


Elimination Distance to Dominated Clusters

Nicole Schirrmacher ✉ 

University of Bremen, Germany

Sebastian Siebertz ✉ 

University of Bremen, Germany

Alexandre Vigny ✉ 

Université Clermont Auvergne, Clermont Auvergne INP, LIMOS, CNRS, Clermont-Ferrand, France

Abstract

In the DOMINATED CLUSTER DELETION problem, we are given an undirected graph G and integers k and d and the question is to decide whether there exists a set of at most k vertices whose removal results in a graph in which each connected component has a dominating set of size at most d . In the ELIMINATION DISTANCE TO DOMINATED CLUSTERS problem, we are again given an undirected graph G and integers k and d and the question is to decide whether we can *recursively* delete vertices up to depth k such that each remaining connected component has a dominating set of size at most d . Bentert et al. [Bentert et al., MFCS 2024] recently provided an almost complete classification of the parameterized complexity of DOMINATED CLUSTER DELETION with respect to the parameters k , d , c , and Δ , where c and Δ are the degeneracy, and the maximum degree of the input graph, respectively. In particular, they provided a non-uniform algorithm with running time $f(k, d) \cdot n^{\mathcal{O}(d)}$. They left as an open problem whether the problem is fixed-parameter tractable with respect to the parameter $k + d + c$. We provide a uniform algorithm running in time $f(k, d) \cdot n^{\mathcal{O}(d)}$ for both DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS. We furthermore show that both problems are FPT when parameterized by $k + d + \ell$, where ℓ is the semi-ladder index of the input graph, a parameter that is upper bounded and may be much smaller than the degeneracy c , positively answering the open question of Bentert et al. We further complete the picture by providing an almost full classification for the parameterized complexity and kernelization complexity of ELIMINATION DISTANCE TO DOMINATED CLUSTERS. The one difficult base case that remains open is whether TREEDEPTH (the case $d = 0$) is NP-hard on graphs of bounded maximum degree.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Fixed parameter tractability

Keywords and phrases Graph theory, Fixed-parameter algorithms, Dominated cluster, Elimination distance

Digital Object Identifier 10.4230/LIPIcs.MFCS.2025.90

Related Version *Full Version*: <https://doi.org/10.48550/arXiv.2504.21675> [34]

Funding This paper is a part of the ANR-DFG project *Unifying Theories for Multivariate Algorithms* (UTMA), which has received funding from the German Research Foundation (DFG) with grant agreement No 446200270, and benefited from state aid managed by the ANR under France 2030 referenced ANR-23-IACL-0006.

1 Introduction

Assume that \mathcal{Q} is a graph problem that is hard to solve on general graphs but \mathcal{C} is a class of graphs where \mathcal{Q} can be solved efficiently. In many cases, we can also solve \mathcal{Q} efficiently on instances that are *close* to the instances of \mathcal{C} , say, on graphs that belong to \mathcal{C} after the deletion of a small number of vertices. Guo et al. [25] formalized this concept under the name *distance from triviality*. For example, the size of a minimum vertex cover is the distance to the class of edgeless graphs and the size of a minimum feedback vertex set is



© Nicole Schirrmacher, Sebastian Siebertz, and Alexandre Vigny;
licensed under Creative Commons License CC-BY 4.0

50th International Symposium on Mathematical Foundations of Computer Science (MFCS 2025).

Editors: Paweł Gawrychowski, Filip Mazowiecki, and Michał Skrzypczak; Article No. 90; pp. 90:1–90:20

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the distance to the class of acyclic graphs. CLUSTER VERTEX DELETION is the problem to determine for a given graph and number k whether there exists a set of at most k vertices whose deletion results in a *cluster graph*, that is, a graph in which every connected component is a clique [5, 35]. Many generalizations and special cases of CLUSTER VERTEX DELETION have been studied. For example, one may require that each connected component in the resulting graph is an s -club (a graph of diameter at most s) [21, 28] or an s -plex (a graph in which each vertex has degree at least $n - s$) [26, 36]. Note that the special case where each clique in the solution graph has to be of size one is again the VERTEX COVER problem. Very recently, Bentert et al. [4] studied the following DOMINATED CLUSTER DELETION problem: Given a graph G and integer parameters k and d , does there exist a set S of at most k vertices such that each connected component of $G - S$ can be dominated by at most d vertices? Bentert et al. provided an almost complete classification of the parameterized complexity and kernelization complexity of DOMINATED CLUSTER DELETION with respect to the parameters k , d , c , and Δ , where c and Δ are the degeneracy and the maximum degree of the input graph, respectively. They proved the following results: DOMINATED CLUSTER DELETION is **para-NP**-hard for parameters $k + \Delta$ and $d + \Delta$, $W[2]$ -hard for parameter $k + d$ and admits a non-uniform algorithm with running time $f(k, d) \cdot n^{\mathcal{O}(d)}$, and is non-uniformly fixed-parameter tractable with respect to parameter $k + d + \Delta$. They left as an open problem whether the problem is fixed-parameter tractable with respect to the parameter $k + d + c$. They also showed that the problem does not admit a polynomial kernel even for $d = 1$ with respect to the parameter $k + c$, or with respect to the parameter $k + d + \Delta$, unless $\text{NP} \subseteq \text{coNP/poly}$.

A related measure is the *elimination distance* to a class \mathcal{C} , which measures the number of *recursive* deletions of vertices needed for a graph G to become a member of \mathcal{C} . Formally, a graph G has elimination distance 0 to \mathcal{C} if $G \in \mathcal{C}$, and otherwise elimination distance $k + 1$, if in every connected component of G , we can delete a vertex such that the resulting graph has elimination distance k to \mathcal{C} . Note that elimination distance to the class containing only the empty graph corresponds to the well-known measure *treedepth*. Elimination distance was introduced by Bulian and Dawar [10] in their study of the parameterized complexity of the graph isomorphism problem. Just as distance to triviality, but with more general applicability, elimination distance allows to lift algorithmic results from a base class to more general graph classes. For example, Bulian and Dawar [10] provided an FPT algorithm for the graph isomorphism problem parameterized by the elimination distance to the class \mathcal{C}_d of graphs with maximum degree bounded by d , for any fixed integer d . Hols et al. [27] proved the existence of polynomial kernels for the vertex cover problem parameterized by the size of a deletion set to graphs of bounded elimination distance to different classes of graphs. Agrawal et al. [2] proved fixed-parameter tractability of ELIMINATION DISTANCE TO \mathcal{C} for any class \mathcal{C} that can be defined by a finite set of forbidden induced subgraphs. Agrawal and Ramanujan studied the ELIMINATION DISTANCE TO CLUSTER GRAPHS [3]. Fomin, Golovach and Thilikos further generalized several of these prior results by considering elimination distances to graph classes expressible by restricted first-order formulas [22]. These results are also implied by the recent algorithmic meta theorem for separator logic [31, 33]. The related concept of recursive backdoors has recently been studied in the context of efficient SAT solving [18, 30]. To the best of our knowledge, we are the first to study elimination distance to classes with small dominating sets.

In this work, we answer the open questions of Bentert et al. [4] for DOMINATED CLUSTER DELETION and furthermore provide an almost full classification of the parameterized complexity and kernelization complexity of ELIMINATION DISTANCE TO DOMINATED CLUSTERS.

More precisely, we prove that DOMINATED CLUSTER DELETION admits a uniform algorithm with running time $f(k, d) \cdot n^{\mathcal{O}(d)}$ as well as a uniform fixed-parameter algorithm with running time $f(k, d, \ell) \cdot n^{\mathcal{O}(1)}$, where ℓ is the semi-ladder index of the input graph. The semi-ladder index was introduced by Fabiański et al. [20] in the study of domination-type problems and is a parameter that is upper bounded and may be much smaller than the degeneracy c of a graph. In fact, our result holds for all classes of graphs where the related PARTIAL DOMINATION problem is fixed-parameter tractable and satisfies an additional technical condition (the concept of PARTIAL DOMINATION and the details will be discussed in a moment). In particular, our result answers the question of Bentert et al. positively.

Our main contributions are the uniform FPT algorithms for DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS on semi-ladder-free graphs.

