AN FPT ALGORITHM FOR ELIMINATION DISTANCE TO BOUNDED DEGREE GRAPHS*

AKANKSHA AGRAWAL[†], LAWQUEEN KANESH [‡], FAHAD PANOLAN [§], M. S. RAMANUJAN[¶], AND SAKET SAURABH[‖]

Abstract. In the literature on parameterized graph problems, there has been an increased effort in recent years aimed at exploring novel notions of graph edit-distance that are more powerful than the size of a modulator to a specific graph class. In this line of research, Bulian and Dawar [Algorithmica, 2016] introduced the notion of elimination distance and showed that deciding whether a given graph has elimination distance at most k to any minor-closed class of graphs is fixed-parameter tractable parameterized by k [Algorithmica, 2017]. They showed that Graph Isomorphism parameterized by the elimination distance to bounded degree graphs is fixed-parameter tractable and asked whether determining the elimination distance to the class of bounded degree graphs is fixed-parameter tractable. Recently, Lindermayr et al. [MFCS 2020] obtained a fixed-parameter algorithm for this problem in the special case where the input is restricted to K_5-minor free graphs.

In this paper, we answer the question of Bulian and Dawar in the affirmative for general graphs. In fact, we give a more general result capturing elimination distance to any graph class characterized by a finite set of graphs as forbidden induced subgraphs.

1. Introduction.

A popular methodology for studying the parameterized complexity of problems is to consider *parameterization by distance from triviality* [16]. In this methodology, the idea is to try and lift the tractability of special cases of generally hard computational problems, to tractability of instances that are "close" to these special cases (i.e., close to triviality) for appropriate notions of "distance from triviality". This way of parameterizing graph problems has led to a rich collection of sophisticated algorithmic and lower bound machinery over the last two decades.

One direction in which this approach has been extended in recent years is by enhancing existing notions of distance from triviality by exploiting some form of *structure* underlying vertex modulators rather than just the size bound. This line of exploration has led to the development of several new notions of distance from triviality [12, 6, 7, 15, 14, 11]. Of primary interest to us in this line of research is the notion of *elimination distance* introduced in [6]. Bulian and Dawar [6] introduced the notion of elimination distance in an effort to define tractable parameterizations that are more general than the modulator size for graph problems. We refer the reader to Section 2 for a formal definition of this parameter. In their work, they focused on the Graph Isomorphism (GI) problem and showed that GI is fixed-parameter tractable (FPT) when parameterized by the elimination distance to graphs of bounded degree. In follow-up work, Bulian and Dawar [7] showed that deciding whether a given graph has elimination distance at most k to any minor-closed class of graphs is fixed-parameter tractable parameterized by k (i.e., can be solved in time $f(k)n^{\mathcal{O}(1)}$) and asked whether computing the elimination distance to graphs of bounded degree is fixed-parameter tractable.

Recently, Lindermayr et al. [18] showed that computing elimination distance to bounded degree graphs is fixed-parameter tractable *when the input is planar.* However their approach is specifically adapted to planar graphs (in fact, more generally, K_5-minor free graphs) and they note that their approach does not appear to extend to general graphs. In this paper, we address the general question and show that the problem is (non-uniformly) fixed-parameter tractable on general graphs. In fact, we prove a more general result and obtain the result for elimination distance to graphs of bounded degree as a consequence. Let $\mathcal{F}$ be a finite family of graphs. We say that a graph G is $\mathcal{F}$-free if G does not contain any induced subgraph isomorphic to a graph in $\mathcal{F}$.

*Extended abstract appeared at STACS 2021 [1]

Funding: Agrawal is supported by New Faculty Initiation Grant IIT Madras. Panolan is supported by Seed grant, IIT Hyderabad (SG/IITH/F224/2020-21/SG-79). Ramanujan is supported by Engineering and Physical Sciences Research Council's New Investigator Award (EP/V007793/1). Saurabh is supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819416); and he also acknowledges the support of Swarnajayanti Fellowship grant DST/SJF/MSA-01/2017-18.

[†]Indian Institute of Technology Madras, Chennai, India, akanksha@cse.iitm.ac.in

[‡]National University of Singapore, Singapore, Singapore, lawqueen@comp.nus.edu.sg

[§]Indian Institute of Technology, Hyderabad, India, panolan@iith.ac.in

[¶]University of Warwick, UK, r.maadapuzhi-sridharan@warwick.ac.uk

[‖]University of Bergen, Norway and Institute of Mathematical Sciences, HBNI, India, saket@imsc.res.in

THEOREM 1.1. *For every fixed finite family $\mathcal{F}$ of finite graphs and $k \in \mathbb{N}$, there is an algorithm $\mathcal{A}_k^{\mathcal{F}}$ that, given a graph G, runs in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some function f and correctly decides whether G has elimination distance at most k to the class of $\mathcal{F}$-free graphs.*

We remark that the exponent of n in the above running time is a constant depending on $\mathcal{F}$ and independent of k. As a corollary (and with a slight modification of the above algorithms), we obtain the following result for determining elimination distance to bounded degree graphs.

