from* $S^\star$.

**Proof.** First we prove the following claim that will allow us to invoke Lemma 18 on certain subraphs of $G$.

▷ Claim 21. For every set $R \subseteq V(G)$ of size at most $\tau$, $G - R$ does not contain a $(2\eta+2)$-protrusion of size at least $\delta' + 2\eta + 2$

Proof. For the sake of contradiction, suppose there exists $R \subseteq V(G)$ of size at most $\tau$ and $A \subseteq V(G-R)$ such that $G[A]$ is a $(2\eta+2)$-protrusion of size at least $\delta' + 2\eta + 2$. That is, $|\partial_{G-R}(A)| \leq 2\eta+2$ and $|A| \geq \delta'+2\eta+2$. Set $X = A \cup R$ and $Y = R \cup (V(G) \setminus (A \setminus \partial_{G-R}(A)))$ and observe that $(X, Y)$ is a separation in $G$ of order at most $|R|+|\partial_{G-R}(A)| \leq \tau+2\eta+2 = \tau'$, and $|X \setminus Y| = |A \setminus \partial_{G-R}(A)| \geq \delta'$. We now claim that $|Y \setminus X| = |V(G) \setminus (A \cup R)| \geq \delta'$. Suppose to the contrary that $|V(G) \setminus (A \cup R)| < \delta'$. Then, $\mathsf{tw}(G) \leq \mathsf{tw}(G[A]) + |V(G) \setminus (A \cup R)| + |R| \leq 2\eta + 2 + \delta' + \tau \leq \Delta$, which is a contradiction to the premise of the lemma. ◁

We next define a recursive procedure FindClauses. The input to FindClauses is $G$, a set $R \subseteq \mathsf{cla}(G)$ and an integer $q \in \{1, \ldots, \tau\}$ such that $|R| + q = \tau$. The output is a family $\mathscr{Q}'$ of subsets of $\mathsf{cla}(G) \setminus R$ of size $q$ each, such that $|\mathscr{Q}'| \leq (k \cdot \eta \cdot \delta')^{c \cdot d \cdot q}$, for some constant $c > 0$. The following are the steps of the procedure FindClauses$(G, R, q)$.

1: First we apply Lemma 18 where $J = G - R$ and $r' = \delta' + 2\eta + 2$ (the above claim guarantees that the premise of this lemma is satisfied by this choice of $J$ and $r'$). Let $\mathcal{D}$ be the output of the algorithm of Lemma 18.
2: If $q = 1$, then we output $\{\{C\} \mid C \in \mathcal{D}\}$ and stop.
3: If $q > 1$, then for each $C \in \mathcal{D}$, we recursively call the procedure FindClauses with input $G$, $R \cup \{C\}$ and $q - 1$ to obtain a set $\mathscr{Q}_C$. Let $\mathscr{Q}'_C = \{Q \cup \{C\} \mid Q \in \mathscr{Q}_C\}$. We return $\mathscr{Q}' = \bigcup_{C \in \mathcal{D}} \mathscr{Q}'_C$ and stop.

Let $c$ be a constant such that the output of Lemma 18 (i.e., $\mathcal{D}$) has cardinality bounded by $(k \cdot \eta \cdot r')^{c \cdot d}$. It can be proved that $|\mathscr{Q}'| \leq (k \cdot \eta \cdot \delta')^{c \cdot d \cdot q}$ by induction on $q$.

As the number of nodes in the recurrence tree is bounded by $|\mathscr{Q}'|$ and the algorithm in Lemma 18 runs in polynomial time, the running time of the procedure FindClauses is bounded by $|\mathscr{Q}'| \cdot n^{\mathcal{O}(1)}$.

Before moving to the description of the main algorithm of this lemma, we develop some additional notation. For every $Z \subseteq \partial(G)$, let $Z' = \mathsf{cla}(Z) \cup \{c \in \mathsf{cla}(G) \mid \exists v \in c : v \in \mathsf{var}(Z)\}$ and let $Z''$ be the subset of $Z'$ comprising the (lexicographically) first $\min\{\tau, |Z'|\}$ elements. In other words, if $Z'$ has at least $\tau$ elements, then we truncate it to size $\tau$ and obtain $Z''$ and otherwise, $Z'' = Z'$.