► **Theorem 1.** *DOMINATED CLUSTER DELETION can be solved in time $f(k, d) \cdot n^{\mathcal{O}(d)}$ and in time $f(k, d, \ell) \cdot n^{\mathcal{O}(1)}$ for a computable function f , where ℓ is the semi-ladder index of the input graph.*

► **Theorem 2.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS can be solved in time $f(k, d) \cdot n^{\mathcal{O}(d)}$ and $f(k, d, \ell) \cdot n^{\mathcal{O}(1)}$ for a computable function f , where ℓ is the semi-ladder index of the input graph.*

The hardness results completing the classification are simple observations. We prove the following. ELIMINATION DISTANCE TO DOMINATED CLUSTERS is

1. para-NP-hard for parameter $k + \Delta$; this is a simple consequence of the fact that the case $k = 0$ corresponds to the DOMINATING SET problem, which is known to be NP-hard on graphs of maximum degree 3 [23], and
2. para-NP-hard for parameter d ; this is a simple consequence of the fact that the case $d = 0$ corresponds to the TREEDEPTH problem, which is known to be NP-hard [32], and
3. W[2]-hard for parameter $k + d$; this is again a simple consequence of the fact that the case $k = 0$ corresponds to DOMINATING SET, which is W[2]-hard with respect to parameter d [15], and
4. uniformly fixed-parameter tractable for parameter $k + d + \ell$. This is our main contribution, we discuss the details below, and
5. uniformly fixed-parameter tractable with respect to k when d is considered to be a constant, that is, admits a uniform algorithm with running time $f(k, d) \cdot n^{\mathcal{O}(d)}$.
6. The problem does not admit a polynomial kernel with respect to parameter k even for fixed parameters $d = 0$ and $c = 2$ unless $\text{NP} \subseteq \text{coNP/poly}$; The parameter $d = 0$ corresponds to TREEDEPTH. We prove the simple observation that TREEDEPTH is NP-complete on 2-degenerate graphs. It is then trivial using an AND-cross-composition to prove that TREEDEPTH on 2-degenerate graphs does not admit a polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$.

The one case that we could not resolve is the parameter $d + \Delta$. The case $d = 0$ corresponds to TREEDEPTH, but we could not resolve whether the problem is NP-hard on graphs of bounded maximum degree. We state the following two conjectures.

► **Conjecture 1.** *TREEDEPTH is NP-hard on some class of graphs with bounded maximum degree.*

► **Conjecture 2.** *TREEDEPTH is NP-hard on planar graphs.*

The latter conjecture is related to the long-standing open question about the complexity of TREEWIDTH on planar graphs. On the other hand, it is known since 1997 that TREEWIDTH is NP-hard on graphs with maximum degree 9 [7] and more recently on graphs with maximum degree 3 [6].

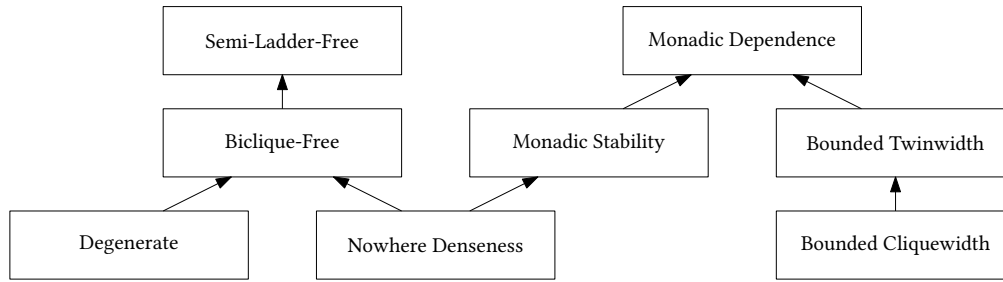
1.1 Techniques

Our main contribution is the uniform algorithm with respect to the parameter $k + d + \ell$ (the algorithm for $k + d$ is a simple modification of this algorithm).

We follow the approach of Bentert et al. [4] to reduce to unbreakable graphs, that is, graphs that cannot be separated into multiple large connected components by small separators. Let us give some formal definitions. A *separation* in a graph G is a pair of vertex subsets $A, B \subseteq V(G)$ such that $A \cup B = V(G)$ and there is no edge with one endpoint in $A \setminus B$ and the other in $B \setminus A$. The order of the separation (A, B) is the size of its *separator* $A \cap B$. For $q, k \in \mathbb{N}$, a set $X \subseteq V(G)$ is (q, k) -*unbreakable* if for every separation (A, B) of G of order at most k , we have $|A \cap X| \leq q$ or $|B \cap X| \leq q$. Intuitively speaking, X is (q, k) -unbreakable if no separation of order k can break X in a balanced way: one of the sides must contain at most q vertices of X . For example, cliques and $k + 1$ -connected graphs are (k, k) -unbreakable, and square grids are $(\mathcal{O}(k^2), k)$ -unbreakable.

Lokshtanov et al. [29] proved a powerful meta theorem for logically defined graph properties. Whenever a CMSO-property (monadic second-order logic with modulo counting predicates) can be solved efficiently on unbreakable graphs (for appropriately chosen parameters k and q), then it can also be decided efficiently on general graphs. There is a small caveat though. The meta theorem uses the non-constructive recursive understanding technique, and hence, one only obtains an efficient non-uniform algorithm on general graphs. Both DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS are CMSO-properties, and hence fall into this framework. Bentert et al. rely on this framework and show how to solve DOMINATED CLUSTER DELETION on unbreakable graphs in time $f(k, d) \cdot n^{\mathcal{O}(d)}$. The key combinatorial observation is that in (q, k) -unbreakable graphs, the deletion of at most k vertices leads to one giant component and a bounded number of remaining small components. One can then guess a dominating set for the giant component in time $\mathcal{O}(n^d)$ and argue combinatorially to efficiently find the vertices to be deleted. Using the result of Lokshtanov et al. one obtains the non-uniform algorithm for general graphs with the desired running time.

We remark that DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS are in fact first-order definable problems, that is, for all parameters k, d , there exist first-order formulas of length $\mathcal{O}_{k,d}(1)$ defining the problems. Hence, the problems are fixed-parameter tractable on all classes on which first-order model checking is fixed-parameter tractable, e.g. on nowhere dense classes [24] and even monadically stable graph classes [16, 17], as well as on classes with bounded cliquewidth [11] (and classes with bounded twinwidth [9] when contraction sequences are given with the input). Note that first-order logic cannot define connected components in general. However, the connected components arising in the recursive deletion of elements must have small diameter when they can be dominated by few vertices. Hence, connected components in positive instances of DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS become first-order definable. Thus, on these classes one obtains algorithms with running time $f(k, d) \cdot n^{\mathcal{O}(1)}$ by the meta theorems for first-order logic. It is conjectured that monadically dependent classes are the most general hereditary graph classes that admit efficient first-order model checking, we hence included them in the inclusion diagram, see Figure 1.



■ **Figure 1** Relation between the discussed graph classes. Arrows indicate inclusion. DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS are fixed-parameter tractable on monadically stable classes by a meta theorem for first-order logic. The same is true for bounded twinwidth classes when contraction sequences are given with the input. Our main result is the fixed-parameter tractability on semi-ladder-free graphs.

This leaves two main questions. First, can we turn the non-uniform algorithm of Bentert et al. running in time $f(k, d) \cdot n^{\mathcal{O}(d)}$ into a uniform algorithm? Second, can we improve the running time on further restricted graph classes? We answer both questions positively.

As mentioned, we follow the approach of Bentert et al. [4] to reduce to unbreakable graphs. We first observe that on unbreakable graphs, the notion of elimination distance almost collapses to distance to triviality, enabling us to treat DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS in a similar way. This idea has already been examined for hereditary classes in several papers see e.g. [1]. Note that being a dominated cluster is not a hereditary notion. In order to avoid the non-uniform approach of Lokshtanov et al. [29], we use a decomposition theorem due to Cygan et al. [13]. The theorem states that every graph admits a tree decomposition with small adhesion such that each bag is unbreakable in the subgraph induced by the bags below. We show that if an annotated version of the PARTIAL DOMINATING SET problem can be solved in a slightly modified graph (that we call the bag graph), then we can implement an efficient dynamic programming algorithm along the tree decomposition to the original input graph for both DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS.

The (ANNOTATED) PARTIAL DOMINATING SET problem is the following problem. We are given a graph with three sets F , R and B and parameters d and k . We ask whether there exist k vertices of $G - F$ (F stands for forbidden) that may be deleted such that the remaining graph has a Red-Blue dominating set of size at most d , i.e. the vertices of R can be dominated by d vertices of B (R and B stand for Red and Blue). Formally, the problem is defined as follows. Given a graph G with (not necessarily disjoint) sets $F \subseteq V(G)$, $R \subseteq V(G)$, and $B \subseteq V(G)$ and parameters d and k , does there exist a set $X \subseteq V(G) - F$ of size at most k and a set $D \subseteq B - X$ of size at most d such that $G[R - X]$ is dominated by D ? The problem can obviously be solved in time $n^{\mathcal{O}(d)}$ (guess the dominating set and check if at most k Red vertices remain undominated). By our dynamic programming approach, we hence get a uniform algorithm running in time $f(k, d) \cdot n^{\mathcal{O}(d)}$ in general graphs. We then turn our attention to restricted graph classes where (ANNOTATED) PARTIAL DOMINATION can be solved more efficiently. Using an approach of Fabiański et al. [20], we observe that (ANNOTATED) PARTIAL DOMINATION can be solved efficiently on all *semi-ladder-free* graph classes, which yields our second main result.