COROLLARY 1.2. *For every $k \in \mathbb{N}$, there is an algorithm $\mathcal{A}_k$ that, given a graph G and integer $d \in \mathbb{N}$, runs in time $f(k, d) \cdot n^{\mathcal{O}(1)}$ for some function f and correctly decides whether G has elimination distance at most k to the class of graphs of degree at most d.*

In the above statement, the exponent of n is a constant independent of both d and k.

Related work.. Hols et al. [17] recently presented a comprehensive study of the classic Vertex Cover problem parameterized by the size of a smallest modulator to graphs that have bounded elimination distance to specific hereditary graph classes. They provided an elegant (partial) characterization of parameterizations that permit polynomial kernelizations for Vertex Cover. Bougeret et al. [4] introduced a measure called bridge-depth and showed that a minor-closed family of graphs $\mathcal{F}$ has bounded bridge-depth precisely when Vertex Cover admits a polynomial kernel parameterized by the size of a modulator to $\mathcal{F}$ (subject to standard complexity theoretic hypotheses). The notion of elimination distance [6] generalizes the notion of *generalized treedepth* introduced by Bouland et al. [5] in an effort to combine the treedepth and max-leaf number parameters. Building on [6] and extending the approach of combining width parameters (treedepth in the case of [6]) and modulator size, Ganian et al. [15] proposed a measure of distance to triviality for CSP that depended on the treewidth of an appropriate graph defined on backdoor sets (these can be thought of as a version of vertex modulators appropriate for use in solving ILP and CSP instances). That is, they introduced a way of combining *treewidth* and modulator size into a single parameter that is stronger than elimination distance. More recently, Eiben et al. [11] continued the line of research into combining modulators and width parameters by studying this parameter in the context of graph problems, where triviality is expressed in terms of bounded rankwidth.

2. Preliminaries. For an undirected graph G, we use n and m to denote $|V(G)|$ and $|E(G)|$ respectively, unless mentioned otherwise. For $X \subseteq V(G)$, $G[X]$ denotes the graph with vertex set X and the edge set $\{\{x, y\} \in E(G) \mid x, y \in X\}$. By $G - X$ we denote the graph $G[V(G) \setminus X]$. Let $v \in V(G)$. Then, by $N_G(v)$ we denote the set of neighbors of v in G, i.e., the set $\{u \in V(G) \mid \{u, v\} \in E(G)\}$. By $N_G[v]$, we denote the closed neighborhood of v in G, i.e., $N_G(v) \cup \{v\}$. For a set $U \subseteq V(G)$, by $N_G(U)$ we denote the set $\cup_{u \in U} N_G(u) \setminus U$, by $N_G[U]$ we denote the set $N_G(U) \cup U$. By $deg_G(v)$, we denote the degree of vertex v in G, i.e., the number of edges incident on v in G. We drop the subscript whenever the context is clear.

A *path* $P = (v_1, v_2, \cdots, v_\ell)$ in G is a subgraph of G, where the set $V(P) = \{v_1, v_2, \cdots, v_\ell\} \subseteq V(G)$ is a set of distinct vertices and $E(P) = \{\{v_i, v_{i+1}\} \mid i \in [\ell - 1]\} \subseteq E(G)$, where $|V(P)| = \ell$ for some $\ell \in [|V(G)|]$. The above defined path P is called as $v_1 - v_\ell$ path. We say that the graph G is *connected* if for every $u, v \in V(G)$, there exists a $u - v$ path in G. A *connected component* of G is an inclusion-wise maximal connected induced subgraph of G. The set $\mathcal{C}(G)$ denotes the set of connected components of G. For a tree T and vertices $u, v \in V(T)$, we denote the unique path between u and v by $\mathsf{Pth}_T(u, v)$. A tree is called as a *rooted tree* if special vertex in tree is designated to be the *root*. Let T be a rooted tree with root $r \in V(T)$. We say that a vertex $v \in V(T) \setminus \{r\}$ is a *leaf* of T if the $deg_T(v) = 1$. Moreover, if $V(T) = \{r\}$, then r is the leaf (as well as the root) of T. A vertex which is not a leaf, is a *non-leaf* vertex. Let $t, t' \in V(T)$ such that $\{t, t'\} \in E(T)$ and t' is not contained in $t - r$ path in T, then we say that t is the *parent* of t' and t' is a *child* of t. A vertex $t' \in V(T)$ (t' can possibly be the same as t) is a *descendant* of t, if in $T - \{\mathsf{par}_T(t)\}$, where $\mathsf{par}_T(t)$ is the parent of t, there is a $t - t'$ path. Note that when $t = r$, then $T - \{\mathsf{par}_T(t)\} = T$, as the parent of r does not exist. (Every vertex in T is a descendant of r.) By $\mathsf{desc}_T(t)$, we denote the set of all descendants of t in T. We drop the subscript T from $\mathsf{par}_T(\cdot)$ and $\mathsf{desc}_T(\cdot)$, when the context is clear. A *rooted forest* is a forest where each connected component is a rooted tree. For a rooted forest F, a vertex $v \in V(F)$ that is not a root of any of its rooted trees is a *leaf* if it is of degree exactly one in F. We denote the set of leaves in a rooted forest by $\mathsf{Lf}(F)$. The *depth*, denoted by $\mathsf{depth}(T)$ of a rooted tree T is the maximum number vertices in a root to leaf path in

T. The *depth*, denoted by $\mathsf{depth}(F)$ of a rooted forest F is the maximum over the depths of its rooted trees.

DEFINITION 2.1 (Forest embedding). A *forest embedding* of a graph G is a pair (F, f), where F is a rooted forest and $f : V(G) \to V(F)$ is a bijective function, such that for each $\{u, v\} \in E(G)$, either $f(u)$ is a descendant of $f(v)$, or $f(v)$ is a descendant of $f(u)$. The *depth* of the forest embedding (F, f) is the depth of the rooted forest F.[1]

Next, we recall the notion of *elimination-distance* introduced by Bulian and Dawar [6]. We rephrase their definition and introduce notation that will facilitate our presentation.