We are now ready to describe our main algorithm. We begin by setting $\mathscr{Q} := \emptyset$. We then compute $Z''$ for every $Z \subseteq \partial(G)$ and for every $Z''$ such that $|Z''| = \tau$, we set $\mathscr{Q} = \mathscr{Q} \cup \{Z''\}$. For every $Z''$ such that $|Z''| < \tau$, we set $\mathscr{Q}'_{Z''} \leftarrow \mathsf{FindClauses}(G, Z'', \tau - |Z''|)$ and we set $\mathscr{Q} = \mathscr{Q} \cup \{\{Q\} \cup Z'' \mid Q \in \mathscr{Q}'_{Z''}\}$. When we have completed iterating over all $Z \subseteq \partial(G)$ and updating $\mathscr{Q}$, we return it.

The correctness of the algorithm results from the following claim.

▷ Claim 22. There exists $\mathcal{Q} \in \mathscr{Q}$ such that each clause in $\mathcal{Q}$ contains a variable from $S^\star$.

It remains to argue the size bound on the output family and the running time of the algorithm. Since $q \leq \tau$ in every call to FindClauses and we have at most $2^{|\partial(G)|} \leq 2^{\tau'}$ such calls it follows that the cardinality of the output family is bounded by $(k \cdot \eta \cdot \delta')^{c \cdot d \cdot \tau} \cdot 2^{2\tau'}$ which is upper bounded by $(\delta')^{\mathcal{O}(d\tau')}$. By the same reasoning and from the upper bound on the running time of FindClauses, the running time of this algorithm is $(\delta')^{\mathcal{O}(d\tau')} n^{\mathcal{O}(1)}$. ◀

## 3.3 Handling boundaried instances with moderately high treewidth

▶ **Definition 23** (Label-wise isomorphism). *Let $G_1$ and $G_2$ be two graphs, and let $t$ be a fixed positive integer. For $i \in \{1, 2\}$, let $f_{G_i}$ be a function that associates with every vertex of $V(G_i)$ some subset of $[t]$. The image of a vertex $v \in G_i$ under $f_{G_i}$ is called the label of that vertex. We say that that $G_1$ is* label-wise isomorphic *to $G_2$, and denote it by $G_1 \cong_t G_2$, if there is a map $h : V(G_1) \to V(G_2)$ such that (a) $h$ is one to one and onto; (b) $(u, v) \in E(G_1)$ if and only if $(h(u), h(v)) \in E(G_2)$ and (c) $f_{G_1}(v) = f_{G_2}(h(v))$. We call $h$ a* label-preserving isomorphism.

Notice that the first two conditions of Definition 23 simply indicate that $G_1$ and $G_2$ are isomorphic. Now, let $G$ be a $t$-boundaried graph with $t$ distinguished vertices, uniquely labeled from 1 to $t$. Given such a $t$-boundaried graph $G$, we define a canonical labeling function $\mu_G : V(G) \to 2^{[t]}$. The function $\mu_G$ maps every distinguished vertex $v$ with label $\ell \in [t]$ to the set $\{\ell\}$, that is $\mu_G(v) = \{\ell\}$, and for all vertices $v \in V(G) \setminus \partial(G)$ we have that $\mu_G(v) = \emptyset$.

Next we define a notion of labeled edge contraction. Let $H$ be a graph together with a function $f_H : V(H) \to 2^{[t]}$ and $(u, v) \in E(H)$. Furthermore, let $H'$ be the graph obtained from $H$ by identifying the vertices $u$ and $v$ into $w_{uv}$, removing all the parallel edges and removing all the loops. Then by *labeled edge contraction* of an edge $(u, v)$ of a graph $H$, we mean obtaining a graph $H'$ with the label function $f_{H'} : V(H') \to 2^{[t]}$. For $x \in V(H') \cap V(H)$ we have that $f_{H'}(x) = f_H(x)$ and for $w_{uv}$ we define $f_{H'}(w_{uv}) = f_H(u) \cup f_H(v)$. Now we introduce a notion of labeled minors of a $t$-boundaried graph. Let $H$ be a graph together with a function $f : V(H) \to 2^{[t]}$ and $G$ be a $t$-boundaried graph with canonical labeling function $\mu_G$. A graph $H$ is called a labeled minor of $G$, if we can obtain a labeled isomorphic copy of $H$ from $G$ by performing edge deletion and labeled edge contraction. The *$h$-folio* of a $t$-boundaried graph $G$ is the set $\mathcal{M}_h(G)$ of all $t$-labeled minors of $G$ on at most $h$ vertices.