Semi-ladder-free are very general graph classes. For example, all degenerate classes are biclique-free (that is, exclude a fixed complete bipartite graph $K_{t,t}$ as a subgraph), which in turn are semi-ladder-free, hence, our result in particular answers the question of Bentert et al.

positively. On the other hand, these classes are incomparable with monadically stable classes and classes with bounded twinwidth on which we can efficiently solve the first-order model checking problem. For example, the class of all ladders has bounded twinwidth (in fact even bounded linear cliquewidth) but is not semi-ladder-free. The class of all co-matchings is monadically stable (and has bounded linear cliquewidth as well) and is not semi-ladder-free. On the other hand, the class of all 1-subdivided cliques is semi-ladder-free (and even 2-degenerate) but is neither monadically stable nor has bounded twinwidth.

As mentioned, we require that (ANNOTATED) PARTIAL DOMINATION needs to be solved on a slightly modified graph class (on the bag graphs), however, the property of being semi-ladder-free is preserved by this modification, so that ultimately we conclude our main results, Theorem 1 and Theorem 2.

Organization. The paper is organized as follows. After recalling the necessary notation in Section 2, we talk more precisely about partial domination and the graph parameters enabling FPT evaluation in Section 3. We then present how to solve DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS on unbreakable graph with bounded semi-ladder index in Section 4. After that, in Section 5, we sketch the uniform algorithm on general semi-ladder-free graphs for DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS with running time $f(k, d) \cdot n^{\mathcal{O}(d)}$. The details can be found in the full version [34]. The hardness results conclude the paper in Section 6.

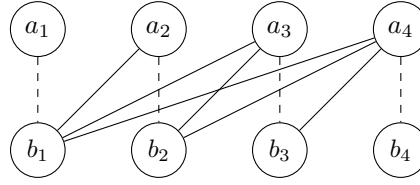
2 Preliminaries

Graphs. We consider finite, undirected graphs without loops. Our notation is standard, and we refer to Diestel's textbook for more background on graphs [14]. We write $|G|$ for the size of the vertex set $V(G)$ and $\|G\|$ for $|V(G)| + |E(G)|$, where $E(G)$ is the edge set of a graph G . A connected component of G is a maximal connected subgraph of G . For a vertex subset $X \subseteq V(G)$, we write $G[X]$ for the subgraph of G induced by X . We write $G - X$ for $G[V(G) \setminus X]$ and for singleton sets $\{v\}$ we write $G - v$ instead of $G - \{v\}$. We denote the open neighborhood of a vertex v by $N(v)$ and the closed neighborhood (including the vertex v) by $N[v]$. For a set $Y \subseteq V(G)$, we write $N[Y]$ for $\bigcup_{v \in Y} N[v]$.

Semi-ladders. Let G be a graph. Two sequences, $a_1, \dots, a_n \in V(G)$ and $b_1, \dots, b_n \in V(G)$ of $2n$ distinct vertices, form a *semi-ladder* of order n in G if $\{a_i, b_j\} \in E(G)$ for all $i, j \in \{1, \dots, n\}$ with $i > j$, and $\{a_i, b_i\} \notin E(G)$ for all $i \in \{1, \dots, n\}$. Note that we do not impose any condition for $i < j$. The *semi-ladder index* of a graph is the maximum order of a semi-ladder-free it contains. A class of graphs has *bounded semi-ladder index* or is called *semi-ladder-free* if there is some ℓ such that the semi-ladder index of all of its members is bounded by ℓ , see Figure 2.

Tree decompositions. A *tree decomposition* of a graph G is a pair $\mathcal{T} = (T, \text{bag})$, where T is a rooted tree and $\text{bag}: V(T) \rightarrow 2^{V(G)}$ is a mapping that assigns to each node $x \in T$ its bag $\text{bag}(x) \subseteq V(G)$ such that the following conditions are satisfied:

- For every vertex $v \in V(G)$, there exists a node $x \in V(T)$ such that $v \in \text{bag}(x)$.
- For every vertex $v \in V(G)$, the set of nodes $x \in V(T)$ satisfying $v \in \text{bag}(x)$ induces a connected subtree of T .
- For every edge $\{u, v\} \in E(G)$, there exists a node $x \in V(T)$ such that $u, v \in \text{bag}(x)$.



■ **Figure 2** Semi-ladder of order 4. Solid lines represent edges while dashed edges represent non-edges.

Let $\mathcal{T} = (T, \text{bag})$ be a tree decomposition of a graph G . The *adhesion* of a node $x \in V(T)$ is defined as $\text{adh}(x) := \text{bag}(\text{parent}(x)) \cap \text{bag}(x)$ and the *margin* of a node $x \in V(T)$ is defined as $\text{mrg}(x) := \text{bag}(x) \setminus \text{adh}(x)$. The *cone* at a node $x \in V(T)$ is defined as $\text{cone}(x) := \bigcup_{y \succ_T x} \text{bag}(y)$ and the *component* at a node $x \in V(T)$ is defined as $\text{comp}(x) := \text{cone}(x) \setminus \text{adh}(x) = \bigcup_{y \succ_T x} \text{mrg}(y)$. Here, $y \succ_T x$ means that y is a descendant of x in T .

Unbreakable graphs. A *separation* in a graph G is a pair of vertex subsets $A, B \subseteq V(G)$ such that $A \cup B = V(G)$ and there is no edge with one endpoint in $A \setminus B$ and the other in $B \setminus A$. The order of the separation (A, B) is the size of its *separator* $A \cap B$.

For $q, k \in \mathbb{N}$, a vertex subset X in a graph G is (q, k) -*unbreakable* if for every separation (A, B) of G of order at most k , we have $|A \cap X| \leq q$ or $|B \cap X| \leq q$. Not every graph is (q, k) -unbreakable, but all graphs admit tree decompositions with small adhesion and unbreakable parts. For fixed $q, k \in \mathbb{N}$, a tree decomposition (T, bag) of a graph G is (q, k) -unbreakable if for every node $x \in V(T)$, the bag $\text{bag}(x)$ is (q, k) -unbreakable in $G[\text{cone}(x)]$. By the result of Cygan et al. [13], such tree decompositions exist and can be computed efficiently.

► **Theorem 3** (Theorem 10 of [13]). *Let G be a graph and k an integer. There exists an integer $q \in 2^{\mathcal{O}(k^2)}$ and an algorithm that computes in time $q \cdot |G|^2 \|G\|$ a tree decomposition (T, bag) of G with at most $|G|$ nodes such that the following conditions hold:*

- *For every node $x \in V(T)$, the bag $\text{bag}(x)$ is (q, k) -unbreakable in the subgraph $G[\text{cone}(x)]$.*
- *For every node $x \in V(T)$, the adhesion $\text{adh}(x)$ is of size at most q .*

A tree decomposition (T, bag) is *regular* if for every non-root node $x \in V(T)$,

- the margin $\text{mrg}(x)$ is non-empty,
- the graph $G[\text{comp}(x)]$ is connected, and
- every vertex of $\text{adh}(x)$ has a neighbor in $\text{comp}(x)$.

We may further assume that the tree decomposition we get from Theorem 3 is regular, see the construction of Lemma 2.8 of [8] and the discussion in [31].

Parameterized complexity. A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is *fixed-parameter tractable (FPT)* if there exists an algorithm that decides for an instance (x, k) whether $(x, k) \in L$ in time $f(k) \cdot |x|^{\mathcal{O}(1)}$ for some computable function f .

3 Partial domination and FPT evaluation

We show how to efficiently solve the ANNOTATED PARTIAL DOMINATION problem on semi-ladder-free graphs. This problem falls into the framework of domination-type problems on semi-ladder-free graphs introduced by Fabiański et al. [20].

We are going to use the following meta theorem of Fabiański et al. [20]. A *domination-type problem* is a formula $\exists \bar{x} \forall \bar{y} \delta(\bar{x}, \bar{y})$, where δ is a positive first-order formula using fixed distances as predicates (called a *distance formula* in [20]). On every graph G , a formula $\delta(\bar{x}, \bar{y})$ defines a bipartite graph G_δ on the vertex set $V(G)^{|\bar{x}|} \times V(G)^{|\bar{y}|}$, where two elements \bar{u} and \bar{v} are connected by an edge if the formula $\delta(\bar{u}, \bar{v})$ is true in G . The semi-ladder index of δ on G is defined as the semi-ladder index of G_δ .