DEFINITION 2.2 (Elimination Distance and $(\eta, \mathcal{H})$-decompositions). Consider a family $\mathcal{H}$ of graphs and an integer $\eta \in \mathbb{N}$. An $(\eta, \mathcal{H})$-decomposition of a graph G is a tuple $(X, Y, F, f : X \to V(F), g : \mathcal{C}(G[Y]) \to \mathsf{Lf}(F) \cup \{\perp\})$, where (X, Y) is a partition of $V(G)$ and F is a rooted forest of depth η, such that the following conditions are satisfied:

1. (F, f) is a forest embedding of $G[X]$,
2. each connected component of $G[Y]$ belongs to $\mathcal{H}$, and
3. for a connected component C of $G[Y]$, a vertex $v \in V(C)$, and an edge $\{u, v\} \in E(G)$, either $u \in Y$ or $f(u)$ is a vertex in the unique path in F from r to $g(C)$, where r is the root of the connected component in F containing the vertex $g(C)$.[2]

We say that G admits an $(\eta', \mathcal{H})$-decomposition if there is some $\eta \leq \eta'$, for which there is an $(\eta, \mathcal{H})$-decomposition of G. The *elimination distance* of G to $\mathcal{H}$ (or the $\mathcal{H}$-*elimination distance* of G) is the smallest integer η^* for which G admits an $(\eta^*, \mathcal{H})$-decomposition.

We remark that when G is a connected graph, no component in $\mathcal{C}(G[Y])$ is mapped to $\perp$ (i.e., g is indeed a function from $\mathcal{C}(G[Y])$ to $\mathsf{Lf}(F)$).

Consider an $(\eta, \mathcal{H})$-decomposition $\mathbb{D} = (X, Y, F, f, g)$ of a graph G. We say that $\mathbb{D}$ is an $(\eta, \mathcal{H})$-decomposition of G on forest F. We say that X is the *interior part* of $\mathbb{D}$ and Y is the *exterior part* of $\mathbb{D}$. For a leaf $u \in \mathsf{Lf}(F)$, by $\widehat{P}_u^{\mathbb{D}}$ we denote the path from u to r in the tree T, where T is the tree rooted at r in F, containing u. Moreover, by $P_u^{\mathbb{D}}$, we denote the graph $G[\{f^{-1}(w) \mid w \in V(\widehat{P}_u^{\mathbb{D}})\}]$. For a connected component $C \in \mathcal{C}(G[Y])$, by $C_{\mathsf{ext}}^{\mathbb{D}}$ we denote the graph $G[V(C) \cup \{f^{-1}(w) \mid w \in V(\mathsf{Pth}_F(g(C), r))\}]$, where r is the root of the component of F containing $g(C)$. (For the above notations we drop the superscript $\mathbb{D}$, when the context is clear.)

For a graph G, the elimination distance of G to $\mathcal{H} = \{(\{u\}, \emptyset)\}$ is the treedepth of G, denoted by $\mathsf{td}(G)$. We use the following simple observation in a later section.

OBSERVATION 1. *Let q be a positive integer and $\mathcal{H}_q$ be a family of graphs where each graph has at most q vertices. If for a graph G, the elimination distance of G to $\mathcal{H}_q$ is η, then the treedepth of G is at most $\eta + q$.*

Consider a graph G, a non-empty set $Q \subseteq V(G)$, and integers $p, q \in \mathbb{N}$. We say that a set $A \subseteq V(G)$ such that $G[A]$ is connected, is a (Q, p, q)-connected set if $Q \subseteq A$, $|A| = p$, and $|N(A)| \leq q$. The next proposition follows from Lemma 3.1 of [13].

PROPOSITION 2.3. *Consider a graph G, a non-empty set $Q \subseteq V(G)$, and integers $p, q \in \mathbb{N}$. The number of (Q, p, q)-connected sets in G is at most 2^{p+q}. Moreover, there exists an algorithm which runs in $2^{p+q} \cdot n^{\mathcal{O}(1)}$ time and enumerates all (Q, p, q)-connected sets in G'.*

Unbreakability.. To formally introduce the notion of unbreakability, we rely on the definition of a separation:

DEFINITION 2.4. [**Separation**] *A pair (X, Y) where $X \cup Y = V(G)$ is a* separation *if $E(X \setminus Y, Y \setminus X) = \emptyset$. The order of (X, Y) is $|X \cap Y|$.*

Roughly speaking, a graph is breakable if it is possible to "break" it into two large parts by removing only a small number of vertices. Formally,

DEFINITION 2.5. [**(s, c)-Unbreakable graph**] *Let G be a graph. 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.