▶ **Lemma 14** (Case 2: Moderately-high treewidth piece)**.** *Suppose that* $\eta < \mathsf{tw}(G - \partial(G)) \leq \Delta$. *We can compute a set* $S \subseteq \mathsf{var}(G)$ *in time* $g(\eta, k, d) \cdot n^{\mathcal{O}(1)}$ *for some computable function* $g$, *such that the following hold.*

1. $S$ *has size at most* $g(\eta, k, d)$, *and*
2. $S$ *has a non-empty intersection with a weak* $\mathcal{F}$*-minor-free-backdoor set of size at most* $k$ *for* $G \oplus H$ *(if one exists).*

**Proof.** Fix a weak $\mathcal{F}$-minor-free-backdoor set $S^\star$ in $G^\star$. Since $G - \partial(G)$ has treewidth at least $\eta + 1$, it must be the case that $S^\star$ intersects $V(G)$. We ensure that the variables in $\partial(G)$ are added to the returned set $S$ and so, we may assume henceforth that $S^\star$ is disjoint from $\partial(G)$.

Let $S_1^\star = S^\star \cap V(G)$. Let $\tau^\star : S^\star \to \{0,1\}$ be a partial assignment such that the incidence graph of $\phi[\tau^\star]$ is $\mathcal{F}$-minor free and let $\tau : \mathsf{var}(\phi) \to \{0,1\}$ be a satisfying assignment for $\phi$ that extends $\tau^\star$. Let $\tau_1^\star$ denote the restriction of $\tau^\star$ to $S_1^\star$. Let $Z_{kill} \subseteq \partial(G)$ denote those clauses in $\partial(G)$ that are satisfied by $\tau^\star$ and hence do not appear in $\phi[\tau^\star]$. Let $Z_{satext} \subseteq \partial(G)$ denote those clauses on the boundary that survive in $\phi[\tau^\star]$ and have at least one neighbor disjoint from $V(G)$ that satisfies it according to $\tau$. Consider the $t'$-boundaried graph $G_1'$ obtained from $G$ by restricting the vertex set to those vertices that survive in $\mathsf{inc}(\phi[\tau^\star])$. Notice that the boundary of $G'$ is precisely $\partial(G) \setminus Z_{kill}$. Finally, consider the $h$-folio $\mathcal{M}_h(G_1')$ where $h$ is an upper bound on the largest graph in $\mathcal{F}$. The labels of the graphs in $\mathcal{M}_h(G_1')$ correspond to the vertices in $\partial(G) \setminus Z_{kill}$.

Now, let $S_2^\star \subseteq \mathsf{var}(G) \setminus \partial(G)$, $\tau_2^\star : S_2^\star \to \{0,1\}$ and $\widetilde{\tau} : \mathsf{var}(G) \to \{0,1\}$ such that the following hold:

- $|S_2^\star| \leq |S_1^\star|$,
- $\widetilde{\tau}$ extends $\tau_2^\star$, $\widetilde{\tau}$ coincides with $\tau$ on $\mathsf{var}(\partial(G))$ and $\widetilde{\tau}$ satisfies all clauses in $\mathsf{cla}(G) \setminus (Z_{satext} \cup Z_{kill})$,
- $\mathcal{M}_h(G_1') = \mathcal{M}_h(G_2')$ where $G_2'$ is the graph obtained from $G$ by deleting $Z_{kill}$ and then restricting the vertex set to those vertices that survive after instantiating the vertices in $S_2^\star$ with $\tau_2^\star$.

▷ Claim 24. $\widehat{S} = (S^\star \setminus S_1^\star) \cup S_2^\star$ is also a weak backdoor set of $\phi$ of size at most $k$ to $\mathcal{F}$-minor-free formulas.