► **Theorem 4** (Theorem 19 of [20]). *Let \mathcal{C} be a class of graphs and let δ be a distance formula. Assume that G_δ has bounded semi-ladder index ℓ for some fixed $\ell \in \mathbb{N}$ on all graphs $G \in \mathcal{C}$. Then there is an algorithm that solves the domination-type problem δ on graphs G from \mathcal{C} in time $f(\ell, |\varphi|) \cdot \|G\|$.*

ANNOTATED PARTIAL DOMINATION can be written with the following first-order formula:

$$\varphi_{k,d} = \exists x_1 \dots \exists x_k \exists y_1 \dots \exists y_d \forall z \left(\bigwedge_{1 \leq i \leq k} \neg F(x_i) \right) \wedge \left(\bigwedge_{1 \leq i \leq d} B(y_i) \right) \wedge \left(R(z) \rightarrow \left(\bigvee_{1 \leq i \leq k} \text{dist}(z, x_i) = 0 \vee \bigvee_{1 \leq i \leq d} \text{dist}(z, y_i) \leq 1 \right) \right)$$

However, φ is not a distance formula as it uses negations. This can be circumvented by inverting the predicates, i.e. $F' = V(G) - F$ and $R' = V(G) - R$. We then obtain the following formula

$$\delta_{k,d} = \exists x_1 \dots \exists x_k \exists y_1 \dots \exists y_d \forall z \left(\bigwedge_{1 \leq i \leq k} F'(x_i) \right) \wedge \left(\bigwedge_{1 \leq i \leq d} B(y_i) \right) \wedge \left(R'(z) \vee \bigvee_{1 \leq i \leq k} \text{dist}(z, x_i) = 0 \vee \bigvee_{1 \leq i \leq d} \text{dist}(z, y_i) \leq 1 \right)$$

We prove next that $\delta_{d,k}$ has bounded semi-ladder index on every graph G with bounded semi-ladder index.

► **Lemma 5.** *Let G be a graph with semi-ladder index ℓ . Then $\delta_{d,k}$ has semi-ladder index at most $f(d, k, \ell)$ on G , for some computable function f .*

Proof. Lemma 4 of [20] states that if $\varphi_1(\bar{x}; \bar{y}), \dots, \varphi_k(\bar{x}; \bar{y})$ are formulas and $\psi(\bar{x}; \bar{y})$ is a positive boolean combination of $\varphi_1, \dots, \varphi_k$, and G is a graph such that $\varphi_1(G), \dots, \varphi_k(G)$ have bounded semi-ladder index, then also $\psi(G)$ has bounded semi-ladder index. This is the case for the above formula $\delta_{k,d}$. By Lemma 5 of [20], the formula $\bigvee_{1 \leq i \leq d} \text{dist}(z, y_i) \leq 1$ has bounded semi-ladder index on every graph with bounded semi-ladder index. All other formulas trivially have bounded semi-ladder index on all graphs. ◀

► **Corollary 6.** *Let \mathcal{C} be a class of graphs with bounded semi-ladder index ℓ . Then ANNOTATED PARTIAL DOMINATION can be solved in time $f(d, k, \ell) \cdot \|G\|$ for all $G \in \mathcal{C}$ for a computable function f .*

While the semi-ladder index is currently one of the most general graph parameters enabling FPT evaluation of $\delta_{k,d}$, some other parameter could have this property.

► **Property 1.** *A graph parameter $t(\cdot)$ has Property 1 if for every graph G ANNOTATED PARTIAL DOMINATION can be solved in time $f(d, k, t(G)) \cdot |G|^{\mathcal{O}(1)}$ for a computable function f .*

As previously discussed, examples of parameters with Property 1 include treewidth, maximum degree, Hadwiger number (i.e. largest clique minor), degeneracy, largest excluded bi-clique, as they all imply bounded semi-ladder index. Other parameters, which are however not as interesting, as the problem can be solved by efficient FO model checking, are cliquewidth and twinwidth (when contraction sequences are given with the input).

As mentioned, we want to solve ANNOTATED PARTIAL DOMINATION on slightly modified graphs. We therefore want a parameter that, if bounded on the graph, bounds a parameter with Property 1 on **bgraph** which is a similar notion as bag graphs (see Definition 27) that is only defined in the full version of the paper [34].

► **Property 2.** *A graph parameter $t_2(\cdot)$ has Property 2 if there is a parameter $t_1(\cdot)$ satisfying Property 1 such that for every graph G , and integers q, k, d , every (q, k) -unbreakable tree decomposition (T, bag) , vertex $x \in T$, and set $S \subseteq \text{cone}(x)$ we have that $t_1(\text{bgraph}_S(x)) \leq f(q, k, d, t_2(G))$.*

In the rest of the paper, we precisely reference parameters having Property 1 or Property 2. In particular, the semi-ladder index also satisfies Property 2. The proof can be found in the full version [34].

4 Algorithm on unbreakable graphs based on Annotated Partial Domination

As sketched above, we first solve DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS on unbreakable graphs. Our algorithm runs efficiently on all graph classes for which we can efficiently solve the ANNOTATED PARTIAL DOMINATION problem, that is, for classes on which a parameter $t(\cdot)$ satisfying Property 1 is bounded.

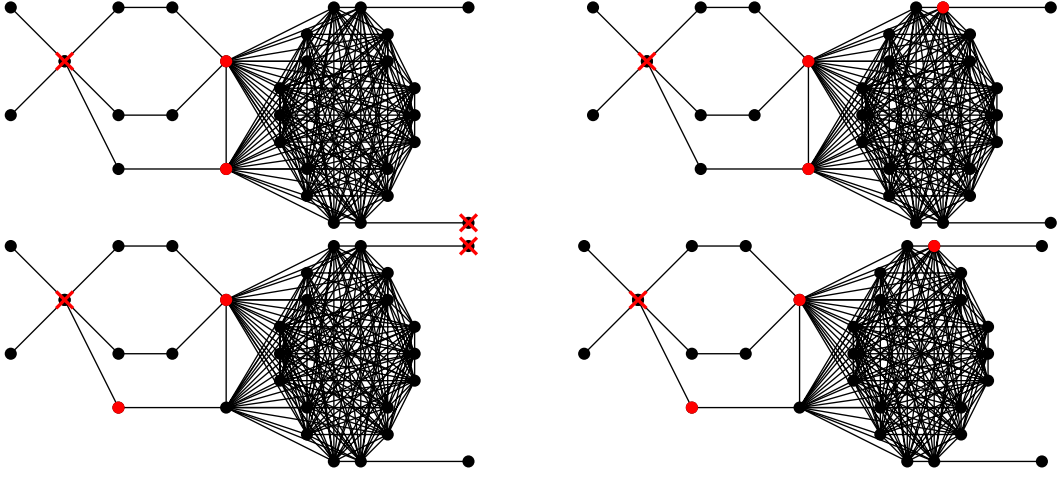
In the following, q, k , and d will always be non-negative integers, and we will not always quantify them for readability. We will also tacitly assume that our graphs have at least $2q + 1$ vertices. Then, whenever we delete a set S of at most k vertices there will be a unique connected component C_0 of $G - S$ with more than q vertices that we call the *large connected component*. When a graph has less vertices, we can solve all our problems by brute-force.

► **Lemma 7.** *Let G be a (q, k) -unbreakable graph, and let $v \in V(G)$. Then $G - v$ is $(q, k - 1)$ -unbreakable.*

Proof. Assume that $G - v$ is not $(q, k - 1)$ -unbreakable. Then, there is a separation (A, B) of $G - v$ such that $|A| \geq q$, $|B| \geq q$ and $|A \cap B| \leq k - 1$. By adding the vertex v to the separator, we obtain a separation (A', B') of G such that $|A'| > q$, $|B'| > q$ and $|A' \cap B'| \leq k$ contradicting that G is (q, k) -unbreakable. ◀

► **Lemma 8.** *Let G be a (q, k) -unbreakable graph. If G is a positive instance of DOMINATED CLUSTER DELETION with parameters k and d , then G has a dominating set of size at most $q + d$.*

Proof. Let S' with $|S'| \leq k$ be a solution for DOMINATED CLUSTER DELETION with parameters d and k . Let $A = S' \cup C_0$, where C_0 is the large connected component of $G - S'$ and $B = S' \cup \bigcup_{1 \leq j \leq m} C_j$, where C_1, \dots, C_m are the other connected components of $G - S'$. Then, (A, B) is a separation of order at most k , hence $|B| \leq q$, as G is (q, k) -unbreakable. As G is a positive instance of DOMINATED CLUSTER DELETION, C_0 can be dominated by a set D_0 of size at most d . Then $D_0 \cup B$ is dominating set of G of size at most $q + d$. ◀



■ **Figure 3** Different skeletons for the dominated cluster deletion problem for parameter $k = 4$ and $d = 1$ on a fixed $(14,4)$ -unbreakable graph. The skeleton S is in red, the witness S' also contains the crossed-out vertices.