[1]Sometimes we slightly abuse the notation for simplicity, and say that, for every $\beta \geq \alpha$, (F, f) is a forest embedding of depth β, where α is the depth of F.

[2]If $g(C) = \perp$, then u must belong to Y.

Counting Monadic Second Order Logic. . The syntax of Monadic Second Order Logic (MSO) of graphs includes the logical connectives $\vee$, $\wedge$, $\neg$, $\leftrightarrow$, $\Rightarrow$, variables for vertices, edges, sets of vertices and sets of edges, the quantifiers $\forall$ and $\exists$, which can be applied to these variables, and five binary relations:

1. $u \in U$, where u is a vertex variable and U is a vertex set variable;
2. $d \in D$, where d is an edge variable and D is an edge set variable;
3. $\mathbf{inc}(d,u)$, where d is an edge variable, u is a vertex variable, and the interpretation is that the edge d is incident to u;
4. $\mathbf{adj}(u,v)$, where u and v are vertex variables, and the interpretation is that u and v are adjacent;
5. equality of variables representing vertices, edges, vertex sets and edge sets.

Counting Monadic Second Order Logic (CMSO) extends Monadic Second Order Logic (MSO) by including atomic sentences testing whether the cardinality of a set is equal to q modulo r, where q and r are integers such that $0 \leq q < r$ and $r \geq 2$. That is, CMSO is MSO with the following atomic sentence: $\mathbf{card}_{q,r}(S) = \mathbf{true}$ if and only if $|S| \equiv q \pmod r$, where S is a set. We refer to [2, 8, 9] for a detailed introduction to CMSO.

We will crucially use the following result of Lokshtanov et al. [19] that allows one to obtain a (non-uniform) FPT algorithm for CMSO-expressible graph problems by designing an FPT algorithm for the problem on unbreakable graphs.

Proposition 2.6 (Theorem 1, [19]). *Let ψ be a CMSO sentence and let $d > 4$ be a positive integer. For all $c \in \mathbb{N}$, there exists $s \in \mathbb{N}$ such that if there exists an algorithm that solves* CMSO$[\psi]$ *on (s,c)-unbreakable graphs in time $\mathcal{O}(n^d)$, then there exists an algorithm that solves* CMSO$[\psi]$ *on general graphs in time $\mathcal{O}(n^d)$.*

3. The algorithm for k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$. In the rest of the paper, we fix the family $\mathcal{F}$ and let d denote the maximum taken over the number of vertices in the graphs in $\mathcal{F}$. Recall that $\mathcal{H}_{\mathcal{F}}$ denotes the family of all $\mathcal{F}$-free graphs. In the k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ problem, the input is a graph G and the goal is to determine whether G has elimination distance at most k to $\mathcal{H}_{\mathcal{F}}$. Notice that k is assumed to be fixed since it is part of the problem definition. For a graph G, we say that $X \subseteq V(G)$ is a *k-elimination distance modulator* of G to $\mathcal{F}$-free graphs (k-ed modulator to $\mathcal{H}_{\mathcal{F}}$) if there exist Y, f, g such that (X, Y, F, f, g) is a (k, H_F)-decomposition of G.

Lindermayr et al. [18] obtain their algorithm by repeatedly either identifying an irrelevant vertex or by identifying substructures that they call 'connectivity patterns' using which they formulate an appropriate CMSO formula over a bounded treedepth structure. In their algorithm, they crucially require that the input graph is K_5-minor free. Moreover, they note that in their approach, all difficulties for elimination distance to bounded degree arise already in bounded degree graphs and conjectured that solving this problem on bounded degree graphs is sufficient to solve the problem on general graphs.

In our case, instead of focussing on bounded degree inputs as the hard case, we focus on input graphs that are well-connected everywhere (i.e., (s,c)-unbreakable for appropriate values of s and c). Towards this, we rely on Proposition 2.6, which requires the CMSO-expressibility of the k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ problem.

Lemma 3.1. *k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ is CMSO-expressible.*

Proof. Notice that $X \subseteq V(G)$ is a k-ed modulator to $\mathcal{H}_{\mathcal{F}}$ if and only if $G - X$ is $\mathcal{F}$-free and the graph $\mathsf{Torso}(X)$ has treedepth at most k. Here, $\mathsf{Torso}(X)$ is defined as the graph obtained from $G[X]$ by making the neighborhood of every connected component of $G - X$ a clique. In other words, for every distinct $u, v \in X$, (u,v) is an edge in $\mathsf{Torso}(X)$ if and only if either $(u,v) \in E(G)$ or there is a path in G whose endpoints are u and v and whose remaining vertices are disjoint from X. We use this characterization of k-ed modulators to $\mathcal{H}_{\mathcal{F}}$ to write our CMSO formula.