In order to compute the required set $S$, it is sufficient to iterate over all $r \in [k]$ (guessing $|S_1^\star|$), assignments to the variables in $\partial(G)$ (guessing $\tau$ on the variables in the boundary), all possible (pairs) of disjoint subsets of $\mathsf{cla}(\partial(G))$ (guessing $Z_{satext}$ and $Z_{kill}$) and all possible $t'$-labeled minors on at most $h$ vertices (guessing $\mathcal{M}_h(G_1')$) and decide whether there exists a set $S_2^\star$ that satisfies the properties listed above and if yes, compute one. Finally, we return the union of all such sets. Notice that the number of such iterations is bounded by a function of $\eta, k, d$ and hence the returned set has size at most $g(\eta, k, d)$ for some computable function $g$. Moreover, a set of the required kind in each iteration can be computed using Courcelle's theorem (see [23, 26] for MSO sentences capturing weak backdoors and minor exclusion). ◀

## 4 Hardness results for Backdoor Detection

Recall that an empty formula is a CNF formula with no clauses. Such formulas are trivially satisfiable.

▶ **Lemma 25.** *Let* $d, d' \geq 2$ *be two positive integers. There is a polynomial-time algorithm that takes as input* $d, d'$, *an instance* $(U, \mathcal{S}, k)$ *of* Set Cover *and outputs a CNF formula* $\phi$ *with the following properties:*

1. *If* $(U, \mathcal{S}, k)$ *is a yes-instance of* Set Cover*, then* $\phi$ *has a strong/weak backdoor set of size at most* $k$ *to the class of empty formulas.*
2. *If* $\phi$ *has strong/weak backdoor set of size at most* $k$ *to any subclass of* $\mathcal{K}_{d,d'}$*-free formulas, then* $(U, \mathcal{S}, k)$ *is a yes-instance of* Set Cover*.*

**Proof.** Let $(U, \mathcal{S}, k)$ be the given instance of Set Cover. Now, we construct a CNF formula $\phi$ as follows. For each element $x \in U$, we have a set of $d'(k+1)$ variables $\{x_{i,j} \mid i \in [d'], j \in [k+1]\}$. For each $S \in \mathcal{S}$, we have a set of $d$ variables $\{y_{S,i} \mid i \in [d]\}$. Thus,

$$\mathsf{var}(\phi) = \{x_{i,j} \mid x \in U, i \in [d'], j \in [k+1]\} \cup \{y_{S,i} \mid S \in \mathcal{S}, i \in [d]\}.$$

Next, we explain the construction of the set of clauses of $\phi$. For each element $x \in U$, let $\mathcal{S}_x$ be the family of sets in $\mathcal{S}$ that contain $x$. For each $x \in U$, we make $d'(k+1)+1$ clauses as follows.

$$C_x = \{y_{S,i}, \overline{y}_{S,i} \mid S \in \mathcal{S}_x, i \in [d]\}, \text{ and}$$

$$\text{for all } i \in [d'] \text{ and } j \in [k+1], \quad T_x(i,j) = C_x \cup \{x_{i,j}, \overline{x}_{i,j}\}.$$

This completes the construction of $\phi$. It is easy to verify that the construction can be done in polynomial time.

Next we prove the correctness of the algorithm. Consider the first statement. Let $\mathcal{P}$ be a solution for $(U, \mathcal{S}, k)$ of Set Cover. We claim that $Z = \{y_{S,1} \mid S \in \mathcal{P}\}$ is a strong backdoor set as well as a weak backdoor set of $\phi$ of size at most $k$ to the class of empty formulas. Since $\mathcal{P}$ is a set cover, we have that for any $x \in U$, $C_x$ contains a variable from $Z$, which appears both positively and negatively in $C_x$. Thus, for any assignment for the variables in $Z$, $C_x$ is satisfied for all $x \in U$. This also implies that for any $x \in U, i \in [d']$, and $j \in [k+1]$, $T_x(i,j)$ is satisfied because $C_x \subseteq T_x(i,j)$. In other words, every clause is satisfied by every assignment to $Z$. This shows that $Z$ is a strong backdoor set of $\phi$ of size at most $k$ to the class of empty formulas. Since each such sub-formula is trivially satisfiable, we also conclude that every assignment to the variables in $Z$ can be extended to a satisfying assignment of $\phi$. This implies that $Z$ is also a weak backdoor set of $\phi$ of size at most $k$ to the class of empty formulas.