4.1 Dominated Cluster Deletion

We first deal with DOMINATED CLUSTER DELETION. Our approach is based on finding *skeletons for dominated cluster deletion*, which are vertex sets that can be extended to the deletion set of a solution for DOMINATED CLUSTER DELETION and in (q, k) -unbreakable graphs capture the part of a solution that essentially separates the large connected component from the other connected components.

► **Definition 9.** Let G be a (q, k) -unbreakable graph. We call a set S a skeleton for dominated cluster deletion for parameters k and d if there is a superset $S' \supseteq S$ of vertices satisfying:

- $|S'| \leq k$,
- every connected component of $G - S'$ has domination number at most d , and
- S contains exactly the vertices of S' that have at least one neighbor in C'_0 and at least one neighbor in $G - N[C'_0]$, where C'_0 is the large connected component of $G - S'$.

Note that S may be the empty set in the third item above if $G - S'$ has only one connected component. Figure 3 shows examples of skeletons for dominated cluster deletion.

The next lemma is the key ingredient to compute the set of skeletons (and to show that there are only few possible skeletons). It shows that there is a small set of vertices that hits every skeleton.

► **Lemma 10.** Let q, k, d be integers, G a (q, k) -unbreakable graph, and assume that S is a non-empty skeleton for dominated cluster deletion for parameters k and d . Given a dominating set X of size at most $q + d$, there is a vertex $v \in S$ that is

1. in the dominating set X , or
2. in the neighborhood $N(x)$ of a vertex $x \in X$, where $\deg(x) \leq q$, or
3. in the neighborhood $N(y)$ of a vertex $y \in N(x)$, where $\deg(x) \leq q$, $\deg(y) \leq q$, and $x \in X$.

Proof. Assume that Case 1 is not satisfied so that no vertices from the skeleton are in the dominating set X . Denote by C'_0 the large connected component of $G - S'$ for a set $S' \supseteq S$ witnessing that S is a skeleton, hence, S contains exactly the vertices of S' that have a neighbor in C'_0 and a second neighbor in $G - N[C'_0]$. Let C_0 be the large connected component of $G - S$. Note that it contains C'_0 .

Let $A = S \cup C_0$ and $B = S \cup \bigcup_{1 \leq j \leq m} C_j$, where C_1, \dots, C_m are the other connected components of $G - S$. Then (A, B) is a separation of order at most $|S| \leq k$, hence $|B| \leq q$, as G is (q, k) -unbreakable. Pick an arbitrary element $v \in S$. By definition of S , v has a neighbor in C_0 and a neighbor w in a second connected component, say C_1 of $G - S$. Hence, w must be a vertex of degree at most q as its neighborhood lies in $G - C_0 \subseteq B$. If w is in X , then this vertex v satisfies Case 2. Otherwise, w must have a neighbor x in X (since X is a dominating set). Since we are not in Case 1, this x is not in S , and is therefore in C_1 . Hence, similarly to w , we have that x has degree at most q , and we are in Case 3. ◀

Lemma 10 gives rise to a bounded search tree branching algorithm that branches over the possible choices for a vertex from S given by the conditions of the lemma.

► **Definition 11.** *The skeleton-algorithm with parameter q, k, d over a graph G returns a family of sets of vertices, each set of size at most k . First, it computes a dominating set X of size at most $q + d$ (returning the empty family if there is no such set). Second, it computes the set $X \cup Y \cup Z$, where Y is the set of all vertices in the neighborhood $N(x)$ of a vertex $x \in X$ with $\deg(x) \leq q$, and the set Z is the set of all vertices in the neighborhood $N(y)$ of a vertex $y \in Y$ where $\deg(y) \leq q$. Third, call the algorithm with parameters $q, k - 1, d$ on $G - v$ for every v in $X \cup Y \cup Z$ (and get a family \mathcal{F}_v). Finally, for every v in $X \cup Y \cup Z$ add v to every set in \mathcal{F}_v , and return the empty set and the union of all the families \mathcal{F}_v .*

■ **Algorithm 1** Skeleton-algorithm.

```

1: procedure SKELETON( $G, q, k, d$ )
2:    $\mathcal{F} := \emptyset$ 
3:   if  $k = 0$  then
4:     return  $\mathcal{F}$ 
5:   end if
6:    $X := \text{DOMINATINGSET}(G, q + d)$ 
7:    $Y := \{N(x) : x \in X \wedge \deg(x) \leq q\}$ 
8:    $Z := \{N(y) : y \in Y \wedge \deg(y) \leq q\}$ 
9:   for all  $v \in X \cup Y \cup Z$  do
10:     $\mathcal{F} := \mathcal{F} \cup \{\emptyset\} \cup (\{v\} \times \text{SKELETON}(G - v, q, k - 1, d))$ 
11:   end for
12:   return  $\mathcal{F}$ 
13: end procedure

```

Note that in the above definition, if a family \mathcal{F}_v is empty, adding v to every set still results in an empty family (e.g. in the case where $G - v$ does not have a $q + d$ dominating set). Do not be confused with \mathcal{F}_v having the empty set as one of the family members (e.g. for branches of the algorithm that stops before reaching $k = 0$).

We now show that the skeleton algorithm 1) runs efficiently and outputs a small family of sets which is proved in Lemma 12, and 2) if the given graph was a positive instance to DOMINATED CLUSTER DELETION, then the skeleton algorithm outputs all possible skeletons; see Lemma 13.

► **Lemma 12.** *Let $t(\cdot)$ be a graph parameter satisfying Property 1, then the skeleton-algorithm runs in time $f(q, k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$, and outputs a family of at most $g(q, k, d)$ sets where f and g are computable functions.*

90:12 Elimination Distance to Dominated Clusters

Proof. Computing X, Y and Z takes time $f(q, k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$ thanks to the fact that $t(\cdot)$ satisfies Property 1. We then branch over $(q + d)q^2$ many choices for v , and the recursion has depth at most k . ◀

► **Lemma 13.** *If G is a (q, k) -unbreakable graph that is a positive instance to DOMINATED CLUSTER DELETION with parameters (k, d) , then the skeleton-algorithm outputs a family that contains every skeleton.*

Proof. By induction on k . If $k = 0$, we simply output the empty set. Otherwise, by Lemma 8 there is such a set X , and by Lemma 10 at least one element v of X is in S . For the correct choices of v , $G - v$ is a positive instance for DOMINATED CLUSTER DELETION with parameters $k - 1, d$, and by Lemma 7, $G - v$ is $(q, k - 1)$ -unbreakable, so the recursive call to SKELETON produces the expected sets by induction. ◀

It remains to verify that S can indeed be enlarged to a set S' of size at most k with the properties.

► **Lemma 14.** *Given a (q, k) -unbreakable graph G and a set S of at most k vertices, we can test in time $f(q, k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$, whether S is a skeleton for DOMINATED CLUSTER DELETION with parameters k and d assuming $t(\cdot)$ satisfies Property 1.*

Proof. We need to check the existence of a set $S' \supseteq S$ with $|S'| \leq k$ such that:

- every connected component of $G - S'$ has domination number at most d , and
- S contains exactly the vertices of S' that have a neighbor in C'_0 and a second neighbor in $G - N[C'_0]$ where C'_0 is the large connected component of $G - S'$.

Note that by these conditions the large connected component C_0 of $G - S$ is separated from the small connected components by S . Let $s = |S|$, what remains to be done is hence the following.

1. Find an optimal solution for $G - S - C_0$. This can be done in time $f(k, d)$, because this subgraph has size at most q and yields a value k' of vertices that need to be deleted. If $k' > k - s$, we conclude that this S is not a skeleton because there is no possible S' .
2. Test whether an additional set W of size $k'' = k - s - k'$ can be deleted from C_0 such that $G[C_0 - W]$ can be dominated by d vertices. Note that by assumption on S and S' the deletion of W will not cause C_0 to break into multiple further connected components. Hence, we simply check whether C_0 is a positive instance of ANNOTATED PARTIAL DOMINATION with parameter k'' , where $R = B = C_0$, and $F = \emptyset$. If S is a skeleton, such a set exists (as $S' \cap C_0$ is a candidate) and hence, if G is a positive instance of DOMINATED CLUSTER DELETION a positive solution will be found. Since $t(\cdot)$ is a graph parameter satisfying Property 1, we can test the existence of such W in time $f(k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$. If no solution is found, then this S is not a skeleton. ◀

The combination of Lemmas 10 and 12–14 yields the following theorem.