It is straightforward to assert that a graph is an induced subgraph of G in CMSO. Therefore, one can write a sentence asserting that X is a vertex subset in G such that $G - X$ has no induced subgraph isomorphic to a graph in $\mathcal{F}$. Now, consider the class of graphs of treedepth at most k and notice that these are minor closed and hence characterized by a set of $\beta(k)$ forbidden minors for some function β. Recall that k is fixed in our setting and so, we may assume that these forbidden minors are 'hardcoded' into our algorithm. Moreover, it is well-known that one can assert that a graph is a minor of a given graph, in CMSO. Hence, it only remains for us to describe how we express in CMSO, the graph $\mathsf{Torso}(\mathsf{X})$ for some $X \subseteq V(G)$. Here, we encode the edge-set of $\mathsf{Torso}(X)$ into a new binary relation $\mathbf{adj}^\star$ such that for $u, v \in X$ $\mathbf{adj}^\star(u, v)$ if and only if $\mathbf{adj}(u, v)$ or there exists a vertex set Y such that $Y \cap X = \{u, v\}$ and $\mathbf{conn}(Y)$ is true, where $\mathbf{conn}(Y)$ is the standard CMSO sentence that asserts that the graph $G[Y]$ is connected (see, e.g., [10]). This completes the proof of the lemma. ∎

3.1. k-Elimination Distance to $\mathcal{H}_\mathcal{F}$ **on** (α_k, k)**-Unbreakable Graphs.** Invoke Proposition 2.6 with $c = k$ and let α_k denote the value of s given by this invocation. Due to this proposition, it is sufficient for us to give an FPT algorithm for k-Elimination Distance to $\mathcal{H}_\mathcal{F}$ on (α_k, k)-unbreakable graphs and that will be our goal in the rest of this section.

We begin by recalling a straightforward enumeration algorithm to enumerate all minimal $\mathcal{H}_\mathcal{F}$-modulators of bounded size.

Observation 2. *There is an algorithm that runs in time $d^{\alpha_k+k}n^{\mathcal{O}(d)}$ and either correctly concludes that there is no $\mathcal{H}_\mathcal{F}$-modulator of G of size at most $\alpha_k + k$ or outputs a family $\mathcal{Z} = \{Z_1, \ldots, Z_\ell\}$ of at most d^{α_k+k} vertex sets comprising every minimal $\mathcal{H}_\mathcal{F}$-modulator of G of size at most $\alpha_k + k$.*

The above algorithm simply locates an induced subgraph of the given graph which is isomorphic to a graph in $\mathcal{F}$ by brute force and then branches on the vertices in this subgraph. We note that in the special case of $\mathcal{F}$ where $\mathcal{F}$-free graphs are precisely the set of graphs of degree at most some r, the exponent of n in the above algorithm can be made independent of r since computing a forbidden structure, i.e., a vertex of degree at least $r + 1$, is polynomial-time solvable. Applying this insight allows us to infer Corollary 1.2 by slightly modifying the application of Observation 2 in the proof of Theorem 1.1.

In what follows (Lemma 3.2–Lemma 3.4), consider an (α_k, k)-unbreakable graph G, and an integer k such that (G, k) is a yes-instance of (α_k, k)-Unbreakable Elimination Distance to $\mathcal{H}_\mathcal{F}$. For ease of presentation, we assume that G is a connected graph.

Lemma 3.2. *Let $\mathbb{D} = (X, Y, F, f : V(X) \to V(F), g : \mathcal{C}(G[Y]) \to \mathsf{Lf}(F))$ be a $(k, \mathcal{H}_\mathcal{F})$-decomposition of G. Then, the following properties hold:*

1. *$G[Y]$ contains at most one connected component of size at least α_k.*
2. *If $G[Y]$ does not contain a connected component of size at least α_k, then $\mathsf{td}(G) \le \alpha_k + k$.*
3. *If $\mathsf{td}(G) > \alpha_k + k$, then, $G[Y]$ contains exactly one connected component of size at least α_k.*

Proof. Consider the first statement. Suppose to the contrary that there are two connected components $C_1, C_2 \in \mathcal{C}(G[Y])$, such that $|V(C_1)| \ge \alpha_k$ and $|V(C_2)| \ge \alpha_k$. Recall that the set $N(C_1)$ is contained in a single root-to-leaf path in F, and hence has size at most k. Formally, let $g(C_1) = u$ and notice that by the definition of $(k, \mathcal{H}_\mathcal{F})$-decomposition (Item 3 of Definition 2.2), $N(C_1) \subseteq V(P_u^\mathbb{D})$. Also, since the depth of F is at most k, $|V(P_u^\mathbb{D})| \le k$, and hence $|N(C_1)| \le k$. Then, the separation $(N[C_1], V(G) \setminus C_1)$ is an (α_k, k)-witnessing separation, a contradiction to G being (α_k, k)-unbreakable.

For the second statement, if $G[Y]$ does not contain a connected component of size at least α_k, then indeed $\mathbb{D}$ is a $(k, \mathcal{H}_{\alpha_k})$-decomposition of G, where $\mathcal{H}_{\alpha_k}$ is the family of graphs with at most α_k vertices each. That is, the elimination distance of G to $\mathcal{H}_{\alpha_k}$ is at most k. Thus, by Observation 1, $\mathsf{td}(G) \le \alpha_k + k$.

From the previous two statements, we have that if $\mathsf{td}(G) > \alpha_k + k$, then $G[Y]$ contains exactly one connected component of size at least α_k, proving the final statement. ∎

The above lemma allows us to assume that either the treedepth (and hence also the treewidth) of the input graph is bounded by $\alpha_k + k$ (in which case one can use Lemma 3.1 and Courcelle's theorem [8] to solve the problem), or conclude that if (G, k) is a yes-instance, then $G[Y]$ contains exactly one connected component of size at least α_k. In the following two lemmas, we assume that G has treedepth at least $\alpha_k + k + 1$ and present crucial structural properties on which our main algorithm is based.