We now prove the second statement. Let $Z$ be a strong or weak $\mathcal{C}$-backdoor set of size at most $k$ for the CNF formula $\phi$, where $\mathcal{C}$ is a subclass of $\mathcal{K}_{d,d'}$-free formulas. Let,

$$\mathcal{P} = \{S \in \mathcal{S} \mid \text{ there exists } j \in [d] \text{ such that } y_{S,j} \in Z\}.$$

We prove that $\mathcal{P}$ is indeed a solution for the Set Cover instance $(U, \mathcal{S}, k)$. Since $|Z| \leq k$, we have that $|\mathcal{P}| \leq k$. For the sake of contradiction assume that $\mathcal{P}$ is not a set cover of $(U, \mathcal{S})$. Then, there exists an element $x \in U$ that does not belong to any set in $\mathcal{P}$. This implies that none of the variables in $C_x$ is present in $Z$. Consider the set of clauses $\mathcal{T} = \{T_x(i,j) \mid i \in [d'], j \in [k+1]\}$. $C_x$ is a subset of every clause in $\mathcal{T}$. Moreover, the clauses in $\{T_x(i,j) \setminus C_x \mid i \in [d'], j \in [k+1]\}$ have pairwise disjoint variable sets. Since $\mathsf{var}(C_x) \cap Z = \emptyset$ and $|Z| \leq k$, there exists $j \in [k+1]$ such that the clauses $T_x(1,j), \ldots T_x(d',j)$ do not contain a variable from $Z$. Let $Q$ be an arbitrary set in $\mathcal{S}$ containing $x$. Notice that $y_{Q,1}, \ldots, y_{Q,d} \notin Z$ because of our assumption that $x$ is not covered by $\mathcal{P}$. As $y_{Q,1}, \ldots, y_{Q_d} \in \mathsf{var}(C_x)$, we have that $y_{Q,1}, \ldots, y_{Q,d} \in \mathsf{var}(T_x(i,j))$ for all $i \in [d']$. That is, the subgraph of $\mathsf{inc}(\phi)$ induced on $y_{Q,1}, \ldots, y_{Q,d}$ and $T_x(1,j), \ldots, T_x(d',j)$ forms a $K_{d,d'}$. Moreover this subgraph remains in the incidence graph of the formula obtained by simplifying $\phi$ based on any assignment to $Z$. This is a contradiction to the assumption that $Z$ is a strong/weak $\mathcal{C}$-free backdoor set for $\phi$. This completes the proof of the second statement. ◀

Lemma 25 implies Theorem 4 as argued in Section 1.

## 5 Conclusion

We have shown the fixed-parameter tractability of SAT parameterized by the size of a smallest weak backdoor to bounded-treewidth formulas, when the input belongs to the class of formulas whose incidence graphs are $K_{d,d}$-free. This implies the same result when the incidence graph of the input formula excludes a fixed (topological) minor or is from a class of bounded expansion or even from a nowhere-dense graph class. Thus, our main result advances our understanding of the parameterized complexity of SAT well beyond previously studied sparse inputs. In fact, we have shown that all nowhere-dense input formulas are tractable in this setting and moreover, by extending our result to biclique-free formulas, we have also identified natural classes of tractable formulas incomparable with nowhere-dense classes, i.e., formulas of bounded degeneracy.

We also reiterate that our main objective was to obtain a classification result and so, we have not attempted to optimize the running time and relied on invocations to Courcelle's theorem where it is standard in the literature on backdoor set detection. However, given the single-exponential (in $k$) running time of the algorithms in [13] that significantly improved upon similarly large exponential factors in [23, 26], it is not unreasonable to expect targeted research in this direction to yield similar speed-ups. A promising starting point towards this could be to consider subclasses of biclique-free graphs (e.g., graphs of bounded expansion) and identify those input classes where such a single-exponential dependence on $k$ can be achieved.

## References


1 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12:308–340, 1991.

2 Christian Bessiere, Clément Carbonnel, Emmanuel Hebrard, George Katsirelos, and Toby Walsh. Detecting and exploiting subproblem tractability. In Francesca Rossi, editor, *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*. IJCAI/AAAI, 2013.

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

4 Clément Carbonnel, Martin C. Cooper, and Emmanuel Hebrard. On backdoors to tractable constraint languages. In *Principles and Practice of Constraint Programming - 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings*, volume 8656 of *Lecture Notes in Computer Science*, pages 224–239. Springer Verlag, 2014.