► **Theorem 15.** *DOMINATED CLUSTER DELETION on (q, k) -unbreakable graphs can be solved in time $f(q, k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$, where $t(\cdot)$ is a parameter satisfying Property 1.*

4.2 Elimination Distance to Dominated Clusters

We now show how to adapt the algorithm for ELIMINATION DISTANCE TO DOMINATED CLUSTERS. The approach is very similar. Of course, the notion of skeletons has to be adapted, however, they behave very similarly on unbreakable graphs.

First, let us properly define ELIMINATION DISTANCE TO DOMINATED CLUSTERS.

► **Definition 16.** A graph G has elimination distance k to a class \mathcal{C} if:

- G is not connected and every connected component of G has elimination distance k to \mathcal{C} , or
- G is connected and there is a vertex v such that $G - v$ has elimination distance $k - 1$ to \mathcal{C} , or
- $G \in \mathcal{C}$ and $k = 0$.

In our case we measure, for every d , the elimination distance to the class \mathcal{C}_d of d -dominated clusters, i.e. graphs where every connected component has a d -dominating set. We interchangeably write “elimination distance k to d -dominated cluster”, and “a positive instance for (k, d) -ELIMINATION DISTANCE TO DOMINATED CLUSTERS”. For algorithmic purposes, the above definition is not the most useful. We will instead look at the set of vertices that needs to be deleted to get d -dominated clusters. While this set is of unbounded size, it has a nice tree structure that is much more useful for our algorithms. This characterization is pretty standard, see the definition of *elimination order* [10, Proposition 4.3]

For a rooted tree T , we write \preceq_T for the ancestor relation, that is, $x \preceq_T y$ if x lies on the unique path from y to the root of T . Note that we treat every node as an ancestor of itself. The least common ancestor of two nodes $y, z \in V(T)$ is the \preceq_T -maximal element x with $x \preceq_T y$ and $x \preceq_T z$.

► **Definition 17.** Let G be a graph and $S \subseteq V(G)$. We say that S is tree-structured if there is a rooted tree T and a bijection $\lambda : S \rightarrow V(T)$ such that every path in G between two vertices u, v of S must contain a common ancestor in T of $\lambda(u)$ and $\lambda(v)$. The tree T is called an *elimination tree* for S . The *elimination depth* of a tree-structured set with respect to the tree T is the depth of T . The *elimination depth* of S is the minimum elimination depth of S over all elimination trees for S .

► **Remark 18.** A graph G is a positive instance for (k, d) -ELIMINATION DISTANCE TO DOMINATED CLUSTERS if and only if there is a tree-structured set S with elimination depth k such that every connected component of $G - S$ has a d -dominating set.

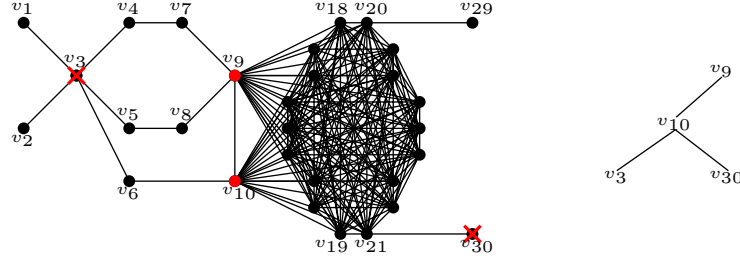
► **Lemma 19.** Let G be a (q, k) -unbreakable graph with at least $3q(k + q)$ vertices. Let S be a tree-structured set with an elimination tree T of depth at most k . Then

1. the largest connected component C_0 of $G - S$ is uniquely determined, and
2. for every component C of $G - S$ there is a subset $S(C) \subseteq S$ of size at most k that separates C from the rest, that is, $G - S(C)$ has C as one of its components.

Proof. Let C be a connected component of $G - S$. As S is tree-structured by T , C is not adjacent to two \preceq_T -incomparable elements, that is, the set $S(C)$ of neighbors of C in S consists of \preceq_T -comparable elements. C is a connected component both of $G - S$ and of $G - S(C)$, as all neighbors of C in S are collected in $S(C)$.

Assume now that there are two connected components C_0, C_1 of $G - S$ of size greater than q . However, C_0 and C_1 are separated by $S(C_0)$ of size at most k , which contradicts the (q, k) -unbreakability of G . So there can only be one connected component of size greater than q .

We now show that there must be at least one such component. First note that there cannot be more than $3q$ connected component in $G - S$. Otherwise, there would be a balance separator of T of size 1, by deleting this vertex and its ancestors, we delete at most k vertices and obtain a separation with both sides containing at least q components of $G - S$, contradicting the unbreakability of G . Last, as there are only $3q$ components C and $|S(C)| \leq k$ we have that $|S| \leq 3qk$ hence $|G - S| \geq 3q^2$ so one of the component must contain at least q vertices. ◀



■ **Figure 4** Example of a skeleton for elimination distance to dominated clusters for parameter $k = 3$ and $d = 1$ on a $(12,3)$ -unbreakable graph (left). The skeleton S is in red, the witness S' also contains the crossed-out vertices. A tree structure for S' of elimination depth 3 is on the right.

► **Lemma 20** (Analog of Lemma 8). *Let G be a (q, k) -unbreakable graph. If G is a positive instance of ELIMINATION DISTANCE TO DOMINATED CLUSTERS with parameters k and d , then G has a dominating set of size at most $q + d$.*

The proof can be taken verbatim from Lemma 8 with S' replaced by $S'(C_0)$.

► **Definition 21.** *Let G be a (q, k) -unbreakable graph. We call a set S a skeleton for elimination distance to dominated clusters for parameters k and d if there is a tree-structured superset $S' \supseteq S$ of elimination depth at most k satisfying:*

- *every connected component of $G - S'$ has domination number at most d , and*
- *S contains exactly the vertices of S' that have a neighbor in C'_0 and a second neighbor in $G - N[C'_0]$ where C'_0 is the large connected component of $G - S'$ (the next lemma shows that this connected component is uniquely determined even though S' can be larger than k).*

The following observation follows immediately from Lemma 19.

► **Observation 22.** *Every skeleton has size at most k .*

Now the analog of Lemma 10 also holds for skeletons for elimination distance to dominated clusters. The proof can be taken verbatim with S' replaced by $S'(C_0)$.

► **Lemma 23.** *Let q, k, d be integers, G be a (q, k) -unbreakable graph and assume that S is a non-empty skeleton for ELIMINATION DISTANCE TO DOMINATED CLUSTERS with parameters k and d . Given a domination set X of size $q + d$, there is a vertex $v \in S$ that is*

1. *in the dominating set X , or*
2. *in the neighborhood $N(x)$ of a vertex $x \in X$ where $\deg(x) \leq q$, or*
3. *in the neighborhood $N(y)$ of a vertex $y \in N(x)$ with $\deg(x) \leq q$, $\deg(y) \leq q$, and $x \in X$.*

Hence, the algorithm to guess a skeleton S can be carried out in exactly the same way as for DOMINATED CLUSTER DELETION. We then guess a tree structure on S (which takes time at most $\mathcal{O}(k!)$). Note that these vertices being adjacent to the large connected component, the tree structure is a linear order. In the final step, when we aim to extend a fixed skeleton S to a tree-structured solution S' , the only difference is that we have to solve the base problem ELIMINATION DISTANCE TO DOMINATED CLUSTERS for the small part $G - S - C'_0$, which again can be simply solved by brute-force. Note that because $G - S - C'_0$ is small, we can also find an elimination tree T for S' and embed S according to the originally guessed order. We conclude the main theorem of this section. The full algorithm for the annotated version is presented in the full version [34].

► **Theorem 24.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS on (q, k) -unbreakable graphs can be solved in time $f(q, k, d, t(G)) \cdot |G|^{\mathcal{O}(1)}$, where $t(\cdot)$ is a parameter satisfying Property 1.*

4.3 Skeletons for general graphs

In this section, we show that by parameterizing only by q, k , and d , we can compute the skeleton of unbreakable graphs in time $f(q, k, d) \cdot |G|^{\mathcal{O}(d)}$. This construction follows the work of Bentert et al. [4]. Having access to this algorithm will then be useful for our uniform version of their result.

► **Lemma 25.** *There is an algorithm that, given a (q, k) -unbreakable graph G , computes all skeletons S for dominated cluster deletion and for elimination distance to dominated clusters for parameters k and d in time $f(q, k, d) \cdot n^{\mathcal{O}(d)}$.*

This lemma resembles and can be proved analogously to [4, Lemma 7].

Proof. Assume the existence of a solution S' and let S be its skeleton. We guess a set D of at most d vertices in time $O(n^d)$. (This will play the role of the dominating set for the big component). We then define $S_1 = G - N[D]$ and $S_2 = \{v \in N(u) \mid u \in S_1 \wedge \deg(u) \leq q\}$. If either $|S_1| > q$ or $|S_2| > q + kq$, we conclude that this D is not a good candidate for the dominating set of the large connected component and terminate this branch. Otherwise, we now claim that $S \subseteq S_2$.