Lemma 3.3. *Let $\mathbb{D} = (X, Y, F, f : V(X) \to V(F), g : \mathcal{C}(G[Y]) \to \mathsf{Lf}(F))$ be a $(k, \mathcal{H}_\mathcal{F})$-*

decomposition of G. Moreover, suppose that $\mathsf{td}(G) > \alpha_k + k$ and let $Z \subseteq X$ such that $G - Z$ is $\mathcal{F}$-free. Let $C^\star$ be the unique connected component of $G - X$ of size at least α_k. Then, the following hold:

1. *There is a unique connected component of $G - Z$ that contains $V(C^\star)$.*
2. *$V(G) \setminus V(C^\star)$ has size at most $\alpha_k + k$.*
3. *$|X| \leq \alpha_k + k$.*

Proof. As $C^\star$ is a connected component in $G - X$ and $Z \subseteq X$, there is a unique connected component in $G - Z$ that contains $V(C^\star)$. This proves the first statement. Now we prove the second statement of the lemma. Let $x := g(C^\star)$. Then, we define $S := N_G(C^\star) \subseteq V(P_x^{\mathbb{D}})$ and have that $|S| \leq |V(P_x^{\mathbb{D}})| \leq k$. Let $V_1 = N[V(C^\star)]$ and $V_2 = V(G) \setminus V(C^\star)$. For the sake of contradiction, suppose that $|V(G) \setminus V(C^\star)| > \alpha_k + k$. Then, (V_1, V_2) is a separation in G of order at most k and $|V_1 \setminus V_2|, |V_2 \setminus V_1| \geq \alpha_k$. This contradicts the assumption that G is a (α_k, k)-unbreakable graph. The final statement follows from the fact that $X \subseteq V(G) \setminus V(C^\star)$. ◻

Lemma 3.4. *Let $\mathbb{D} = (X, Y, F, f : V(X) \to V(F), g : \mathcal{C}(G[Y]) \to \mathsf{Lf}(F))$ be a $(k, \mathcal{H}_{\mathcal{F}})$-decomposition of G and let $Z \subseteq X$ be a $\mathcal{H}_{\mathcal{F}}$-modulator of G. Let $C^\star$ be the unique connected component of $G - X$ of size at least α_k. Let $v^\star$ be an arbitrary vertex in $C^\star$ and let C be the component of $G - Z$ that contains $v^\star$. Let $J \subseteq N(C)$ be an arbitrary set of size $\min\{k+1, |N(C)|\}$. Let $J_C \subseteq V(C)$ be an arbitrary set of size at most $|J|$ such that $N(J_C) \supseteq J$. Then, one of the following statements hold:*

1. *There is a leaf $u \in \mathsf{Lf}(F)$ such that $J \subseteq V(\widehat{P}_u^{\mathbb{D}})$. That is, the vertices in J lie on a single root-to-leaf path in F.*
2. *$J_C \cap X \neq \emptyset$.*
3. *$J_C \nsubseteq V(C^\star)$.*

Proof. Suppose to the contrary that none of these statements hold. That is, there are vertices $v_1, v_2 \in J$ such that v_p is not a descendant of v_q in F for distinct $p, q \in \{1, 2\}$, $J_C \cap X = \emptyset$ and $J_C \subseteq V(C^\star)$. Since $N(J_C) \supseteq J$ and $J_C \subseteq V(C^\star)$, it follows that $J \subseteq N(C^\star)$. Item 3 of Definition 2.2 guarantees that J is contained in $\widehat{P}^{\mathbb{D}}_{g(C^\star)}$, implying that out of v_1 and v_2, one is the descendant of the other in F. This gives us a contradiction. ◻

We are now ready to present our algorithm for k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ on (α_k, k)-unbreakable graphs.

Lemma 3.5. *k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ on (α_k, k)-unbreakable graphs can be solved in time $f^\star(k, \alpha_k) \cdot n^{\mathcal{O}(1)}$ for some function $f^\star$.*

Proof. Let (G, k) be the input. We first check if the treewidth of G is bounded by $\alpha_k + k$ using the algorithm of Bodlaender [3]. If yes, then we solve the problem using Courcelle's theorem. Suppose that this is not the case. Suppose that the input (G, k) is a yes-instance of k-Elimination Distance to $\mathcal{H}_{\mathcal{F}}$ and let $\mathbb{D} = (X, Y, F, f : V(X) \to V(F), g : \mathcal{C}(G[Y]) \to \mathsf{Lf}(F))$ be a hypothetical $(k, \mathcal{H}_{\mathcal{F}})$-decomposition of G. Using Lemma 3.2, we may assume that there is a unique component $C^\star$ in $G[Y]$ that has size at least α_k. Moreover, as the treewidth of G is at least $\alpha_k + k$, the third statement of Lemma 3.3 guarantees that X has size at most $\alpha_k + k$. Our algorithm begins by guessing $v^\star$ and then uses Observation 2 to guess a set $Z \subseteq X$ that is a minimal $\mathcal{H}_{\mathcal{F}}$-modulator of size at most $\alpha_k + k$. There are n choices for $v^\star$ and $d^{\alpha_k + k}$ choices for Z. If the algorithm of Observation 2 concludes that a $\mathcal{H}_{\mathcal{F}}$-modulator of size at most $\alpha_k + k$ does not exist, then the second statement of Lemma 3.3 is used to correctly conclude that the input is a no-instance.