5 Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. *SIAM J. Comput.*, 45(4):1171–1229, 2016. doi:10.1137/15M1032077.

6 Pratibha Choudhary, Lawqueen Kanesh, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. Parameterized complexity of feedback vertex sets on hypergraphs. In Nitin Saxena and Sunil Simon, editors, *40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2020, December 14-18, 2020, BITS Pilani, K K Birla Goa Campus, Goa, India (Virtual Conference)*, volume 182 of *LIPIcs*, pages 18:1–18:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.FSTTCS.2020.18.

7 Bruno Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Information and Computation*, 85:12–75, 1990.

8 Bruno Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. *Handbook of Graph Grammars*, pages 313–400, 1997.

**9** Y. Crama, O. Ekin, and P. L. Hammer. Variable and term removal from Boolean formulae. *Discr. Appl. Math.*, 75(3):217–230, 1997.

**10** Rina Dechter and Judea Pearl. Tree clustering for constraint networks. *Artif. Intell.*, pages 353–366, 1989.

**11** Pål Grønås Drange, Markus Sortland Dregi, Fedor V. Fomin, Stephan Kreutzer, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, Felix Reidl, Fernando Sánchez Villaamil, Saket Saurabh, Sebastian Siebertz, and Somnath Sikdar. Kernelization and sparseness: the case of dominating set. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*, pages 31:1–31:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.STACS.2016.31.

**12** Eduard Eiben, Mithilesh Kumar, Amer E. Mouawad, Fahad Panolan, and Sebastian Siebertz. Lossy kernels for connected dominating set on sparse graphs. *SIAM J. Discret. Math.*, 33(3):1743–1771, 2019. doi:10.1137/18M1172508.

**13** Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, M. S. Ramanujan, and Saket Saurabh. Solving $d$-sat via backdoors to small treewidth. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 630–641. SIAM, 2015. doi:10.1137/1.9781611973730.43.

**14** Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar F-deletion: Approximation, kernelization and optimal FPT algorithms. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 470–479. IEEE, 2012.

**15** Eugene C. Freuder. A sufficient condition for backtrack-bounded search. *J. ACM*, 32(4):755–761, 1985. doi:10.1145/4221.4225.

**16** Robert Ganian, Sebastian Ordyniak, and M. S. Ramanujan. Going beyond primal treewidth for (M)ILP. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 815–821. AAAI Press, 2017. URL: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14272.

**17** Robert Ganian, M. S. Ramanujan, and Stefan Szeider. Discovering archipelagos of tractability for constraint satisfaction and counting. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1670–1681. SIAM, 2016. doi:10.1137/1.9781611974331.ch114.

**18** Robert Ganian, M. S. Ramanujan, and Stefan Szeider. Combining treewidth and backdoors for CSP. In Heribert Vollmer and Brigitte Vallée, editors, *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, volume 66 of *LIPIcs*, pages 36:1–36:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.STACS.2017.36.

**19** Valentin Garnero, Christophe Paul, Ignasi Sau, and Dimitrios M Thilikos. Explicit Linear Kernels via Dynamic Programming. In *STACS*, pages 312–324, 2014.

**20** Serge Gaspers, Neeldhara Misra, Sebastian Ordyniak, Stefan Szeider, and Stanislav Zivny. Backdoors into heterogeneous classes of SAT and CSP. In Carla E. Brodley and Peter Stone, editors, *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 2652–2658. AAAI Press, 2014.

**21** Serge Gaspers, Sebastian Ordyniak, M. S. Ramanujan, Saket Saurabh, and Stefan Szeider. Backdoors to q-horn. *Algorithmica*, 74(1):540–557, 2016. doi:10.1007/s00453-014-9958-5.

**22** Serge Gaspers, Sebastian Ordyniak, and Stefan Szeider. Backdoor sets for CSP. In Andrei A. Krokhin and Stanislav Zivny, editors, *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*, pages 137–157. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/DFU.Vol7.15301.5.

**23** Serge Gaspers and Stefan Szeider. Backdoors to acyclic SAT. In *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 7391 of *Lecture Notes in Computer Science*, pages 363–374. Springer, 2012.