To this end, let C'_0 be the large connected component of $G - S'$ and assume that the algorithm guessed D as a dominating set for C'_0 . Observe that $C'_0 \subseteq N[D] \subseteq N[C'_0]$. A vertex of $G - N[C'_0]$ has no neighbor in C'_0 , and therefore only has degree at most q . Finally, remember that any vertex v of S must have a neighbor u that is not in $N[C'_0]$ and hence not in $N[D]$. So u is in S_1 and u has degree at most q . So v belongs to S_2 . Regarding the cardinality constraints. Note that at most q vertices are in $G - C'_0$ and hence at most q are in $G - N[D] = S_1$. Also, the only vertices of S_2 that are in C_0 must be neighbors of a vertex of S' of degree at most q , hence $|S_2 \cap C_0| \leq kq$, and since $|G - C_0| \leq q$, we get that $|S_2| \leq q + kq$.

We conclude that $S \subseteq S_2$. So we can guess S in time $f(q, k, d) \cdot n^{\mathcal{O}(d)}$. ◀

Note that Lemma 25 applies to skeletons for both DOMINATED CLUSTER DELETION and ELIMINATION DISTANCE TO DOMINATED CLUSTERS. Furthermore, the dominating set D for the large connected component is computed (by brute-force).

5 Annotated versions for Dominated Cluster Deletion

► **Definition 26** (ANNOTATED DOMINATED CLUSTER DELETION). *Given a graph G , integers k, d , and vertex sets F, R, B , the (k, d) -ANNOTATED DOMINATED CLUSTER DELETION problem is to find at most k vertices $S \subseteq G - F$ such that every connected component C of $G - S$ has a Red-Blue dominating set of size at most d (i.e. d Blue vertices dominating all Red vertices).*

The next proposition states that this problem can be solved on instances we call bag graphs.

► **Definition 27.** *A graph G is called a (q, k, d) -bag graph, if the vertices of G can be partitioned into two vertex sets I, E called respectively interior vertices and exterior vertices, such that:*

- every connected component of $G[E]$ has at most $2d + 1$ vertices,
- every connected component of $G[E]$ has at most q neighbors in I , and
- for every separation L, R of G of order k such that $L \cap R \subseteq I$, we have that $|L \cap I| \leq q$ or $|R \cap I| \leq q$.

► **Proposition 28.** *There is an algorithm such that, given a (q, k, d) -bag graph $G = (I, E)$, and vertex sets F, R, B of G satisfying $E \subseteq F$ and $|E \cap B| \leq qd$, the algorithm decides whether G is a positive instance to (k, d) -ANNOTATED DOMINATED CLUSTER DELETION, in time $f(q, k, d, t(G)) \cdot |G|^{O(1)}$, where $t(\cdot)$ is a parameter satisfying Property 1.*

We prove Proposition 28 in the full version [34]. Using this, we then prove the following statement via dynamic programming on an unbreakable tree decomposition, see [34].

► **Theorem 29.** *There is an algorithm solving ANNOTATED DOMINATED CLUSTER DELETION in time $f(q, k, d, t(G)) \cdot |G|^{O(1)}$, where $t(\cdot)$ is a parameter satisfying Property 2.*

Remember that Property 2 implies that some parameter with Property 1 is bounded on bag graphs built in our algorithm, so we can use Proposition 28 on these bag graphs. Note that Theorem 29 implies Theorem 1, by setting $F = \emptyset$ and $R = B = V(G)$. The annotated version for ELIMINATION DISTANCE TO DOMINATED CLUSTERS is introduced and solved in the full version [34].

6 Hardness Results

In this section, we give the details of the hardness results mentioned in the introduction.

Para-NP-hardness for parameter $k + \Delta$ and for parameter d . A problem is para-NP-hard with respect to a parameter if the restriction to instances whose parameter value is at most a certain constant, is itself an NP-hard problem.

► **Theorem 30.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS is para-NP-hard for parameter $k + \Delta$.*

Proof. Setting $k = 0$ and $\Delta = 3$ corresponds to DOMINATING SET on degree 3 graphs. This problem is known to be NP-hard [23], which implies the statement. ◀

► **Theorem 31.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS is para-NP-hard for parameter d .*

Proof. Setting $d = 0$ corresponds to TREEDEPTH. This problem is known to be NP-hard [32], which implies the statement. ◀

W[2]-hardness for parameter $k + d$. A problem is W[2]-hard if a W[2]-hard problem reduces to it via an FPT-reduction.

► **Theorem 32.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS is W[2]-hard for parameter $k + d$.*

Proof. The W[2]-hard DOMINATING SET [15] reduces to ELIMINATION DISTANCE TO DOMINATED CLUSTERS by simply setting $k = 0$. This is an FPT-reduction. ◀

No polynomial kernels for parameter $k + c + d$. A kernelization algorithm (or kernel) for a parameterized problem is a polynomial-time algorithm that maps each instance (x, k) of a parameterized problem L to an instance (x', k') of L such that

- $(x, k) \in L \Leftrightarrow (x', k') \in L$, and
- $|x'| + k' \leq f(k)$ for a computable function f .

A kernel is polynomial if f is. We use the AND-cross-composition technique [19] to show that ELIMINATION DISTANCE TO DOMINATED CLUSTERS does not admit a polynomial kernel unless $\text{NP} \subseteq \text{coNP/poly}$. If an NP-hard language L AND-cross-composes into a parameterized language Q , then Q does not admit a polynomial kernel, unless $\text{NP} \subseteq \text{coNP/poly}$. Since the treedepth of a disjoint union of a family of graphs is equal to the maximum over the treedepth of these graphs, the disjoint union yields an AND-cross-composition from the unparameterized version of TREEDEPTH into the parameterized one. As computing the treedepth of a graph is NP-hard [32], it follows that TREEDEPTH not admit a polynomial kernel, unless $\text{NP} \subseteq \text{coNP/poly}$. It remains to prove that TREEDEPTH is also NP-complete on 2-degenerate graphs. Recall that c denotes the degeneracy of a graph. A graph G is c -degenerate if every subgraph H has a vertex of degree at most c (in H).

► **Theorem 33.** *TREEDEPTH is NP-hard on 2-degenerate graphs.*

Proof. We reduce TREEDEPTH to TREEDEPTH on 2-degenerate graphs. Given a graph G and parameter k , we compute H by replacing each edge of G by two disjoint paths of length 2 with the same endpoints. We set $k' = k + 1$ and prove that G has treedepth at most k if and only if H has treedepth at most k' . Observe that H is 2-degenerate.

Assume that G has treedepth at most k with an elimination tree T . We take the same elimination tree for H and in the very last step eliminate the isolated subdivision vertices.

Vice versa assume that H has treedepth at most $k + 1$ with an elimination tree of depth at most $k + 1$. Observe that every subdivision vertex must be comparable in the tree-order with both of its endpoints, as each edge must be controlled by the ancestor-descendant relation in the elimination tree. Now, if a subdivision vertex is eliminated before one of its endpoints, we can simply exchange the subdivision vertex with this endpoint and thereby obtain an elimination tree that is not deeper than the original one. In this way, we obtain an elimination tree where the subdivision vertices are on the lowest level. Then the tree with the subdivision vertices removed is an elimination tree for G of depth at most k . ◀

► **Corollary 34.** *ELIMINATION DISTANCE TO DOMINATED CLUSTERS does not admit a polynomial kernel with respect to parameter k even for fixed parameters $d = 0$ and $c = 2$.*

7 Conclusion

We have studied the DOMINATED CLUSTER DELETION problem and resolved the open question by Bentert et al. [4] whether the problem is fixed-parameter tractable with respect to the parameters $k + d + c$, by proving uniform fixed-parameter tractability even for the smaller parameter $k + d + \ell$, where ℓ is the semi-ladder index of the input graph. We also introduced the more general ELIMINATION DISTANCE TO DOMINATED CLUSTERS problem and almost fully classified its parameterized and kernelization complexity. The most interesting missing part of the classification is whether the case $d = 0$, which corresponds to TREEDEPTH, is NP-complete on graphs of bounded maximum degree. We conjecture that this is the case. Our proofs combine the main tools for addressing cut problems, the decomposition theorem into unbreakable parts by Cygan et al. [12] with the main tools for addressing domination-type problems [20].