It then calls upon a subroutine $\mathsf{Alg\text{-}special}$ that takes as input G, k and a set $\widehat{Z}$ of size at most $\alpha_k + k$ and either correctly concludes that there is a set $X \supseteq \widehat{Z}$ of size at most $\alpha_k + k$ that is a k-elimination distance modulator to $\mathcal{H}_{\mathcal{F}}$, or correctly concludes that one does not exist. Our main algorithm invokes $\mathsf{Alg\text{-}special}$ with input G, k and $\widehat{Z} := Z$. In the rest of the proof, we describe and analyze $\mathsf{Alg\text{-}special}$.

$\mathsf{Alg\text{-}special}$ first checks whether $|\widehat{Z}| \leq \alpha_k + k$. If not, then it returns NO. Otherwise, it locates the component C of $G - \widehat{Z}$ that contains $v^\star$ (and hence also contains $V(C^\star)$) and computes J and J_C as described in Lemma 3.4. It then considers the following three exhaustive possibilites and branches over all of them:

(i) the vertices in J lie on a root-to-leaf path in F or

(ii) J_C intersects X or
(iii) at least one vertex of J_C is not contained in $C^\star$.

In Case (i), notice that $|J|$ must be at most k since the depth of F is at most k. Hence, by the definition of J, it must be the case that $J = N(C)$. We now observe that without loss of generality, C is disjoint from X. This is because C is $\mathcal{F}$-free, $N(C) \subseteq X$ and moreover, $N(C)$ lies on a root-to-leaf path in F. Thus, to solve the problem in Case (i), it is sufficient to check for the existence of a set X such that $X \subseteq V(G) \setminus V(C)$, $X \supseteq \widehat{Z}$, X has size at most $\alpha_k + k$ and X is a k-elimination distance modulator to $\mathcal{H}_{\mathcal{F}}$. However, since $X \subseteq V(G) \setminus V(C)$, it follows that X is contained in $V(G) \setminus V(C^\star)$, which has size at most $\alpha_k + k$ (second statement of Lemma 3.3). Therefore, the existence of X can be verified by going over all possible subsets of $X \subseteq V(G) \setminus V(C)$. This completes the description of the algorithm in Case (i).

To handle Case (ii), we recursively call $\mathsf{Alg\text{-}special}$ on input $(G, k, \widehat{Z} \cup \{v\})$ for each $v \in J_C$ and return YES if at least one of the recursive calls return YES.

Suppose that Cases (i) and (ii) do not hold. We now handle Case (iii) as follows. We know that for at least one vertex of J_C, say, $y^\star$, the connected component of $G[Y]$ containing $y^\star$ is not the same as $C^\star$. Let $C_{y^\star}$ be this component. From the third statement of Lemma 3.2, we have that exactly one component of $G[Y]$ has size at least α_k. Hence, $C_{y^\star}$ has size at most α_k. Moreover, $N(C_{y^\star})$ must have size at most k since it is contained in $V(P^{\mathbb{D}}_{g(C_{y^\star})})$, which has size at most k. Hence, we guess $y^\star \in J_C$, enumerate all $(y^\star, \alpha_k, k)$-connected sets B and recursively call $\mathsf{Alg\text{-}special}$ on $(G, k, \widehat{Z} \cup N(B))$. Notice that since $y^\star$ and $v^\star$ lie in the same component of $G - \widehat{Z}$, $N(B)$ contains at least one vertex from $V(C)$ and therefore, $|\widehat{Z} \cup N(B)| > |\widehat{Z}|$, indicating that we make progress towards the upper bound of $\alpha_k + k$ on $\widehat{Z}$.

The correctness follows from the fact that the branching is exhaustive (Lemma 3.4). Now we bound the running time of the algorithm $\mathsf{Alg\text{-}special}$. Notice that when $\widehat{Z} > \alpha_k + k$, the algorithm immediately outputs NO. Moreover, notice that the algorithm only recurses in Cases (ii) and (iii), where the size of $\widehat{Z}$ strictly increases. The number of recursive calls made in either of these cases is dominated by Case (iii), which is upper bounded by $2^{\mathcal{O}(\alpha_k+k)}$ (see Proposition 2.3). Hence, we conclude that the number of nodes in the branching tree is bounded by $2^{\mathcal{O}((\alpha_k+k)^2)}$. The running time at each node is upper bounded by the brute-force solution in Case (i) (which is bounded by $2^{\mathcal{O}((\alpha_k+k))}n^{\mathcal{O}(1)}$), plus the time to compute J_C and run the algorithm of Proposition 2.3 $|J_C|$ times. Hence, we conclude that the running time at each node is bounded by $2^{\mathcal{O}((\alpha_k+k))}n^{\mathcal{O}(1)}$, giving us a bound of $2^{\mathcal{O}((\alpha_k+k)^2)}n^{\mathcal{O}(1)}$ on the running time of $\mathsf{Alg\text{-}special}$. Observing that all other computational steps in our algorithm have running time dominated by the invocation of Courcelle's theorem on a graph of treewidth at most $\alpha_k + k$ completes the proof of the lemma. This completes the proof of the lemma. ∎

Lemma 3.5, Lemma 3.1 and Proposition 2.6 imply Theorem 1.1.