24 Serge Gaspers and Stefan Szeider. Strong backdoors to nested satisfiability. In Alessandro Cimatti and Roberto Sebastiani, editors, *Theory and Applications of Satisfiability Testing - SAT 2012 - 15th International Conference, Trento, Italy, June 17-20, 2012. Proceedings*, volume 7317 of *Lecture Notes in Computer Science*, pages 72–85. Springer, 2012. doi:10.1007/978-3-642-31612-8_7.

25 Serge Gaspers and Stefan Szeider. Backdoors to satisfaction. In *The Multivariate Algorithmic Revolution and Beyond*, pages 287–317, 2012. doi:10.1007/978-3-642-30891-8_15.

26 Serge Gaspers and Stefan Szeider. Strong backdoors to bounded treewidth sat. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 489–498. IEEE Computer Society, 2013.

27 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. *J. ACM*, 64(3):17:1–17:32, 2017. doi:10.1145/3051095.

28 Donald E. Knuth. Nested satisfiability. *Acta Informatica*, 28(1):1–6, 1990. doi:10.1007/BF02983372.

29 Stephan Kreutzer, Roman Rabinovich, and Sebastian Siebertz. Polynomial kernels and wideness properties of nowhere dense graph classes. *ACM Trans. Algorithms*, 15(2):24:1–24:19, 2019. doi:10.1145/3274652.

30 Daniel Lokshtanov, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Reducing CMSO model checking to highly connected graphs. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 135:1–135:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.135.

31 Jaroslav Nesetril and Patrice Ossona de Mendez. First order properties on nowhere dense structures. *J. Symb. Log.*, 75(3):868–887, 2010. doi:10.2178/jsl/1278682204.

32 Jaroslav Nesetril and Patrice Ossona de Mendez. On nowhere dense graphs. *Eur. J. Comb.*, 32(4):600–617, 2011. doi:10.1016/j.ejc.2011.01.006.

33 Naomi Nishimura, Prabhakar Ragde, and Stefan Szeider. Detecting backdoor sets with respect to Horn and binary clauses. In *Proceedings of SAT 2004 (Seventh International Conference on Theory and Applications of Satisfiability Testing, 10–13 May, 2004, Vancouver, BC, Canada)*, pages 96–103, 2004.

34 Geevarghese Philip, Venkatesh Raman, and Somnath Sikdar. Polynomial kernels for dominating set in graphs of bounded degeneracy and beyond. *ACM Trans. Algorithms*, 9(1):11:1–11:23, 2012. doi:10.1145/2390176.2390187.

35 Igor Razgon and Barry O'Sullivan. Almost 2-SAT is fixed parameter tractable. *J. of Computer and System Sciences*, 75(8):435–450, 2009.

36 Neil Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *J. Combin. Theory Ser. B*, 41(1):92–114, 1986.

37 Marko Samer and Stefan Szeider. Backdoor sets of quantified boolean formulas. In J. Marques-Silva and K. A. Sakallah, editors, *Proceedings of SAT 2007, Tenth International Conference on Theory and Applications of Satisfiability Testing, May 28-31, 2007, Lisbon, Portugal,*, volume 4501 of *Lecture Notes in Computer Science*, pages 230Â–243, 2007.

38 Marko Samer and Stefan Szeider. Constraint satisfaction with bounded treewidth revisited. *J. Comput. Syst. Sci.*, 76(2):103–114, 2010. doi:10.1016/j.jcss.2009.04.003.

39 Stefan Szeider. Matched formulas and backdoor sets. In J. Marques-Silva and K. A. Sakallah, editors, *Proceedings of SAT 2007, Tenth International Conference on Theory and Applications of Satisfiability Testing, May 28-31, 2007, Lisbon, Portugal,*, volume 4501 of *Lecture Notes in Computer Science*, pages 94–99, 2007.

40 Jan Arne Telle and Yngve Villanger. FPT algorithms for domination in sparse graphs and beyond. *Theor. Comput. Sci.*, 770:62–68, 2019. doi:10.1016/j.tcs.2018.10.030.

41 Ryan Williams, Carla Gomes, and Bart Selman. Backdoors to typical case complexity. In Georg Gottlob and Toby Walsh, editors, *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, IJCAI 2003*, pages 1173–1178. Morgan Kaufmann, 2003.