References

- 1 Akanksha Agrawal, Lawqueen Kanesh, Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, Saket Saurabh, and Meirav Zehavi. Deleting, eliminating and decomposing to hereditary classes are all fpt-equivalent. In Joseph (Seffi) Naor and Niv Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 1976–2004. SIAM, 2022. doi:10.1137/1.9781611977073.79.
- 2 Akanksha Agrawal, Lawqueen Kanesh, Fahad Panolan, MS Ramanujan, and Saket Saurabh. A fixed-parameter tractable algorithm for elimination distance to bounded degree graphs. *SIAM Journal on Discrete Mathematics*, 36(2):911–921, 2022. doi:10.1137/21m1396824.
- 3 Akanksha Agrawal and MS Ramanujan. On the parameterized complexity of clique elimination distance. In *15th International Symposium on Parameterized and Exact Computation (IPEC 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.IPEC.2020.1.
- 4 Matthias Bentert, Michael R. Fellows, Petr A. Golovach, Frances A. Rosamond, and Saket Saurabh. Breaking a graph into connected components with small dominating sets. In Rastislav Kráľovic and Antonín Kucera, editors, *49th International Symposium on Mathematical Foundations of Computer Science, MFCS 2024, August 26-30, 2024, Bratislava, Slovakia*, volume 306 of *LIPIcs*, pages 24:1–24:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.MFCS.2024.24.
- 5 Stéphane Bessy, Marin Bougeret, Dimitrios M Thilikos, and Sebastian Wiederrecht. Kernelization for graph packing problems via rainbow matching. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3654–3663. SIAM, 2023. doi:10.1137/1.9781611977554.ch139.
- 6 Hans L. Bodlaender, Édouard Bonnet, Lars Jaffke, Dusan Knop, Paloma T. Lima, Martin Milanic, Sebastian Ordyniak, Sukanya Pandey, and Ondrej Suchý. Treewidth is NP-complete on cubic graphs. In *18th International Symposium on Parameterized and Exact Computation, (IPEC 2023)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.IPEC.2023.7.
- 7 Hans L Bodlaender and Dimitrios M Thilikos. Treewidth for graphs with small chordality. *Discrete Applied Mathematics*, 79(1-3):45–61, 1997. doi:10.1016/S0166-218X(97)00031-0.
- 8 Mikolaj Bojanczyk and Michal Pilipczuk. Definability equals recognizability for graphs of bounded treewidth. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 407–416. ACM, 2016. doi:10.1145/2933575.2934508.
- 9 Édouard Bonnet, Eun Jung Kim, Stéphan Thomassé, and Rémi Watrigant. Twin-width I: tractable FO model checking. *ACM Journal of the ACM (JACM)*, 69(1):1–46, 2021. doi:10.1145/3486655.
- 10 Jannis Bulian and Anuj Dawar. Graph isomorphism parameterized by elimination distance to bounded degree. *Algorithmica*, 75(2):363–382, 2016. doi:10.1007/s00453-015-0045-3.
- 11 Bruno Courcelle, Johann A Makowsky, and Udi Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000. doi:10.1007/s002249910009.
- 12 Marek Cygan, Paweł Komosa, Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, Saket Saurabh, and Magnus Wahlström. Randomized contractions meet lean decompositions. *ACM Transactions on Algorithms (TALG)*, 17(1):1–30, 2020. doi:10.1145/3426738.
- 13 Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. Minimum bisection is fixed-parameter tractable. *SIAM J. Comput.*, 48(2):417–450, 2019. doi:10.1137/140988553.
- 14 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

- 15 Rod G Downey and Michael R Fellows. Fixed-parameter tractability and completeness II: On completeness for W [1]. *Theoretical Computer Science*, 141(1-2):109–131, 1995. doi:10.1016/0304-3975(94)00097-3.
- 16 Jan Dreier, Ioannis Eleftheriadis, Nikolas Mählmann, Rose McCarty, Michal Pilipczuk, and Szymon Torunczyk. First-order model checking on monadically stable graph classes. *CoRR*, abs/2311.18740, 2023. doi:10.48550/arXiv.2311.18740.
- 17 Jan Dreier, Nikolas Mählmann, and Sebastian Siebertz. First-order model checking on structurally sparse graph classes. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC 2023)*, pages 567–580, 2023. doi:10.1145/3564246.3585186.
- 18 Jan Dreier, Sebastian Ordyniak, and Stefan Szeider. Sat backdoors: Depth beats size. *Journal of Computer and System Sciences*, 142:103520, 2024. doi:10.1016/j.jcss.2024.103520.
- 19 Andrew Drucker. New limits to classical and quantum instance compression. *SIAM Journal on Computing*, 44(5):1443–1479, 2015. doi:10.1137/130927115.
- 20 Grzegorz Fabianski, Michal Pilipczuk, Sebastian Siebertz, and Szymon Torunczyk. Progressive algorithms for domination and independence. *CoRR*, abs/1811.06799, 2018. doi:10.48550/arXiv.1811.06799.
- 21 Aleksander Figiel, Anne-Sophie Himmel, André Nichterlein, and Rolf Niedermeier. On 2-clubs in graph-based data clustering: theory and algorithm engineering. In Tiziana Calamoneri and Federico Corò, editors, *Algorithms and Complexity - 12th International Conference, CIAC 2021, Virtual Event, May 10-12, 2021, Proceedings*, volume 12701 of *Lecture Notes in Computer Science*, pages 216–230. Springer, Springer, 2021. doi:10.1007/978-3-030-75242-2_15.
- 22 Fedor V Fomin, Petr A Golovach, and Dimitrios M Thilikos. Parameterized complexity of elimination distance to first-order logic properties. *ACM Transactions on Computational Logic (TOCL)*, 23(3):1–35, 2022. doi:10.1145/3517129.
- 23 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, volume 174. W. H. Freeman, 1979.
- 24 Martin Grohe, Stephan Kreutzer, and Sebastian Siebertz. Deciding first-order properties of nowhere dense graphs. *Journal of the ACM (JACM)*, 64(3):1–32, 2017. doi:10.1145/3051095.
- 25 Jiong Guo, Falk Hüffner, and Rolf Niedermeier. A structural view on parameterizing problems: Distance from triviality. In Rodney G. Downey, Michael R. Fellows, and Frank K. H. A. Dehne, editors, *Parameterized and Exact Computation, First International Workshop, IWPEC 2004, Bergen, Norway, September 14-17, 2004, Proceedings*, volume 3162 of *Lecture Notes in Computer Science*, pages 162–173. Springer, Springer, 2004. doi:10.1007/978-3-540-28639-4_15.
- 26 Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. A more relaxed model for graph-based data clustering: s-plex cluster editing. *SIAM Journal on Discrete Mathematics*, 24(4):1662–1683, 2010. doi:10.1137/090767285.
- 27 Eva-Maria C Hols, Stefan Kratsch, and Astrid Pieterse. Elimination distances, blocking sets, and kernels for vertex cover. *SIAM Journal on Discrete Mathematics*, 36(3):1955–1990, 2022. doi:10.1137/20m1335285.
- 28 Hong Liu, Peng Zhang, and Daming Zhu. On editing graphs into 2-club clusters. In Jack Snoeyink, Pinyan Lu, Kaile Su, and Lusheng Wang, editors, *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management - Joint International Conference, FAW-AAIM 2012, Beijing, China, May 14-16, 2012. Proceedings*, volume 7285 of *Lecture Notes in Computer Science*, pages 235–246. Springer, Springer, 2012. doi:10.1007/978-3-642-29700-7_22.
- 29 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)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ICALP.2018.135.

- 30 Nikolas Mählmann, Sebastian Siebertz, and Alexandre Vigny. Recursive backdoors for SAT. In Filippo Bonchi and Simon J. Puglisi, editors, *46th International Symposium on Mathematical Foundations of Computer Science (MFCS 2021)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.MFCS.2021.73.
- 31 Michal Pilipczuk, Nicole Schirrmacher, Sebastian Siebertz, Szymon Torunczyk, and Alexandre Vigny. Algorithms and data structures for first-order logic with connectivity under vertex failures. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ICALP.2022.102.
- 32 Alex Pothén. The complexity of optimal elimination trees. *Technical Report*, 1988.
- 33 Nicole Schirrmacher, Sebastian Siebertz, and Alexandre Vigny. First-order logic with connectivity operators. *ACM Transactions on Computational Logic*, 24(4):1–23, 2023. doi:10.1145/3595922.
- 34 Nicole Schirrmacher, Sebastian Siebertz, and Alexandre Vigny. Elimination distance to dominated clusters. *CoRR*, abs/2504.21675, 2025. doi:10.48550/arXiv.2504.21675.
- 35 Dekel Tsur. Faster parameterized algorithm for cluster vertex deletion. *Theory of Computing Systems*, 65(2):323–343, 2021. doi:10.1007/s00224-020-10005-w.
- 36 René Van Bevern, Hannes Moser, and Rolf Niedermeier. Approximation and tidying—a problem kernel for s-plex cluster vertex deletion. *Algorithmica*, 62:930–950, 2012. doi:10.1007/s00453-011-9492-7.