4. Discussions and future work. In this work, we have answered the question of Bulian and Dawar regarding the fixed-parameter tractability of computing elimination distance to bounded degree graphs. In fact, we give a more general result capturing elimination distance to any graph class characterized by a finite set of graphs as forbidden induced subgraphs.

Two further natural directions for further research on this topic arise.

1. For which other "base" graph classes $\mathcal{H}$ is deciding elimination distance to $\mathcal{H}$, fixed-parameter tractable? Can we characterize these perhaps, in an appropriate fragment of logic?
2. A second direction is to investigate the parameterized complexity of various graphs problems parameterized by the elimination distance to $\mathcal{F}$-free graphs. This will enable one to go beyond the work of Bulian and Dawar for Graph Isomorsphism parameterized by elimination distance to bounded degree graphs.

The success of the distance-from-triviality programme in parameterized complexity indicates that the study of parameterization of graph problems by elimination distance to well-studied graph classes is a promising direction for future research and we have given widely applicable techniques for the task of computing the elimination distance to CMSO-expressible graph classes.

REFERENCES

[1] A. Agrawal, L. Kanesh, F. Panolan, M. S. Ramanujan, and S. Saurabh. An FPT algorithm for elimination

distance to bounded degree graphs. In M. Bläser and B. Monmege, editors, *38th International Symposium on Theoretical Aspects of Computer Science, STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference)*, volume 187 of *LIPIcs*, pages 5:1–5:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[2] S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12:308–340, 1991.

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

[4] M. Bougeret, B. M. P. Jansen, and I. Sau. Bridge-depth characterizes which structural parameterizations of vertex cover admit a polynomial kernel. In A. Czumaj, A. Dawar, and E. Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 16:1–16:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[5] A. Bouland, A. Dawar, and E. Kopczynski. On tractable parameterizations of graph isomorphism. In D. M. Thilikos and G. J. Woeginger, editors, *Parameterized and Exact Computation - 7th International Symposium, IPEC 2012, Ljubljana, Slovenia, September 12-14, 2012. Proceedings*, volume 7535 of *Lecture Notes in Computer Science*, pages 218–230. Springer, 2012.

[6] J. Bulian and A. Dawar. Graph isomorphism parameterized by elimination distance to bounded degree. *Algorithmica*, 75(2):363–382, 2016.

[7] J. Bulian and A. Dawar. Fixed-parameter tractable distances to sparse graph classes. *Algorithmica*, 79(1):139–158, 2017.

[8] B. Courcelle. The monadic second-order logic of graphs I: Recognizable sets of finite graphs. *Inform. and Comput.*, 85:12–75, 1990.

[9] B. Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In *Handbook of graph grammars and computing by graph transformation, Vol. 1*, pages 313–400. World Sci. Publ, River Edge, NJ, 1997.

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

[11] E. Eiben, R. Ganian, T. Hamm, and O. Kwon. Measuring what matters: A hybrid approach to dynamic programming with treewidth. In P. Rossmanith, P. Heggernes, and J. Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 42:1–42:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[12] E. Eiben, R. Ganian, and S. Szeider. Meta-kernelization using well-structured modulators. In T. Husfeldt and I. A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPIcs*, pages 114–126. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2015.

[13] F. V. Fomin and Y. Villanger. Treewidth computation and extremal combinatorics. *Comb.*, 32(3):289–308, 2012.

[14] R. Ganian, S. Ordyniak, and M. S. Ramanujan. Going beyond primal treewidth for (M)ILP. In S. P. Singh and S. 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.

[15] R. Ganian, M. S. Ramanujan, and S. Szeider. Combining treewidth and backdoors for CSP. In H. Vollmer and B. 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.

[16] J. Guo, F. Hüffner, and R. Niedermeier. A structural view on parameterizing problems: Distance from triviality. In R. G. Downey, M. R. Fellows, and F. 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, 2004.

[17] E. C. Hols, S. Kratsch, and A. Pieterse. Elimination distances, blocking sets, and kernels for vertex cover. In C. Paul and M. Bläser, editors, *37th International Symposium on Theoretical Aspects of Computer Science, STACS 2020, March 10-13, 2020, Montpellier, France*, volume 154 of *LIPIcs*, pages 36:1–36:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[18] A. Lindermayr, S. Siebertz, and A. Vigny. Elimination distance to bounded degree on planar graphs. In J. Esparza and D. Král', editors, *45th International Symposium on Mathematical Foundations of Computer Science, MFCS 2020, August 24-28, 2020, Prague, Czech Republic*, volume 170 of *LIPIcs*, pages 65:1–65:12. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[19] D. Lokshtanov, M. S. Ramanujan, S. Saurabh, and M. Zehavi. Reducing CMSO model checking to highly connected graphs. In I. Chatzigiannakis, C. Kaklamanis, D. Marx, and D. 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.